Topology configuration variables include three subsets:

<table>
<thead>
<tr>
<th>Cleanup variables</th>
<th>Properties for cleanup models.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cleanup action variables</td>
<td>Properties for cleanup actions.</td>
</tr>
<tr>
<td>Topology variables</td>
<td>Properties for topologies.</td>
</tr>
</tbody>
</table>

To allocate a set of configuration variables, regardless of which subset you will be using, use `tpm_varalloc`, which returns an `ade_id` for the set that it creates.

To free a set of configuration variables, use `tpm_varfree`.

To get the value of a configuration variable, use `tpm_varget`.

To set the value of a configuration variable, use `tpm_varset`.

To get the properties of a given cleanup action, first allocate a new set of variables, and then use `tpm_cleanactionlistgetat`.

To get the properties of a given topology, first allocate a new set of variables, and then use `tpm_infobuildvar`.

To list all the values of a given set of variables, use `tpm_varlist`. 
Data Extension: Class-Based API

The API for data extension covers the following areas of functionality:

- Queries, including Query Libraries and Range Tables
- Drawing Sets, including Drawings and Drive Aliases
- Object Data
- Operations on Queried Objects, including Property Alteration
- Application Options
- User Management
- Coordinate Transformation

<table>
<thead>
<tr>
<th>Object Model</th>
<th>Containment relationships.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inheritance</td>
<td>Inheritance relationships.</td>
</tr>
<tr>
<td>Classes</td>
<td>Classes sorted by functional group.</td>
</tr>
<tr>
<td>Functions</td>
<td></td>
</tr>
<tr>
<td>Types</td>
<td></td>
</tr>
<tr>
<td>Enumerations</td>
<td></td>
</tr>
<tr>
<td>Error Codes and Error Types</td>
<td></td>
</tr>
</tbody>
</table>
Containment relationships.

- `AcMapSession (Model Root)`
- `AcMapProject`
- `AcMapODContainer`
- `AcMapQuery`
Inheritance relationships.

Query Classes Boundary Classes
Data Extension Classes

Session Classes
acmapsession acmapsessionoptionsreactor
acmapsessionreactor

Project Classes
acmapproject
acmapprojectiterator
acmapprojectoptionsreactor

Drive-Alias Classes
acmapaliases
acmapaliasessreactor
acmapdrivealias

Drawing Set Classes
acmapattacheeddrawing
acmapdrawingset
acmapdrawingsetreactor

Drawing Object ID (Entity ID) Class
acmapobjectid

Expression Class
acmapexpression

Object Data Classes
acmappodcolumnndefinition
acmappodcontainer
acmappodrecorditerator
acmappodtable
acmappodtabledefinition
acmappodtablerecord
acmapvalue

Query Classes
acmapquery
acmapqueryattribute
acmapquerybranch
acmapquerycategory
acmapqueryunit
acmapreporttemplate
acmaptemplateline
acmapsaveset

**Query Library Classes**
acmapquerylibrary
acmapquerylibraryreactor

**Query Condition Classes**
acmapquerycondition
acmapdatacondition
acmaplocationcondition
acmappropertycondition
acmapsqcondition

**Query Boundary Classes**
acmapallboundary
acmapbufferfenceboundary
acmapbufferpolylineboundary
acmapcircleboundary
acmapclosedpolylineboundary
acmapfenceboundary
acmaplocationboundary
acmappointboundary
acmappolygonboundary
acmappolylineboundary
acmapwindowboundary

**Property Alteration Classes**
acmappropertyalteration
acmappropertyalterationdefinition
acmaphatchalteration
acmaptextalteration

**Range Table Classes**
acmaprangelibrary
acmaprangeline
acmaprangetable

**Error Classes**
acmaperrorentry
acmaperrorparameter
acmaperrorstack

**Utility Classes**
acmapobjarray
acmapobjptrarray
acmapstringarray
Data Extension Functions

AcMapGetSession
**Data Extension Types**

**AcMapId**

typedef unsigned long AcMapId;

*File*  MapConstants.h

**AcMapObjectIdArray**

typedef AcMapObjArray&lt;AcMapObjectId&gt; AcMapObjectIdArray;

*File*  MapArxApi.h

**AcMapOperandArray**

typedef AcMapObjPtrArray&lt;AcMapQueryUnit&gt; AcMapOperandArray;

*File*  MapQuery.h

**AcMapVertexArray**

typedef AcMapObjArray&lt;AcMapPolylineVertex&gt; AcMapVertexArray;

*File*  MapBoundary.h
Enumeration members of struct AcMap.

EAdeDwgStatus  EAdeDwgUpdateStatus
EAlterationType
EClassId
EConditionOperator
EDataQueryType
EDataType
EErrCode
EErrType
EJoinOperator
ELocationType
EOpenMode
EPrefType
EPreviewDefinitionsFrom
EProjectOptionType
EPropertyType
EQueryDialogOptions
EQueryType
ERangeOperator
ESaveQueryOptions
ETableType
EUserRights
SaveSetObjectType
Data Extension Error Codes and Error Types

The error enumerations are members of struct `AcMap`.

`EErrCode` `EErrType`
Data Extension Global-Function API (Deprecated)

Except for coordinate transformation functions, the global function API for data extension is deprecated in favor of the class-based API.

The API for data extension covers the following areas of functionality:

- Queries, including Query Libraries and Range Tables
- Drawing Sets, including Drawings and Drive Aliases
- Object Data
- Operations on Queried Objects, including Property Alteration
- Application Options
- User Management
- Coordinate Transformation

<table>
<thead>
<tr>
<th>Data Extension Function Catalog</th>
<th>Data extension functions sorted by name.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Extension Function Synopsis</td>
<td>Data extension functions sorted by functional group.</td>
</tr>
<tr>
<td>Data Extension Types</td>
<td></td>
</tr>
<tr>
<td>Data Extension Constants</td>
<td></td>
</tr>
</tbody>
</table>
Data Extension Function Catalog

Data Extension Global-Function API

Data Extension functions sorted by name.

Note Except for coordinate transformation functions, the global function API is for data extension is deprecated in favor of the class-based API.

alias | altp | ds | dwg | edit | ent | err | expr | key | od
os | pref | proj | ql | qry | rt | save | sql | ss | user | ver

ade_aliasadd ade_aliasdelete
ade_aliasgetlist
ade_aliasupdate

ade_altpclear
ade_altpdefine
ade_altpdelprop
ade_altpgetprop
ade_altpplist
ade_altpsetprop

ade_dsattach
ade_dsdetach
ade_dsisnested
ade_dslist
ade_dsproplist

ade_dwgactivate
ade_dwgactualpath
ade_dwgaliaspath
ade_dwgattriblist
ade_dwgdeactivate
ade_dwggetid
ade_dwggetsetting
ade_dwghaslocks
ade_dwgindex
ade_dwgindexdef
ade_dwgisactive
ade_qrylist
ade_qrysave
ade_qrysetaltprop
ade_qrysetcond
ade_qrysetretransform
ade_qrysettype
ade_qryungroup
ade_rtdfrange
ade_rtdeltable
ade_rtdgetid
ade_rtdgetprop
ade_rttlist
ade_saveobjs
ade_savetodwg
ade_sqlgetenvstring
ade_ssfree
ade_userget
ade_usergetrights
ade_userlist
ade_userset
ade_usersetrights
ade_version
Data Extension Function Synopsis

Data Extension Global-Function API

Data Extension functions sorted by functional group.

**Note** Except for coordinate transformation functions, the global function API for data extension is deprecated in favor of the class-based API.

<table>
<thead>
<tr>
<th>Coordinate Transformation Functions</th>
<th>Drawing Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Drawing Set Functions</td>
<td></td>
</tr>
<tr>
<td>Drive Alias Functions</td>
<td></td>
</tr>
<tr>
<td>Error Message Functions</td>
<td></td>
</tr>
<tr>
<td>Expression Evaluation Function</td>
<td></td>
</tr>
<tr>
<td>Object Data Functions</td>
<td></td>
</tr>
<tr>
<td>Object Editing Functions</td>
<td></td>
</tr>
<tr>
<td>Object Saving Functions</td>
<td></td>
</tr>
<tr>
<td>Option Functions</td>
<td></td>
</tr>
<tr>
<td>Property Alteration Functions</td>
<td></td>
</tr>
<tr>
<td>Query Functions</td>
<td></td>
</tr>
<tr>
<td>Query Library Functions</td>
<td></td>
</tr>
<tr>
<td>Range Table Functions</td>
<td></td>
</tr>
<tr>
<td>SQL Environment Functions</td>
<td></td>
</tr>
<tr>
<td>User Security Functions</td>
<td></td>
</tr>
<tr>
<td>Other Functions</td>
<td></td>
</tr>
</tbody>
</table>
Data Extension Types

Data Extension Global-Function API

**ade_boolean**

typedef unsigned short ade_boolean;

**ade_id**

typedef double ade_id;

**File**

AdeAds.h
Data Extension Constants

**Data Extension Global-Function API**

<table>
<thead>
<tr>
<th>Constant</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADEMEMERROR</td>
<td>-1001</td>
</tr>
<tr>
<td>ADEINVERROR</td>
<td>-1002</td>
</tr>
<tr>
<td>ADEERROR</td>
<td>-1003</td>
</tr>
<tr>
<td>ADE_REALFAIL</td>
<td>-1.0</td>
</tr>
<tr>
<td>ADE_NULLID</td>
<td>0</td>
</tr>
<tr>
<td>ADE_TRUE</td>
<td>1</td>
</tr>
<tr>
<td>ADE_FALSE</td>
<td>0</td>
</tr>
</tbody>
</table>

**File**

AdeAds.h
Annotation

- Namespaces

   AcMapAnnotationManager  Manages annotation features.
Manages annotation features. For more information, search for *annotation* in AutoCAD Map Help.

- **Enumerations**
  - `eAnnotationExpressionFields`: Enumerates annotation expression fields.
  - `eInsertPrecedenceMode`: Enumerates annotation insert precedence modes.

- **Functions**
  - `AnnotationBlockReferenceAssociatedObjectId`: Returns the object ID of the entity that an annotation reference references.
  - `AnnotationTemplateBlockDefinitionId`: Returns an annotation template's block table record.
  - `AnnotationTemplateExists`: Determines whether an annotation template exists.
  - `AnnotationTemplateReferencedObjIds`: Lists all annotation references of an annotation template in the current drawing.
  - `CreateAnnotationTemplate`: Creates an annotation template.
  - `CreateAnnotationText`: Creates a new annotation text object in a template.
  - `DeleteAnnotationTemplate`: Deletes an annotation template.
  - `GetExpressionString`: Retrieves an expression that is stored with an annotation template, annotation reference, or annotation text object.
  - `GetTemplateColor`: Retrieves the block color property of an annotation template. See also `SetTemplateColor()`.
GetTemplateLayer
Retrieves the block layer property of an annotation template. See also SetTemplateLayer().

GetTemplateLinetype
Retrieves the block linetype property of an annotation template. See also SetTemplateLinetype().

GetTemplateLineWeight
Retrieves the block lineweight property of an annotation template. See also SetTemplateLineWeight().

GetTemplateNames
Lists all the annotation templates defined in the current drawing.

GetTemplateRotation
Retrieves the block rotation property of an annotation template. See also SetTemplateRotation().

GetTemplateScaleFactor
Retrieves the block scale factor property of an annotation template. See also SetTemplateScaleFactor().

InsertAnnotationReference
Attaches an annotation reference to an associated entity.

InsertAnnotationReference
Attaches an annotation reference to an associated entity.

InsertAnnotationReference
Attaches an annotation reference to an associated entity.

InsertAnnotationReference
Attaches an annotation reference to an associated entity.

InsertAnnotationReferences
Attaches an annotation reference to one or more
**InsertAnnotationReferences**
Attaches an annotation reference to one or more associated entities.

**IsAnnotationBlockReference**
Determines whether a block reference is an annotation reference.

**IsAnnotationTemplate**
Determines whether a block table record is an annotation template.

**IsAnnotationTemplateReferenced**
Determines whether an annotation template has any annotation references.

**IsAnnotationText**
Determines whether an attribute definition is an annotation text entity.

**RefreshAnnotationReferences**
Refreshes all annotation references in the current drawing that refer to the specified annotation template.

**SetAnnotationInsertPrecedenceMode**
Sets the annotation insert precedence mode.

**SetExpressionString**
Sets an expression string to store with an annotation template, annotation reference, or annotation text object.

**SetTemplateColor**
Sets the block color property of an annotation template. See also GetTemplateColor().

**SetTemplateLayer**
Sets the block layer property of an annotation template. See also GetTemplateLayer().
<table>
<thead>
<tr>
<th>Function Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| **SetTemplateLinetype** | Sets the block linetype property of an annotation template. See also `GetTemplateLinetype()`.
| **SetTemplateLineWeight** | Sets the block linewidth property of an annotation template. See also `GetTemplateLineWeight()`.
| **SetTemplateRotation** | Sets the block rotation property of an annotation template. See also `GetTemplateRotation()`.
| **SetTemplateScaleFactor** | Sets the block scale factor property of an annotation template. See also `GetTemplateScaleFactor()`.
| **TemplateNameBTRPrefix** | Returns the prefix that is prepended to all annotation block table records internally.
| **UpdateAnnotationReferences** | Updates all annotation references in the current drawing that refer to the specified annotation template.

**Structures**

<table>
<thead>
<tr>
<th>Structure Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| **AnnotationOverrides** | Structure that holds all static-property and expression-string overrides used for creating or updating annotations.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Returns an annotation template's block table record.

```cpp
AcDbObjectId AnnotationTemplateBlockDefinitionId(
    const ACHAR * pszTemplateName
);
```

**Parameters**

- `pszTemplateName`: Input name of the template to examine.

**Returns**

Returns the object ID of the template's AcDbBlockTableRecord if the template exists; otherwise, returns AcDbObjectId::kNull.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Determines whether an annotation template exists.

```c
bool AnnotationTemplateExists(
    const ACHAR * pszTemplateName
);
```

**File**

`AcMapAnnotationManager.h`

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pszTemplateName</td>
<td>Input name of the template to find.</td>
</tr>
</tbody>
</table>

**Returns**

Returns true if the template is found in the current drawing; otherwise, returns false.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Lists all annotation references of an annotation template in the current drawing.

```cpp
Acad::ErrorStatus AnnotationTemplateReferencedObjIds(
    AcDbObjectIdArray & objectIdArray,
    const ACHAR * pszTemplateName
);
```

**File**

AcMapAnnotationManager.h

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>objectIdArray</td>
<td>Output array of object IDs of all the specified template's annotation references.</td>
</tr>
<tr>
<td>pszTemplateName</td>
<td>Input name of the template to examine.</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns an error code.

Created with a commercial version of [Doc-O-Matic](https://www.doc-o-matic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapAnnotationManager Namespace

AcMapAnnotationManager:: CreateAnnotationTemplate Function
AcMapAnnotationManager Namespace

Creates an annotation template.

```
AcDbObjectId CreateAnnotationTemplate(
    const ACHAR * pszTemplateName
);
```

File

AcMapAnnotationManager.h

Parameters Description
pszTemplateName Input name of the template to create.

Returns

Returns the object ID of the newly created template if successful; otherwise, returns AcDbObjectId::kNull if an error occurs.

Remarks

If a template with the specified name already exists, it is not overwritten; instead, the null object AcDbObjectId::kNull is returned.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Deletes an annotation template.

```cpp
Acad::ErrorStatus DeleteAnnotationTemplate(
    const ACHAR * pszTemplateName
);
```

**File**

AcMapAnnotationManager.h

**Parameters**

<table>
<thead>
<tr>
<th>Description</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>pszTemplateName</td>
<td>Input name of the template to delete.</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns an error code.

**Remarks**

You can delete a template only if it has no annotation references.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves an expression that is stored with an annotation template, annotation reference, or annotation text object.

```cpp
Acad::ErrorStatus GetExpressionString(
    ACHAR*& pszExpressionString,
    const AcDbObject* pObj,
    AcMapAnnotationManager::eAnnotationExpressionFields field
);
```

### Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pszExpressionString</td>
<td>Output expression string stored with the object. The caller must free this object explicitly, typically with acutDelString().</td>
</tr>
<tr>
<td>pObj</td>
<td>Input pointer to the AcDbBlockTableRecord (template), AcDbBlockReference (reference), or AcDbAttributeDefinition (text) to retrieve the</td>
</tr>
<tr>
<td></td>
<td>expression from.</td>
</tr>
<tr>
<td>field</td>
<td>Input <code>eAnnotationExpressionFields</code> value of the expression to retrieve. For annotation templates and annotation references, use only kBlockXxx codes; for annotation text entities, use only kAttDefXxx codes.</td>
</tr>
</tbody>
</table>

### Returns

Returns Acad::eOk if successful; otherwise, returns an error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the block color property of an annotation template. See also SetTemplateColor().

```cpp
Acad::ErrorStatus GetTemplateColor(
    AcCmColor & color,
    const ACHAR * pszTemplateName
);```

**Parameters**

- **color**
  - Description: Output template block color value. This value defaults to the current entity color when a template is created.
- **pszTemplateName**
  - Description: Input name of the template to examine.

**Returns**

Returns Acad::eOk if successful; otherwise, returns an error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the block layer property of an annotation template. See also SetTemplateLayer().

Acad::ErrorStatus GetTemplateLayer(
   ACHAR *& pszLayer,
   const ACHAR * pszTemplateName
);

File

AcMapAnnotationManager.h

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pszLayer</td>
<td>Output template block layer value. This value defaults to the current entity layer when a template is created. The caller must free this object explicitly, typically with acutDelString().</td>
</tr>
<tr>
<td>pszTemplateName</td>
<td>Input name of the template to examine.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns an error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the block linetype property of an annotation template. See also SetTemplateLinetype().

```cpp
Acad::ErrorStatus GetTemplateLinetype(
    ACHAR *&pszLinetype,
    const ACHAR *pszTemplateName
);
```

File

AcMapAnnotationManager.h

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pszLinetype</td>
<td>Output template block linetype value. This value defaults to the current entity linetype when a template is created. The caller must free this object explicitly, typically with acutDelString().</td>
</tr>
<tr>
<td>pszTemplateName</td>
<td>Input name of the template to examine.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns an error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the block lineweight property of an annotation template. See also SetTemplateLineWeight().

```cpp
Acad::ErrorStatus GetTemplateLineWeight(
    AcDb::LineWeight & lineweight,
    const ACHAR * pszTemplateName
);
```

File

AcMapAnnotationManager.h

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>lineweight</td>
<td>Output template block lineweight value. This value defaults to the current entity lineweight when a template is created.</td>
</tr>
<tr>
<td>pszTemplateName</td>
<td>Input name of the template to examine.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns an error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Lists all the annotation templates defined in the current drawing.

```cpp
Acad::ErrorStatus GetTemplateNames(
    AcArray<ACHAR *> & pszTemplateNameArray
);
```

**File**

AcMapAnnotationManager.h

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pszTemplateNameArray</td>
<td>Output array of the name of each annotation template defined in the current drawing. The caller must free the memory allocated to this array's template names before destroying the array.</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns an error code.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the block rotation property of an annotation template. See also SetTemplateRotation().

```cpp
Acad::ErrorStatus GetTemplateRotation(
    double& dRotation,
    const ACHAR * pszTemplateName
);
```

File

AcMapAnnotationManager.h

Parameters

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dRotation</td>
<td>Output template block rotation value, expressed in radians. This value defaults to 0.0 when a template is created.</td>
</tr>
<tr>
<td>pszTemplateName</td>
<td>Input name of the template to examine.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns an error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the block scale factor property of an annotation template. See also
SetTemplateScaleFactor().

```c++
Acad::ErrorStatus GetTemplateScaleFactor(
    double & dScalefactor,
    const ACHAR * pszTemplateName)
);
```

File

AcMapAnnotationManager.h

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dScalefactor</td>
<td>Output template block scale factor value. This value defaults to 1.0 when a template is created.</td>
</tr>
<tr>
<td>pszTemplateName</td>
<td>Input name of the template to examine.</td>
</tr>
</tbody>
</table>

Returns Acad::eOk if successful; otherwise, returns an error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Attaches an annotation reference to an associated entity.

```cpp
Acad::ErrorStatus InsertAnnotationReference(
    AcDbObjectId& newBlockReferenceId,
    const ACHAR * pszTemplateName,
    const AcDbObjectId assocEnt,
    const AcDbBlockReference * pMatchThisBlockReference = NULL);
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>newBlockReferenceId</code></td>
<td>Input/Output object ID of the newly created annotation reference. If <code>newBlockReferenceId</code> is <code>AcDbObjectId::kNull</code>, then a new annotation reference is created. If not, then the <code>AcDbBlockReference</code> whose Object ID is <code>newBlockReferenceId</code> is reused.</td>
</tr>
<tr>
<td><code>pszTemplateName</code></td>
<td>Input name of the annotation template to attach.</td>
</tr>
<tr>
<td><code>assocEnt</code></td>
<td>Input object ID of the entity to which the reference will be attached.</td>
</tr>
<tr>
<td><code>pMatchThisBlockReference</code></td>
<td>Input optional pointer to an existing annotation reference, whose static-property and expression-string overrides are used to create the new annotation reference.</td>
</tr>
</tbody>
</table>

**Returns**

Returns `Acad::eOk` if successful; otherwise, returns an error code.

**Remarks**

This function takes a named annotation template and an associated entity object ID. The new block reference object ID is an output parameter. If the caller
provides an optional pointer to an existing annotation reference, that reference's static-property and expression-string overrides are used to create the new annotation reference.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Attaches an annotation reference to an associated entity.

```cpp
Acad::ErrorStatus InsertAnnotationReference(
    AcDbObjectId& newBlockReferenceId,
    const ACHAR * pszTemplateName,
    const AcDbObjectId assocEnt,
    const AnnotationOverrides & overrides
);
```

### Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>newBlockReferenceId</td>
<td>Input/Output object ID of the newly created annotation reference. If newBlockReferenceId is AcDbObjectId::kNull, then a new annotation reference is created. If not, then the AcDbBlockReference whose Object ID is newBlockReferenceID is reused.</td>
</tr>
<tr>
<td>pszTemplateName</td>
<td>Input name of the annotation template to attach.</td>
</tr>
<tr>
<td>assocEnt</td>
<td>Input object ID of the entity to which the reference will be attached.</td>
</tr>
<tr>
<td>overrides</td>
<td>Input AnnotationOverrides struct containing the static-property and expression-string overrides to be used when creating the new annotation reference. In the AnnotationOverrides structure, pass a NULL for each individual property that you do not want to override; instead, these property values will be computed from the template's corresponding expression or static property.</td>
</tr>
</tbody>
</table>

### Returns

Returns Acad::eOk if successful; otherwise, returns an error code.

### Remarks
This function takes a named annotation template and an associated entity object ID. The new block reference object ID is an output parameter. The static-property and expression-string overrides of the input structure are used to create the new annotation reference (unless a NULL prevents an override).

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Attaches an annotation reference to one or more associated entities.

```cpp
Acad::ErrorStatus InsertAnnotationReferences(
    AcDbObjectIdArray& newBlockReferenceIds,
    const ACHAR * pszTemplateName,
    const AcDbObjectIdArray& assocEntArray,
    const AcDbBlockReference * pMatchThisBlockReference = NULL
);
```

### File

AcMapAnnotationManager.h

### Parameters

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>newBlockReferenceIds</strong></td>
</tr>
<tr>
<td>Output array of object IDs of the newly created annotation references.</td>
</tr>
<tr>
<td><strong>pszTemplateName</strong></td>
</tr>
<tr>
<td>Input name of the annotation template to attach.</td>
</tr>
<tr>
<td><strong>assocEntArray</strong></td>
</tr>
<tr>
<td>Input array of object IDs of the entities to which the references will be attached.</td>
</tr>
<tr>
<td><strong>pMatchThisBlockReference</strong></td>
</tr>
<tr>
<td>Input optional pointer to an existing annotation reference, whose static-property and expression-string overrides are used to create the new annotation reference.</td>
</tr>
</tbody>
</table>

### Returns

Returns Acad::eOk if successful; otherwise, returns an error code.

### Remarks

This function takes a named annotation template and an array of associated entity object IDs. An array of new block reference object IDs is an output parameter. If the caller provides an optional pointer to an existing annotation reference, that reference's static-property and expression-string overrides are used to create the new annotation references.
Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determines whether a block table record is an annotation template.

```c
bool IsAnnotationTemplate(
    ACHAR*& pszTemplateName,
    AcDbBlockTableRecord* pBTR
);
```

**File**

AcMapAnnotationManager.h

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pszTemplateName</td>
<td>Output name of the template if the specified block table record is an annotation template; otherwise, this parameter is not modified. The caller must free this object explicitly, typically with acutDelString().</td>
</tr>
<tr>
<td>pBTR</td>
<td>Input pointer to the block table record to examine.</td>
</tr>
</tbody>
</table>

**Returns**

Returns true if the block table record is an annotation template in the current drawing; otherwise, returns false.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determines whether an annotation template has any annotation references.

```cpp
bool IsAnnotationTemplateReferenced(
    const ACHAR * pszTemplateName
);
```

File

AcMapAnnotationManager.h

Parameters | Description
-----------|----------------
pszTemplateName | Input name of the annotation template to examine.

Returns

Returns true if the template has any references; otherwise, returns false.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Refreshes all annotation references in the current drawing that refer to the specified annotation template.

```cpp
Acad::ErrorStatus RefreshAnnotationReferences(
   const ACHAR * pszTemplateName,
   bool bFullAnnotation
);
```

**File**

AcMapAnnotationManager.h

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pszTemplateName</td>
<td>Input name of the annotation template to examine.</td>
</tr>
<tr>
<td></td>
<td>Input that specifies how the annotation references are refreshed. If true,</td>
</tr>
<tr>
<td></td>
<td>all expressions of the annotation text entities within the reference(s) are</td>
</tr>
<tr>
<td></td>
<td>re-evaluated. If false, only the annotation text string is re-evaluated,</td>
</tr>
<tr>
<td></td>
<td>causing the annotation text values to change but other properties to remain</td>
</tr>
<tr>
<td></td>
<td>the same.</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns an error code.

**Remarks**

Call this function after an entity with an attached annotation reference changes.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets an expression to store with an annotation template, annotation reference, or annotation text object.

```cpp
Acad::ErrorStatus SetExpressionString(
    const ACHAR* pszExpressionString,
    AcDbObject* pObj,
    AcMapAnnotationManager::eAnnotationExpressionFields field)
);
```

File

AcMapAnnotationManager.h

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pszExpressionString</td>
<td>Input expression to store with the object.</td>
</tr>
<tr>
<td>pObj</td>
<td>Input pointer to the AcDbBlockTableRecord (template), AcDbBlockReference (reference), or AcDbAttributeDefinition (text) to store the expression with.</td>
</tr>
<tr>
<td>field</td>
<td>Input <code>eAnnotationExpressionFields</code> value of the expression to store. For annotation templates and annotation references, use only kBlockXxx codes; for annotation text entities, use only kAttDefXxx codes.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns an error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the block color property of an annotation template. See also GetTemplateColor().

```cpp
Acad::ErrorStatus SetTemplateColor(
    const ACHAR * pszTemplateName,
    AcCmColor color
);
```

File AcMapAnnotationManager.h

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pszTemplateName</td>
<td>Input name of the template whose static property to set.</td>
</tr>
<tr>
<td>color</td>
<td>Input new value for the block color.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns an error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the block layer property of an annotation template. See also GetTemplateLayer().

```cpp
Acad::ErrorStatus SetTemplateLayer(
    const ACHAR * pszTemplateName,
    const ACHAR * pszLayer
);
```

Parameters

- `pszTemplateName`: Input name of the template whose static property to set.
- `pszLayer`: Input new value for the block layer.

Returns

Returns Acad::eOk if successful; otherwise, returns an error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the block linetype property of an annotation template. See also GetTemplateLinetype().

```
Acad::ErrorStatus SetTemplateLinetype(
    const ACHAR * pszTemplateName,
    const ACHAR * pszLinetype
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pszTemplateName</td>
<td>Input name of the template whose static property to set.</td>
</tr>
<tr>
<td>pszLinetype</td>
<td>Input new value for the block linetype.</td>
</tr>
</tbody>
</table>

Returns Acad::eOk if successful; otherwise, returns an error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the block linewidth property of an annotation template. See also GetTemplateLineWeight().

```c
Acad::ErrorStatus SetTemplateLineWeight(
    const ACHAR * pszTemplateName,
    AcDb::LineWeight lineweight
);
```

Parameters

- `pszTemplateName`: Input name of the template whose static property to set.
- `lineweight`: Input new value for the block linewidth.

Returns

Returns Acad::eOk if successful; otherwise, returns an error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the block rotation property of an annotation template. See also
GetTemplateRotation().

\[
\text{Acad::ErrorStatus SetTemplateRotation(}
\begin{align*}
\text{\hspace{1cm} const ACHAR * pszTemplateName,} \\
\text{\hspace{1cm} double dRotation}
\end{align*}
\);
\]

File

\text{AcMapAnnotationManager.h}

Parameters

pszTemplateName

Description

Input name of the template whose static property to set.

dRotation

Description

Input new value for the block rotation, expressed in radians.

Returns

Returns Acad::eOk if successful; otherwise, returns an error code.

Created with a commercial version of \text{Doc-O-Matic}. In order to make this
message disappear you need to register this software. If you have problems
registering this software please contact us at \text{support@toolsfactory.com}.
Sets the block scale factor property of an annotation template. See also GetTemplateScaleFactor().

```
Acad::ErrorStatus SetTemplateScaleFactor(
    const ACHAR * pszTemplateName,
    double dScalefactor
);
```

File

AcMapAnnotationManager.h

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pszTemplateName</td>
<td>Input name of the template whose static property to set.</td>
</tr>
<tr>
<td>dScalefactor</td>
<td>Input new value for the block scale factor.</td>
</tr>
</tbody>
</table>

Returns Acad::eOk if successful; otherwise, returns an error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Updates all annotation references in the current drawing that refer to the specified annotation template.

```cpp
Acad::ErrorStatus UpdateAnnotationReferences(
    const ACHAR * pszTemplateName,
    bool bRetainLocalOverrides
);
```

File

AcMapAnnotationManager.h

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pszTemplateName</td>
<td>Input name of the annotation template to examine.</td>
</tr>
<tr>
<td>bRetainLocalOverrides</td>
<td>Input that specifies the manner in which the annotation references are updated. If true, all override expressions contained in each annotation reference are retained during the update. If false, the local overrides are discarded, and the annotation template's expressions and static properties are used.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns an error code.

Remarks

Call this function after an annotation template changes.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Classification

- **Namespaces**
  - AcMapObjClass: Namespace for classification error codes.

- **Classes**
  - AcMapClassificationManager: Performs general classification operations at the drawing level.
  - AcMapObjClassDefinition: Manages a feature class definition stored in the feature-definition file.
  - AcMapObjClassProperty: Manages a property for a feature class definition.
  - AcMapObjClassReactor: Base class that is notified of classification events.
  - AcMapObjClassSystem: Manages the registration and unregistration of classification reactors.
Links
Classification
AcMapObjClass Namespace
Classification

Namespace for classification error codes. For more information, search for *feature classification* and *feature definitions* in AutoCAD Map Help.

Enumerations

- **EErrCode** Classification error codes.

Created with a commercial version of [Doc-O-Matic](#). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Classification error codes.

```cpp
enum EErrCode {
    eOk = 0x1,
    eFailed,
    eFileAlreadyExists,
    eFileNotFound,
    eFileNameInvalid,
    eNoSchemaFileAttached,
    eSchemaFileAttachedAndFound,
    eSchemaFileAttachedButNotFound,
    eOutOfRange,
    eMissingProperty,
    eClassNotFound,
    eBaseClassNotFound,
    eClassAlreadyExists,
    eClassNameInvalid,
    eClassNameTooLong,
    eBaseClassOnly,
    eUnsupportedEntitytypeName,
    eRxClassNotFound,
    eAlreadyClassified,
    eAlreadyUnclassified,
    eEntityNotClassified,
    ePropertyNotFound,
    ePropertyAlreadyExists,
    eFailedSavingSchema,
    eInvalidType,
    eClassUnsupportedCreateType,
    eProductUnsupportedCreateType,
    eNoUserPrivilegeToAlterSchema,
    eClassNotFromCurrentSchema,
    ePropertyReadOnly
};
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>eOk</td>
<td>The action completed successfully.</td>
</tr>
</tbody>
</table>
eFailed
The action failed.
eFileAlreadyExists
A file with the same name already exists.
eFileNotFound
Could not find the file at the specified location.
eFileNameInvalid
The specified file name is invalid.
eNoSchemaFileAttached
No feature-definition file is attached to the drawing.
eSchemaFileAttachedAndFound
The feature-definition file is attached to the drawing and was found at the specified location.
eSchemaFileAttachedButNotFound
The feature definition-file is attached to the drawing but was not found at the specified location.
eOutOfRange
At least one property value is out of range.
eMissingProperty
At least one property is missing.
eClassNotFound
Could not find the specified feature class in the feature-definition file.
eBaseClassNotFound
Could not find the specified base feature class in the feature-definition file.
eClassAlreadyExists
The specified feature class already exists in the feature-definition file.
eClassNameInvalid
The specified feature class name is invalid.
eClassNameTooLong
The specified feature class name must have 256 or fewer characters.
eBaseClassOnly
The specified feature class cannot be used to classify because it is a strict base class.
eUnsupportedEntityType
The feature class does not support the specified entity type.
eRxClassNotFound
No registered AcRxClass corresponds to the specified entity type.
eAlreadyClassified
The entity is already classified, preventing reclassification.
eAlreadyUnclassified
The entity is already unclassified and cannot be unclassified again.
<table>
<thead>
<tr>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>eEntityNotClassified</td>
<td>The entity is unclassified. No classification-related action can be performed on this entity.</td>
</tr>
<tr>
<td>ePropertyNotFound</td>
<td>Could not find the specified property in the feature class definition.</td>
</tr>
<tr>
<td>ePropertyAlreadyExists</td>
<td>The specified property already exists in the feature class definition.</td>
</tr>
<tr>
<td>eFailedSavingSchema</td>
<td>Could not save the feature-definition file.</td>
</tr>
<tr>
<td>eInvalidType</td>
<td>Found an invalid type of property value or an invalid entity type.</td>
</tr>
<tr>
<td>eClassUnsupportedCreateType</td>
<td>The specified feature class definition does not support the create method.</td>
</tr>
<tr>
<td>eProductUnsupportedCreateType</td>
<td>The product does not support the create method for any feature class definition.</td>
</tr>
<tr>
<td>eNoUserPrivilegeToAlterSchema</td>
<td>The current user lacks the privileges to change the feature-definition file.</td>
</tr>
<tr>
<td>eClassNotFromCurrentSchema</td>
<td>The AcMapObjClassDefinition instance comes from a feature-definition file other than the current one. The instance is unusable until the original feature-definition file is reattached to the current drawing.</td>
</tr>
<tr>
<td>ePropertyReadOnly</td>
<td>The property is a read-only property whose value cannot be changed.</td>
</tr>
</tbody>
</table>

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
AcMapClassificationManager

Performs general classification operations at the drawing level. For more information, search for feature classification and feature definitions in AutoCAD Map Help.

AcMapObjClassDefinition

Manages a feature class definition stored in the feature-definition file. Because all information related to a feature class definition is stored in the feature-definition file, it is possible to manage an AcMapObjClassDefinition instance only if the correct feature-definition file is attached to the current drawing. If the feature-definition file is missing, these functions will fail. For more information, search for feature classification and feature definitions in AutoCAD Map Help.

AcMapObjClassProperty

Manages a property for a feature class definition. For more information, search for feature classification and feature definitions in AutoCAD Map Help.

AcMapObjClassReactor

Base class that is notified of classification events. Custom reactors are classes derived from AcMapObjClassReactor. See the AcMapObjClassSystemclass for a description of adding and registering a custom reactor.

AcMapObjClassSystem

Manages the registration and unregistration of classification reactors. Reactors are classes derived from AcMapObjClassReactor.

To add a custom reactor: Derive a custom class AcMapObjClassMyReactor from AcMapObjClassReactor:

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems
registering this software please contact us at support@toolsfactory.com.
Performs general classification operations at the drawing level. For more information, search for *feature classification* and *feature definitions* in AutoCAD Map Help.

```cpp
class AcMapClassificationManager;
```

**File**

`AcMapClassificationManager.h`

**Methods**

- `~AcMapClassificationManager`  
  Destroys an instance of this class.

- `AcMapClassificationManager`  
  Constructs an instance of this class by using the specified AutoCAD database.

- `AttachFeatureDefinitionFile`  
  Attaches an existing feature-definition file to the current drawing.

- `Audit`  
  Examines an entity for out-of-range or missing classified-property values, fixing these values if desired.

- `Audit`  
  Examines multiple entities for out-of-range or missing classified-property values, listing and fixing these values if desired.

- `CanCurrentUserAlterSchema`  
  Determines whether the current user has sufficient privileges to change the classification information in the feature-definition file.

- `Classify`  
  Classifies an entity with a feature class name.

- `Classify`  
  Classifies multiple entities with a feature class name.

- `Unclassify`  
  Unclassifies an entity entirely. Unclassifying entities does not trigger...
<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ClearAllTags</strong></td>
<td>internal transactions. This function differs from Unclassify() because it ignores the feature-definition file that was used to classify the entity. Unclassifies multiple entities entirely. Unclassifying entities does not trigger internal transactions. This function differs from Unclassify() because it ignores the feature-definition file that was used to classify the entities.</td>
</tr>
<tr>
<td><strong>CreateFeatureClassDefinition</strong></td>
<td>Creates a new feature class definition and adds it to the feature-definition file attached to the current drawing.</td>
</tr>
<tr>
<td><strong>CreateFeatureClassDefinition</strong></td>
<td>Creates a new feature class definition and adds it to the feature-definition file attached to the current drawing.</td>
</tr>
<tr>
<td><strong>CreateFeatureDefinitionFile</strong></td>
<td>Creates a new feature-definition file and attaches it to the current drawing.</td>
</tr>
<tr>
<td><strong>DeleteFeatureClassDefinition</strong></td>
<td>Deletes a feature class definition from the feature-definition file attached to the current drawing.</td>
</tr>
<tr>
<td><strong>DetachCurrentFeatureDefinitionFile</strong></td>
<td>Detaches the current feature-definition file from the current drawing.</td>
</tr>
<tr>
<td><strong>DuplicateFeatureClassDefinition</strong></td>
<td>Creates a copy of an existing feature class definition.</td>
</tr>
<tr>
<td><strong>GetAllTags</strong></td>
<td>Lists all the classification tags of an entity.</td>
</tr>
<tr>
<td><strong>GetClassifiedEntities</strong></td>
<td>Lists all the classified entities in the current drawing.</td>
</tr>
<tr>
<td><strong>GetClassifiedEntities</strong></td>
<td>Lists all the classified entities in the current drawing for a feature class name.</td>
</tr>
<tr>
<td><strong>GetClassifiedProperties</strong></td>
<td>Lists all the classified properties and their values of an entity.</td>
</tr>
<tr>
<td><strong>GetClassifiedProperties</strong></td>
<td>Lists all the classified properties and their values of an entity.</td>
</tr>
<tr>
<td><strong>GetFeatureClassDefinition</strong></td>
<td>Retrieves an existing feature class definition.</td>
</tr>
<tr>
<td>Method Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>GetFeatureClassDefinitionCount</td>
<td>Counts the number of feature class definitions in the feature-definition file attached to the current drawing.</td>
</tr>
<tr>
<td>GetFeatureClassNames</td>
<td>Lists all the feature class definition names in the feature-definition file attached to the current drawing.</td>
</tr>
<tr>
<td>GetFeatureDefinitionFileAttached</td>
<td>Retrieves the name of the feature-definition file, if any, attached to the current drawing.</td>
</tr>
<tr>
<td>GetProperties</td>
<td>Lists all the properties and their values of an entity.</td>
</tr>
<tr>
<td>GetUnclassifiedEntities</td>
<td>Lists all the unclassified entities in the current drawing.</td>
</tr>
<tr>
<td>GetUndefinedEntities</td>
<td>Lists all the undefined entities in the current drawing.</td>
</tr>
<tr>
<td>IsClassified</td>
<td>Determines whether an entity is classified.</td>
</tr>
<tr>
<td>IsFeatureClassDefinitionPresent</td>
<td>Determines whether a class name is defined in the feature-definition file attached to the current drawing.</td>
</tr>
<tr>
<td>ReloadCurrentFeatureDefinitionFile</td>
<td>Reloads the feature-definition file attached to the current drawing.</td>
</tr>
<tr>
<td>RenameFeatureClassDefinition</td>
<td>Renames a feature class definition in the feature-definition file attached to the current drawing.</td>
</tr>
<tr>
<td>SaveCurrentFeatureDefinitionFile</td>
<td>Saves the feature-definition file attached to the current drawing.</td>
</tr>
<tr>
<td>SaveCurrentFeatureDefinitionFileAs</td>
<td>Saves with a different name the feature-definition file attached to the current drawing.</td>
</tr>
<tr>
<td>Unclassify</td>
<td>Unclassifies an entity. Unclassifying entities does not trigger internal transactions. Unclassify only entities</td>
</tr>
</tbody>
</table>
Unclassify

classified with the currently attached feature-definition file. If you unclassify an entity that was classified with a different feature-definition file, the corresponding tags will not be changed. To remove all existing tags, regardless of which feature-definition file was used, use ClearAllTags().

Unclassifies multiple entities. Unclassifying entities does not trigger internal transactions. Unclassify only entities classified with the currently attached feature-definition file. If you unclassify an entity that was classified with a different feature-definition file, the corresponding tags will not be changed. To remove all existing tags, regardless of which feature-definition file was used, use ClearAllTags().

Unclassify

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Attaches an existing feature-definition file to the current drawing.

\[
\text{AcMapObjClass::EErrCode AttachFeatureDefinitionFile(}
\text{ \ const ACHAR* pszSchemaFileName)};
\]

**Parameters**

<table>
<thead>
<tr>
<th>pszSchemaFileName</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input name of the feature-definition file to attach. If no path is specified, the current path is used.</td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

Returns \text{AcMapObjClass::EErrCode eOk} if successful. Returns \text{AcMapObjClass::EErrCode eFileNameInvalid} if the file name is invalid. Returns \text{AcMapObjClass::EErrCode eFileNotFound} if the file does not exist. Returns \text{AcMapObjClass::EErrCode eFailed} if the process failed for some other reason.

**Remarks**

If a feature-definition file is currently attached, it will be automatically replaced.

Created with a commercial version of \text{Doc-O-Matic}. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Classifies an entity with a feature class name.

```cpp
AcMapObjClass::EErrCode Classify(
    AcDbObjectId& entId,
    const ACHAR* pszClassName,
    bool bIncludeNonConforming,
    bool bClassifyEvenIfAlreadyClassified
);
```

**Parameters**

- `entId` (Input) ID of the entity to classify.
- `pszClassName` (Input) name of the feature class.
- `bIncludeNonConforming` (Input) true to include entities with missing or out-of-range properties, or false to include only conforming properties.
- `bClassifyEvenIfAlreadyClassified` (Input) true to replace the current classification if the entity is already classified, or false to prevent reclassification.

**Returns**

Returns `AcMapObjClass::EErrCode` eOk if successful. Returns `AcMapObjClass::EErrCode` eNoSchemaFileAttached if no feature-definition file is attached to the current drawing. Returns `AcMapObjClass::EErrCode` eClassNameInvalid if the class name is invalid. Returns `AcMapObjClass::EErrCode` eClassNameTooLong if the class name is too long. Returns `AcMapObjClass::EErrCode` eClassNotFound if the class is not in the feature-definition file. Returns `AcMapObjClass::EErrCode` eBaseClassOnly if the class is a strict base class that cannot be used for classification. Returns `AcMapObjClass::EErrCode` eUnsupportedEntityType if the feature class does not support the specified entity type. Returns `AcMapObjClass::EErrCode` eAlreadyClassified if the entity is already classified, preventing reclassification. Returns `AcMapObjClass::EErrCode` eOutOfRange if at least one property value is out of range. Returns `AcMapObjClass::EErrCode` eMissingProperty if the entity is missing at least one property. Returns `AcMapObjClass::EErrCode`
eFailed if the process failed for some other reason.

Remarks

Classifying entities does not trigger internal transactions.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Classifies multiple entities with a feature class name.

```c++
AcMapObjClass::EErrCode Classify(
   AcDbObjectIdArray* paFailedEntIds,
   AcArray<AcMapObjClass::EErrCode>* paFailedErrCodes,
   AcDbObjectIdArray& aEntIds,
   const ACHAR* pszClassName,
   bool bIncludeNonConforming,
   bool bClassifyEvenIfAlreadyClassified
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>paFailedEntIds</td>
<td>Output IDs of failed entities, or NULL if not needed by the caller. The operation traverses aEntIds from beginning to the end, reporting failures.</td>
</tr>
<tr>
<td>paFailedErrCodes</td>
<td>Output error code of each failed entity, or NULL if not needed by the caller. The array index of each error code matches the index of each corresponding failed entity in paFailedEntIds.</td>
</tr>
<tr>
<td>aEntIds</td>
<td>Input IDs of the entities to classify.</td>
</tr>
<tr>
<td>pszClassName</td>
<td>Input name of the feature class.</td>
</tr>
<tr>
<td>bIncludeNonConforming</td>
<td>Input true to include entities with missing or out-of-range properties, or false to include only conforming properties.</td>
</tr>
<tr>
<td>bClassifyEvenIfAlreadyClassified</td>
<td>Input true to replace the current classification if the entity is already classified, or false to prevent reclassification.</td>
</tr>
</tbody>
</table>

Returns

Returns `AcMapObjClass::EErrCode eOk` if all entities are classified successfully. Returns `AcMapObjClass::EErrCode eNoSchemaFileAttached` if no feature-definition file is attached to the current drawing. Returns
AcMapObjClass::EErrCode eClassNameInvalid if the class name is invalid. Returns AcMapObjClass::EErrCode eClassNameTooLong if the class name is too long. Returns AcMapObjClass::EErrCode eClassNotFound if the class is not in the feature-definition file. Returns AcMapObjClass::EErrCode eBaseClassOnly if the class is a strict base class that cannot be used for classification. Returns AcMapObjClass::EErrCode eFailed if the process failed for at least one entity.

Remarks

Classifying entities does not trigger internal transactions.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Creates a new feature class definition and adds it to the feature-definition file attached to the current drawing.

```c
AcMapObjClass::EErrCode CreateFeatureClassDefinition(
    AcMapObjClassDefinition*& pFeatureClassDef,
    const ACHAR* pszName,
    const ACHAR* pszDerivedFromClassName,
    const AcArray<AcRxClass*>& aSupportedEntityTypes,
    const AcMapStringArray& aBlockNames,
    bool bBaseClassOnly
);
```

**Parameters**

- **pFeatureClassDef**
  - Output [AcMapObjClassDefinition](#) feature class.
  - The caller must free this object.

- **pszName**
  - Input feature class name.

- **pszDerivedFromClassName**
  - Input superclass name, or NULL if no base class is needed.

- **aSupportedEntityTypes**
  - Input array of AcRxClass objects defining the supported entity types. This list typically matches the value that AcRxObject::desc() returns.

- **aBlockNames**
  - Input array of supported block names.

- **bBaseClassOnly**
  - Input true to create a strict base class; otherwise, false. You can derive other classes from a strict base class but not classify objects with it.

**Returns**

- Returns [AcMapObjClass::EErrCode](#) eOk if successful. Returns [AcMapObjClass::EErrCode](#) eClassNameInvalid if the class name is invalid.
- Returns [AcMapObjClass::EErrCode](#) eClassNameTooLong if the class name is too long. Returns [AcMapObjClass::EErrCode](#) eNoSchemaFileAttached if no feature-definition file is attached to the current drawing. Returns [AcMapObjClass::EErrCode](#) eClassAlreadyExists if a class with same name already exists. Returns [AcMapObjClass::EErrCode](#) eBaseClassNotFound if the base class is not in the feature-definition file. Returns
AcMapObjClass::EErrCode eUnsupportedEntityType if the feature class does not support the specified entity type. Returns AcMapObjClass::EErrCode eNoUserPrivilegeToAlterSchema if the current user lacks the privileges to change the feature-definition file. Returns AcMapObjClass::EErrCode eFailed if the process failed for some other reason.

Remarks

A feature class definition can support any block reference, or specific block reference names. For any block reference, the corresponding entity type AcDbBlockReference::desc() is input in aSupportedEntityTypes. For specific block reference names, the names are input in aBlockNames.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Creates a new feature class definition and adds it to the feature-definition file attached to the current drawing.

**AcMapObjClass::EErrCode** CreateFeatureClassDefinition(
    *AcMapObjClassDefinition* & pFeatureClassDef,
    *const* ACHAR* pszName,
    *const* ACHAR* pszDerivedFromClassName,
    *const* AcMapStringArray& aSupportedEntityTypes,
    *const* AcMapStringArray& aBlockNames,
    *bool* bBaseClassOnly
);

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pFeatureClassDef</td>
<td>Output <em>AcMapObjClassDefinition</em> feature class. The caller must free this object.</td>
</tr>
<tr>
<td>pszName</td>
<td>Input feature class name.</td>
</tr>
<tr>
<td>pszDerivedFromClassName</td>
<td>Input superclass name, or NULL if no base class is needed.</td>
</tr>
<tr>
<td>aSupportedEntityTypes</td>
<td>Input array of strings defining the supported entity types (such as &quot;AcDbCircle&quot; for circle entities).</td>
</tr>
<tr>
<td>aBlockNames</td>
<td>Input array of supported block names.</td>
</tr>
<tr>
<td>bBaseClassOnly</td>
<td>Input true to create a strict base class; otherwise, false. You can derive other classes from a strict base class but not classify objects with it.</td>
</tr>
</tbody>
</table>

**Returns**

Returns *AcMapObjClass::EErrCode* eOk if successful. Returns *AcMapObjClass::EErrCode* eClassNameInvalid if the class name is invalid. Returns *AcMapObjClass::EErrCode* eClassNameTooLong if the class name is too long. Returns *AcMapObjClass::EErrCode* eNoSchemaFileAttached if no feature-definition file is attached to the current drawing. Returns *AcMapObjClass::EErrCode* eClassAlreadyExists if a class with same name already exists. Returns *AcMapObjClass::EErrCode* eBaseClassNotFound if the
base class is not in the feature-definition file. Returns
AcMapObjClass::EErrCode eUnsupportedEntityType if the feature class does
not support the specified entity type. Returns AcMapObjClass::EErrCode
eNoUserPrivilegeToAlterSchema if the current user lacks the privileges to
change the feature-definition file. Returns AcMapObjClass::EErrCode eFailed if
the process failed for some other reason.

Remarks

A feature class definition can support any block reference, or specific block
reference names. For any block reference, the corresponding entity type
AcDbBlockReference::desc()->name() is input in aSupportedEntityTypes. For
specific block reference names, the names are input in aBlockNames.

Created with a commercial version of Doc-O-Matic. In order to make this
message disappear you need to register this software. If you have problems
registering this software please contact us at support@toolsfactory.com.
Create a new feature-definition file and attaches it to the current drawing.

```cpp
AcMapObjClass::EErrCode CreateFeatureDefinitionFile(const ACHAR* pszSchemaFileName);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pszSchemaFileName</td>
<td>Input name and full path of the feature-definition file to create.</td>
</tr>
</tbody>
</table>

Returns

- Returns **AcMapObjClass::EErrCode** eOk if successful.
- Returns **AcMapObjClass::EErrCode** eFileNameInvalid if the file name is invalid.
- Returns **AcMapObjClass::EErrCode** eFileAlreadyExists if a file with the same name already exists.
- Returns **AcMapObjClass::EErrCode** eNoUserPrivilegeToAlterSchema if the current user lacks the privileges to change the feature-definition file.
- Returns **AcMapObjClass::EErrCode** eFailed if the process failed for some other reason.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Deletes a feature class definition from the feature-definition file attached to the current drawing.

```cpp
AcMapObjClass::EErrCode DeleteFeatureClassDefinition(
    const ACHAR* pszClassName
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pszClassName</td>
<td>Input name of the feature class to delete.</td>
</tr>
</tbody>
</table>

Returns

- `AcMapObjClass::EErrCode` eOk if successful. Returns `AcMapObjClass::EErrCode` eClassNameInvalid if the class name is invalid. Returns `AcMapObjClass::EErrCode` eClassNameTooLong if the class name is too long. Returns `AcMapObjClass::EErrCode` eNoSchemaFileAttached if no feature-definition file is attached to the current drawing. Returns `AcMapObjClass::EErrCode` eClassNotFound if the class is not in the feature-definition file. Returns `AcMapObjClass::EErrCode` eNoUserPrivilegeToAlterSchema if the current user lacks the privileges to change the feature-definition file. Returns `AcMapObjClass::EErrCode` eFailed if the process failed for some other reason.

**Remarks**

Entities classified with the deleted feature class are not updated; instead, they retain their classification but become undefined.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapClassificationManager Class, AcMapClassificationManager Class
AcMapClassificationManager:: DuplicateFeatureClassDefinition Method
AcMapClassificationManager Class | AcMapClassificationManager Class

Creates a copy of an existing feature class definition.

AcMapObjClass::EErrCode DuplicateFeatureClassDefinition(
    AcMapObjClassDefinition* & pFeatureClassDefCopied,
    const ACHAR* pszName,
    const ACHAR* pszCopyName
);  
Parameters Description
pFeatureClassDefCopied Output new AcMapObjClassDefinition feature class
definition. The caller must free this object.
pszName Input name of the feature class definition to copy.
pszCopyName Input name of the new feature class definition.

Returns
Returns AcMapObjClass::EErrCode eOk if successful. Returns
AcMapObjClass::EErrCode eClassNameInvalid if the class name is invalid.
Returns AcMapObjClass::EErrCode eClassNameTooLong if the class name is
too long. Returns AcMapObjClass::EErrCode eNoSchemaFileAttached if no
feature-definition file is attached to the current drawing. Returns
AcMapObjClass::EErrCode eClassNotFound if the feature class is not in the
feature-definition file. Returns AcMapObjClass::EErrCode eClassAlreadyExists
if a class with same name already exists. Returns AcMapObjClass::EErrCode
eNoUserPrivilegeToAlterSchema if the current user lacks the privileges to
change the feature-definition file. Returns AcMapObjClass::EErrCode eFailed if
the process failed for some other reason.

Created with a commercial version of Doc-O-Matic. In order to make this
message disappear you need to register this software. If you have problems
registering this software please contact us at support@toolsfactory.com.
Lists all the classified entities in the current drawing for a feature class name.

```cpp
bool GetClassifiedEntities(
    AcDbObjectIdArray& aEntIds,
    const ACHAR* pszClassName,
    bool bReturnMachingDerivedClassToo
) const;
```

### Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>aEntIds</td>
<td>Output IDs of classified entities.</td>
</tr>
<tr>
<td>pszClassName</td>
<td>Input name of the feature class.</td>
</tr>
<tr>
<td>bReturnMachingDerivedClassToo</td>
<td>Input true to also retrieve entities classified with classes derived from pszClassName, or false to ignore derived classes. Derived-class functionality is available only if a feature-definition file is attached to the current drawing. (The feature-definition file stores the class hierarchy; only the class name resides on the entity.)</td>
</tr>
</tbody>
</table>

### Returns

Returns true if at least one entity ID is returned; otherwise, returns false if no entity ID is returned or the process failed.

### Remarks

An entity is considered to be classified even if the corresponding feature class definition is not in the attached feature-definition file.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Lists all the classified properties and their values of an entity.

```cpp
AcMapObjClass::EErrCode GetClassifiedProperties(
    AcArray<AcMapObjClassProperty>*& aProperties,
    AcArray<VARIANT>*& paValues,
    const AcDbEntity * pEntity,
    const ACHAR* pszClassName
) const;
```

**Parameters**

- **aProperties**: Output array of `AcMapObjClassProperty` properties. The caller must free this object.
- **paValues**: Output array of property values, or NULL if not needed by the caller. The array index of each property value matches the index of each corresponding property in `aProperties`. The caller must free this object.
- **pEntity**: Input entity to examine.
- **pszClassName**: Input name of the feature class.

**Returns**

Returns `AcMapObjClass::EErrCode` eOk if successful. Returns `AcMapObjClass::EErrCode` eClassNameInvalid if the class name is invalid. Returns `AcMapObjClass::EErrCode` eClassNameTooLong if the class name is too long. Returns `AcMapObjClass::EErrCode` eNoSchemaFileAttached if no feature-definition file is attached to the current drawing. Returns `AcMapObjClass::EErrCode` eEntityNotClassified if the entity is unclassified. Returns `AcMapObjClass::EErrCode` eClassNotFound if the entity is classified but the class is not in the feature-definition file. Returns `AcMapObjClass::EErrCode` eFailed if the process failed for some other reason.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Lists all the classified properties and their values of an entity.

```
AcMapObjClass::EErrCode GetClassifiedProperties(
    AcArray<AcMapObjClassProperty>* aProperties,
    AcArray<VARIANT>* paValues,
    const AcDbObjectId& entId,
    const ACHAR* pszClassName
) const;
```

### Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>aProperties</code></td>
<td>Output array of <code>AcMapObjClassProperty</code> properties. The caller must free this object.</td>
</tr>
<tr>
<td><code>paValues</code></td>
<td>Output array of property values, or NULL if not needed by the caller. The array index of each property value matches the index of each corresponding property in <code>aProperties</code>. The caller must free this object.</td>
</tr>
<tr>
<td><code>entId</code></td>
<td>Input ID of the entity to examine.</td>
</tr>
<tr>
<td><code>pszClassName</code></td>
<td>Input name of the feature class.</td>
</tr>
</tbody>
</table>

### Returns

Returns `AcMapObjClass::EErrCode` `eOk` if successful. Returns `AcMapObjClass::EErrCode` `eClassNameInvalid` if the class name is invalid. Returns `AcMapObjClass::EErrCode` `eClassNameTooLong` if the class name is too long. Returns `AcMapObjClass::EErrCode` `eNoSchemaFileAttached` if no feature-definition file is attached to the current drawing. Returns `AcMapObjClass::EErrCode` `eEntityNotClassified` if the entity is unclassified. Returns `AcMapObjClass::EErrCode` `eClassNotFound` if the entity is classified but the class is not in the feature-definition file. Returns `AcMapObjClass::EErrCode` `eFailed` if the process failed for some other reason.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
AcMapClassificationManager:: GetFeatureClassDefinition Method

Retrieves an existing feature class definition.

```
AcMapObjClass::EErrCode GetFeatureClassDefinition(
    AcMapObjClassDefinition*& pFeatureClassDef,
    const ACHAR* pszName
);
```

**Parameters**

- `pFeatureClassDef`: Output `AcMapObjClassDefinition` feature class definition. The caller must free this object.
- `pszName`: Input name of the feature class definition to retrieve.

**Returns**

Returns `AcMapObjClass::EErrCode` `eOk` if successful. Returns `AcMapObjClass::EErrCode` `eClassNameInvalid` if the class name is invalid. Returns `AcMapObjClass::EErrCode` `eClassNameTooLong` if the class name is too long. Returns `AcMapObjClass::EErrCode` `eNoSchemaFileAttached` if no feature-definition file is attached to the current drawing. Returns `AcMapObjClass::EErrCode` `eClassNotFound` if the feature class is not in the feature-definition file. Returns `AcMapObjClass::EErrCode` `eNoUserPrivilegeToAlterSchema` if the current user lacks the privileges to change the feature-definition file. Returns `AcMapObjClass::EErrCode` `eFailed` if the process failed for some other reason.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the name of the feature-definition file, if any, attached to the current drawing.

```c
AcMapObjClass::EErrCode GetFeatureDefinitionFileAttached(  
    ACHAR*& pszSchemaFileName
) const;
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pszSchemaFileName</td>
<td>Output the full pathname of the attached feature-definition file, or NULL if no file is attached. The caller must free this object, typically with acutDelString().</td>
</tr>
</tbody>
</table>

**Returns**

Returns `AcMapObjClass::EErrCode` eSchemaFileAttachedAndFound if the feature-definition file is attached to the current drawing and was found. Returns `AcMapObjClass::EErrCode` eSchemaFileAttachedButNotFound if the feature-definition file is attached to the current drawing but was not found. Returns `AcMapObjClass::EErrCode` eNoSchemaFileAttached if no feature-definition file is attached to the current drawing. Returns `AcMapObjClass::EErrCode` eFailed if the process failed for some other reason.

**Remarks**

It is possible for a feature-definition file to be attached to the current drawing but physically missing from the indicated location (path); use the returned error code to determine the file status.

Created with a commercial version of [Doc-O-Matic](https://www.thecadlibrary.com/doc-o-matic). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Determines whether an entity is classified.

```cpp
bool IsClassified(
    AcMapStringArray& aClassNames,
    const AcDbObjectId& entId,
    const ACHAR* pszSchemaFileName
) const;
```

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>aClassNames</td>
<td>Output array of class names, or empty if the entity is unclassified. This array typically contains one item because the classification feature permits only a single classification for a given feature-definition file.</td>
</tr>
<tr>
<td>entId</td>
<td>Input ID of the entity to examine.</td>
</tr>
<tr>
<td>pszSchemaFileName</td>
<td>Input name of the feature-definition file. This value, which determines the class definition's location, can be the name of the current or a now-detached feature-definition file.</td>
</tr>
</tbody>
</table>

### Returns

Returns true if the entity is classified for the specified feature-definition file; otherwise, returns false if the entity is unclassified.

### Remarks

An entity is considered to be classified even if the corresponding feature class definition is not in the attached feature-definition file.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determines whether a class name is defined in the feature-definition file attached to the current drawing.

```cpp
bool IsFeatureClassDefinitionPresent(
    const ACHAR* pszClassName
) const;
```

Parameters

- `pszClassName`: Input name of the feature class.

Returns

Returns true if the class name is defined; otherwise, returns false if it is undefined or no feature-definition file is attached to the current drawing.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapClassificationManager Class, AcMapClassificationManager Class
AcMapClassificationManager:: RenameFeatureClassDefinition Method
AcMapClassificationManager Class | AcMapClassificationManager Class

Renames a feature class definition in the feature-definition file attached to the current drawing.

**AcMapObjClass::EErrCode** RenameFeatureClassDefinition(
    **const** ACHAR* pszClassName,
    **const** ACHAR* pszNewClassName
);

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pszClassName</td>
<td>Input current name of the feature class.</td>
</tr>
<tr>
<td>pszNewClassName</td>
<td>Input new name of the feature class.</td>
</tr>
</tbody>
</table>

Returns

Returns **AcMapObjClass::EErrCode** eOk if successful. Returns **AcMapObjClass::EErrCode** eClassNotFound if the class is not in the feature-definition file. Returns **AcMapObjClass::EErrCode** eClassAlreadyExists if a class with the same name already exists. Returns **AcMapObjClass::EErrCode** eNoSchemaFileAttached if no feature-definition file is attached to the current drawing. Returns **AcMapObjClass::EErrCode** eClassNameInvalid if the class name is invalid. Returns **AcMapObjClass::EErrCode** eClassNameTooLong if the class name is too long. Returns **AcMapObjClass::EErrCode** eNoUserPrivilegeToAlterSchema if the current user lacks the privileges to change the feature-definition file. Returns **AcMapObjClass::EErrCode** eFailed if the process failed for some other reason.

Remarks

The new name must be a non-empty string with 256 or fewer characters. The valid characters are: a-z A-Z 0-9 $ _ - (including accented characters). Entities classified with the renamed feature class are not updated; instead, they become undefined.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Saves with a different name the feature-definition file attached to the current drawing.

**AcMapObjClass::EErrCode** `SaveCurrentFeatureDefinitionFileAs`(  
`const` ACHAR* `pszSchemaFileName`  
) `const`;

**Parameters**  
`pszSchemaFileName`  
**Description**  
Input name of the new feature-definition file. If no path is specified, the current path is used.

**Returns**

Returns **AcMapObjClass::EErrCode** `eOk` if successful. Returns **AcMapObjClass::EErrCode** `eNoSchemaFileAttached` if no feature-definition file is attached to the current drawing. Returns **AcMapObjClass::EErrCode** `eFileAlreadyExists` if a file with the same name already exists. Returns **AcMapObjClass::EErrCode** `eFailedSavingSchema` if the feature-definition file could not be saved. Returns **AcMapObjClass::EErrCode** `eNoUserPrivilegeToAlterSchema` if the current user lacks the privileges to change the feature-definition file. Returns **AcMapObjClass::EErrCode** `eFailed` if the process failed for some other reason.

**Remarks**

This process simply duplicates the existing file; it does not replace it or attach it to current drawing.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at **support@toolsfactory.com**.
Manages a feature class definition stored in the feature-definition file. Because all information related to a feature class definition is stored in the feature-definition file, it is possible to manage an AcMapObjClassDefinition instance only if the correct feature-definition file is attached to the current drawing. If the feature-definition file is missing, these functions will fail. For more information, search for *feature classification* and *feature definitions* in AutoCAD Map Help.

```cpp
class AcMapObjClassDefinition;
```

### File

AcMapObjClassDefinition.h

### Methods

- **~AcMapObjClassDefinition**
  Destroys an instance of this class.

- **AddProperty**
  Adds a new property to this feature class definition.

- **DeleteProperty**
  Deletes a property from this feature class definition.

- **GetCreateMethod**
  Retrieves the AutoCAD entity type that this feature class definition uses when a digitize process runs for this class.

- **GetCreateMethodName**
  Retrieves the AutoCAD entity type that this feature class definition uses when a digitize process runs for this class.

- **GetDescription**
  Retrieves the description of this feature class definition.

- **GetDirectBaseClassName**
  Retrieves the direct base class name of this feature class definition.

- **GetFeatureDefinitionFile**
  Retrieves the full pathname of the feature-definition file that this feature class definition belongs to.
**GetIconName**
Retrieves the icon name of this feature class definition.

**GetName**
Retrieves the name of this feature class definition.

**GetProperties**
Lists all the classified properties of this feature class definition.

**GetProperty**
Retrieves a property of this feature class definition.

**GetSupportedEntityTypes**
Lists the AutoCAD entity types that this feature class definition supports.

**IsBaseClassOf**
Determines whether this feature class definition is a base class of the specified class, either directly or deeper in the class hierarchy.

**IsBaseClassOnly**
Determines whether this feature class definition is a strict base class.

**IsDerivedClassOf**
Determines whether this feature class definition is derived from the specified class, either directly or from higher in the class hierarchy.

**IsDirectBaseClassOf**
Determines whether this feature class definition is the direct base class of the specified class.

**IsPropertyClassified**
Determines whether a property of this feature class definition is classified.

**IsVisibleInWorkspace**
Determines whether this feature class definition is visible in the AutoCAD Map project workspace.
This function allows a client program to find out whether a link to the external data is going to be preserved to the FDO or the whole record is conveyed to the FDO.

**LinkedDataMovedToFdo**
Determines whether a property of this feature class definition is a link template property and returns the link template name.

**PropertyLinkTemplate**
Sets the AutoCAD entity type that this feature class definition uses when a digitize process runs for this class.

**SetCreateMethod**
Sets the AutoCAD entity type that this feature class definition uses when a digitize process runs for this class.
<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SetCreateMethod</td>
<td>class definition uses when a digitize process runs for this class.</td>
</tr>
<tr>
<td>SetDescription</td>
<td>Sets the description of this feature class definition.</td>
</tr>
<tr>
<td>SetIconName</td>
<td>Sets the icon name of this feature class definition.</td>
</tr>
<tr>
<td>SetLinkedDataMovedToFdo</td>
<td>This function allows a client program to define the behavior of the linked data in process of querying and saving from/to FDO. A link to the external data only will be preserved to the FDO if the boolean parameter is set to true. Otherwise, records will be moved to the FDO. conveyed to the FDO.</td>
</tr>
<tr>
<td>SetName</td>
<td>Sets the name of this feature class definition.</td>
</tr>
<tr>
<td>SetVisibleInWorkspace</td>
<td>Sets this feature class definition to visible or invisible in the AutoCAD Map project workspace.</td>
</tr>
</tbody>
</table>

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at **support@toolsfactory.com**.
Deletes a property from this feature class definition.

```cpp
AcMapObjClass::EErrCode DeleteProperty(
    const AcMapStringArray& aStrParentToSubCategoryNames,
    const ACHAR* pszPropertyName
) const;
```

**Parameters**

- `aStrParentToSubCategoryNames`: Input category of the property. This array must contain the hierarchy of all categories for the property, from the top base category to the last sub-category that is the property's direct category.
- `pszPropertyName`: Input name of the property to delete.

**Returns**

Returns `AcMapObjClass::EErrCode` `eOk` if successful. Returns `AcMapObjClass::EErrCode` `eClassNotFromCurrentSchema` if the class is not from the current feature-definition file. Returns `AcMapObjClass::EErrCode` `ePropertyNotFound` if the property does not exist. Returns `AcMapObjClass::EErrCode` `eFailed` if the process failed for some other reason.

Created with a commercial version of [Doc-O-Matic](https://www.doc-o-matic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the AutoCAD entity type that this feature class definition uses when a
digitize process runs for this class.

```cpp
AcMapObjClass::EErrCode GetCreateMethod(
    AcRxClass*& prxEntityType,
    ACHAR*& pszBlockName
) const;
```

**Parameters**

- `prxEntityType` Output entity type defining the create method, or
  NULL if undefined.
- `pszBlockName` Output block name, or NULL if undefined or no block
  is used. The caller must free this object, typically with
  acutDelString().

**Returns**

Returns `AcMapObjClass::EErrCode` `eOk` if successful. Returns
`AcMapObjClass::EErrCode` `eClassNotFromCurrentSchema` if the class is not
from the current feature-definition file. Returns `AcMapObjClass::EErrCode` `eFailed` if the process failed for some other reason.

**Remarks**

The entity type typically matches the value that `AcRxObject::desc()` returns. If
the feature class definition uses a specific block reference name as a create
method, `AcDbBlockReference::desc()` is output in `prxEntityType` and the name
is output in `pszBlockName`.

Created with a commercial version of [Doc-O-Matic](https://www.dbdoc.com/). In order to make this
message disappear you need to register this software. If you have problems
registering this software please contact us at support@toolsfactory.com.
Retrieve the AutoCAD entity type that this feature class definition uses when a digitize process runs for this class.

\[
\text{AcMapObjClass::EErrCode GetCreateMethodName(}
\begin{align*}
& \text{ACHAR*} & \text{pszEntityType}, \\
& \text{ACHAR*} & \text{pszBlockName}
\end{align*}
\) \text{ const;}
\]

**Parameters**

- **pszEntityType**: Output entity type defining the create method, or NULL if undefined. The caller must free this object, typically with acutDelString().
- **pszBlockName**: Output block name, or NULL if undefined or no block is used. The caller must free this object, typically with acutDelString().

**Returns**

Returns \text{AcMapObjClass::EErrCode eOk} if successful. Returns \text{AcMapObjClass::EErrCode eClassNotFromCurrentSchema} if the class is not from the current feature-definition file. Returns \text{AcMapObjClass::EErrCode eFailed} if the process failed for some other reason.

**Remarks**

The entity type typically matches the value that AcRxClass::name() returns. If the feature class definition uses a specific block reference name as a create method, AcDbBlockReference::desc()->name() is output in pszEntityType and the name is output in pszBlockName.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapObjClassDefinition Class, AcMapObjClassDefinition Class
AcMapObjClassDefinition::GetDescription Method
AcMapObjClassDefinition Class | AcMapObjClassDefinition Class

Retrieves the description of this feature class definition.

```
AcMapObjClass::EErrCode GetDescription( 
    ACHAR*& pszDescription
) const;
```

Parameters

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pszDescription</td>
</tr>
</tbody>
</table>

Returns

Returns `AcMapObjClass::EErrCode eOk` if successful. Returns `AcMapObjClass::EErrCode eClassNotFromCurrentSchema` if the class is not from the current feature-definition file. Returns `AcMapObjClass::EErrCode eFailed` if the process failed for some other reason.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the direct base class name of this feature class definition.

```c
AcMapObjClass::EErrCode GetDirectBaseClassName(
    const ACHAR*& pszBaseClassName
) const;
```

**Parameters**

| pszBaseClassName | Output the name of direct base class. |

**Returns**

Returns `AcMapObjClass::EErrCode` eOk if successful. Returns `AcMapObjClass::EErrCode` eClassNotFromCurrentSchema if the class is not from the current feature-definition file. Returns `AcMapObjClass::EErrCode` eFailed if the process failed for some other reason.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Retrieve the icon name of this feature class definition.

```cpp
AcMapObjClass::EErrCode GetIconName(
    ACHAR*& pszIconName
) const;
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pszIconName</td>
<td>Output full pathname of the icon's bitmap file, or an empty string if no icon name is set (meaning that the class is using the standard icon). The caller must free this object, typically with acutDelString().</td>
</tr>
</tbody>
</table>

Returns

- Returns `AcMapObjClass::EErrCode eOk` if successful.
- Returns `AcMapObjClass::EErrCode eClassNotFromCurrentSchema` if the class is not from the current feature-definition file.
- Returns `AcMapObjClass::EErrCode eFailed` if the process failed for some other reason.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves a property of this feature class definition.

```cpp
AcMapObjClass::EErrCode GetProperty(
    AcMapObjClassProperty* & pProperty,
    const AcMapStringArray& aStrParentToSubCategoryNames,
    const ACHAR* pszPropertyName
) const;
```

### Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pProperty</td>
<td>Output <code>AcMapObjClassProperty</code> property object. The caller must free this object.</td>
</tr>
<tr>
<td>aStrParentToSubCategoryNames</td>
<td>Input category. This array must contain the hierarchy of all categories for the property, from the top base category to the last sub-category that is the property's direct category.</td>
</tr>
<tr>
<td>pszPropertyName</td>
<td>Input name of the property to retrieve.</td>
</tr>
</tbody>
</table>

### Returns

Returns `AcMapObjClass::EErrCode` `eOk` if successful. Returns `AcMapObjClass::EErrCode` `eClassNotFromCurrentSchema` if the class is not from the current feature-definition file. Returns `AcMapObjClass::EErrCode` `ePropertyNotFound` if the property does not exist. Returns `AcMapObjClass::EErrCode` `eFailed` if the process failed for some other reason.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Determines whether this feature class definition is a base class of the specified class, either directly or deeper in the class hierarchy.

```c
AcMapObjClass::EErrCode IsBaseClassOf(
    bool* pbIsBaseClass,
    const ACHAR* pszName
) const;
```

### Parameters
- **pbIsBaseClass**
  - Output true if this feature class definition is a direct or indirect base class of the specified class; otherwise, false.
- **pszName**
  - Input name of the feature class definition to examine.

### Returns
- Returns **AcMapObjClass::EErrCode** eOk if successful. Returns **AcMapObjClass::EErrCode** eClassNotFromCurrentSchema if the class is not from the current feature-definition file. Returns **AcMapObjClass::EErrCode** eFailed if the process failed for some other reason.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at **support@toolsfactory.com**.
Links
AcMapObjClassDefinition Class, AcMapObjClassDefinition Class
AcMapObjClassDefinition:: IsDerivedClassOf Method
AcMapObjClassDefinition Class | AcMapObjClassDefinition Class

Determines whether this feature class definition is derived from the specified class, either directly or from higher in the class hierarchy.

AcMapObjClass:: EErrCode IsDerivedClassOf(
    bool* pbIsDerivedClass,
    const ACHAR* pszName
) const;

Parameters
pbIsDerivedClass Output true if this feature class definition is a descendant of the specified class; otherwise, false.
pszName Input name of the feature class definition to examine.

Returns
Returns AcMapObjClass:: EErrCode eOk if successful. Returns AcMapObjClass:: EErrCode eClassNotFromCurrentSchema if the class is not from the current feature-definition file. Returns AcMapObjClass:: EErrCode eFailed if the process failed for some other reason.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determines whether this feature class definition is the direct base class of the specified class.

```
AcMapObjClass::EErrCode IsDirectBaseClassOf(
   bool* pbIsDirectBaseClass,
   const ACHAR* pszName
) const;
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pbIsDirectBaseClass</td>
<td>Output true if this feature class definition is a direct base class of the specified class; otherwise, false.</td>
</tr>
<tr>
<td>pszName</td>
<td>Input name of the feature class definition to examine.</td>
</tr>
</tbody>
</table>

**Returns**

Returns `AcMapObjClass::EErrCode` eOk if successful. Returns `AcMapObjClass::EErrCode` eClassNotFromCurrentSchema if the class is not from the current feature-definition file. Returns `AcMapObjClass::EErrCode` eFailed if the process failed for some other reason.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determines whether a property of this feature class definition is classified.

```c
AcMapObjClass::EErrCode IsPropertyClassified(
    bool* pbIsPropertyClassified,
    const AcMapStringArray& aStrParentToSubCategoryNames,
    const ACHAR* pszPropertyName
) const;
```

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pbIsPropertyClassified</td>
<td>Output true if the property is classified; otherwise, false.</td>
</tr>
<tr>
<td>aStrParentToSubCategoryNames</td>
<td>Input category of the property. This array must contain the hierarchy of all categories for the property, from the top base category to the last sub-category that is the property's direct category.</td>
</tr>
<tr>
<td>pszPropertyName</td>
<td>Input name of the property to examine.</td>
</tr>
</tbody>
</table>

### Returns

Returns `AcMapObjClass::EErrCode` eOk if successful. Returns `AcMapObjClass::EErrCode` eClassNotFromCurrentSchema if the class is not from the current feature-definition file. Returns `AcMapObjClass::EErrCode` eFailed if the process failed for some other reason.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determines whether a property of this feature class definition is a link template property and returns the link template name.

**AcMapObjClass::EErrCode** PropertyLinkTemplate(
    ACHAR* & pcLinkTemplate,
    const AcMapStringArray& aStrParentToSubCategoryNames,
    const ACHAR* pszPropertyName
) const;

**Parameters**

- **pcLinkTemplate**
  - Description: Output link template name if the property is LinkTemplate related, otherwise NULL.
- **aStrParentToSubCategoryNames**
  - Description: Input category of the property. This array must contain the hierarchy of all categories for the property, from the top base category to the last sub-category that is the property's direct category.
- **pszPropertyName**
  - Description: Input name of the property to examine.

**Returns**

Returns **AcMapObjClass::EErrCode** eOk if successful. Returns **AcMapObjClass::EErrCode** eFailed if the process failed for some other reason.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at **support@toolsfactory.com**.
Sets the AutoCAD entity type that this feature class definition uses when a digitize process runs for this class.

```
AcMapObjClass::EErrCode SetCreateMethod(
    const ACHAR* pszEntityType,
    const ACHAR* pszBlockName
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pszEntityType</td>
<td>Input entity type.</td>
</tr>
<tr>
<td>pszBlockName</td>
<td>Input block name.</td>
</tr>
</tbody>
</table>

**Returns**

Returns `AcMapObjClass::EErrCode` eOk if successful. Returns `AcMapObjClass::EErrCode` eClassNotFromCurrentSchema if the class is not from the current feature-definition file. Returns `AcMapObjClass::EErrCode` eInvalidType if the entity type is invalid. This error code also is returned if pszBlockName is empty or NULL, even if pszEntityTypedef is AcDbBlockReference::desc()->name(); a name is required to define a valid block create method. Returns `AcMapObjClass::EErrCode` eClassUnsupportedCreateType if the feature class definition does not support the create method. Returns `AcMapObjClass::EErrCode` eProductUnsupportedCreateType if the product does not support the create method. Returns `AcMapObjClass::EErrCode` eFailed if the process failed for some other reason.

**Remarks**

The entity type typically matches the value that AcRxClass::name() returns. Invalid types generate an error. The entity types that the digitize process supports are only a subset of the valid entity types, so a type that the feature class definition supports can still be invalid. If a feature class definition uses a specific block reference name as a create method, set pszBlockName to the name and set pszEntityTypedef to NULL.
Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the AutoCAD entity type that this feature class definition uses when a digitize process runs for this class.

```cpp
AcMapObjClass::EErrCode SetCreateMethod(
    const AcRxClass* prxEntityType,
    const ACHAR* pszBlockName
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>prxEntityType</td>
<td>Input entity type.</td>
</tr>
<tr>
<td>pszBlockName</td>
<td>Input block name.</td>
</tr>
</tbody>
</table>

**Returns**

Returns `AcMapObjClass::EErrCode` eOk if successful. Returns `AcMapObjClass::EErrCode` eClassNotFromCurrentSchema if the class is not from the current feature-definition file. Returns `AcMapObjClass::EErrCode` eInvalidType if the entity type is invalid. This error code also is returned if pszBlockName is empty or NULL, even if pszEntityName is AcDbBlockReference::desc(); a name is required to define a valid block create method. Returns `AcMapObjClass::EErrCode` eClassUnsupportedCreateType if the feature class definition does not support the create method. Returns `AcMapObjClass::EErrCode` eProductUnsupportedCreateType if the product does not support the create method. Returns `AcMapObjClass::EErrCode` eFailed if the process failed for some other reason.

**Remarks**

The entity type typically matches the value that AcRxObject::desc() returns. Invalid types generate an error. The entity types that the digitize process supports are only a subset of the valid entity types, so a type that the feature class definition supports can still be invalid. If a feature class definition uses a specific block reference name as a create method, set pszBlockName to the name and set prxEntityName to NULL.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this
message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the description of this feature class definition.

```cpp
AcMapObjClass::EErrCode SetDescription(
    const ACHAR* pszDescription
);
```

Parameters

- **pszDescription**
  - Input description.

Returns

- Returns **AcMapObjClass::EErrCode** eOk if successful.
- Returns **AcMapObjClass::EErrCode** eClassNotFromCurrentSchema if the class is not from the current feature-definition file.
- Returns **AcMapObjClass::EErrCode** eFailed if the process failed for some other reason.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at **support@toolsfactory.com**.
Sets the icon name of this feature class definition.

```c
AcMapObjClass::EErrCode SetIconName(
    const ACHAR* pszIconName
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pszIconName</td>
<td>Input full pathname of the icon's bitmap file, or NULL or an empty string to specify the standard icon.</td>
</tr>
</tbody>
</table>

**Returns**

Returns `AcMapObjClass::EErrCode` `eOk` if successful. Returns `AcMapObjClass::EErrCode` `eClassNotFromCurrentSchema` if the class is not from the current feature-definition file. Returns `AcMapObjClass::EErrCode` `eFailed` if the process failed for some other reason.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the name of this feature class definition.

\texttt{AcMapObjClass::EErrCode \texttt{SetName(}}
\begin{verbatim}
const ACHAR* pszName
\end{verbatim}
\texttt{);}  

\textbf{Parameters} \hspace{1cm} \textbf{Description}

\begin{itemize}
  \item \texttt{pszName} \hspace{1cm} Input class name.
\end{itemize}

\textbf{Returns}

Returns \texttt{AcMapObjClass::EErrCode} \texttt{eOk} if successful. Returns \texttt{AcMapObjClass::EErrCode} \texttt{eClassNotFound} if the class is not in the feature-definition file. Returns \texttt{AcMapObjClass::EErrCode} \texttt{eClassAlreadyExists} if a class with the same name already exists. Returns \texttt{AcMapObjClass::EErrCode} \texttt{eClassNameInvalid} if the class name contains invalid characters. Returns \texttt{AcMapObjClass::EErrCode} \texttt{eClassNameTooLong} if the class name contains more than 255 characters. Returns \texttt{AcMapObjClass::EErrCode} \texttt{eClassNotFromCurrentSchema} if the class is not from the current feature-definition file. Returns \texttt{AcMapObjClass::EErrCode} \texttt{eFailed} if the process failed for some other reason.

\textbf{Remarks}

If you change the name of an existing feature class definition, entities so classified are not updated; instead, they become undefined.

Created with a commercial version of \texttt{Doc-O-Matic}. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at \texttt{support@toolsfactory.com}.
Manages a property for a feature class definition. For more information, search for *feature classification* and *feature definitions* in AutoCAD Map Help.

```c
class AcMapObjClassProperty;
```

**File**

`AcMapObjClassProperty.h`

**Methods**

- `~AcMapObjClassProperty` Destroys an instance of this class.
- `FromString` Generates a value from a string representation.
- `GetCategory` Retrieves the direct category of this property.
- `GetDefaultValue` Retrieves the default value of this property.
- `GetName` Retrieves the name of this property.
- `GetRange` Retrieves the range of valid values for this property.
- `GetType` Retrieves the type of this property.
- `GetValue` Retrieves the value of this property from the specified entity.
- `IsInRange` Determines whether a value falls in this property's range of valid values.
- `IsReadOnly` Determines whether this property is read-only or read-write.
- `IsVisible` Determines whether this property is visible or invisible.
- `SetDefaultValue` Sets the default value of this property.
- `SetRange` Sets or unsets the range of valid values for this property.
- `SetReadOnly` Sets this property to read-only or read-write.
- `SetValue` Sets the value of this property for the specified entity.

The entity and property can be classified or unclassified. If this property is classified, this function checks the property's range of valid values and, if the value is
SetValue

invalid, sets the value to its default value if bFixOutOfRangeIfClassified is true. Properties retrieved with AcMapClassificationManager::GetClassifiedProperties (twoforms) are classified properties. Properties retrieved with AcMapClassificationManager::GetProperties() (twoforms) are unclassified properties, and SetValue() sets property values regardless of the value of bFixOutOfRangeIfClassified.

SetVisible

Sets this property to visible or invisible.

ToString

Returns the string representation of a value.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Generates a value from a string representation.

```c
void FromString(
    VARIANT* pvarValue,
    const ACHAR* pszValue
) const;
```

**Parameters**

- **pvarValue**: Output value. The caller must free this object, typically with VariantClear().
- **pszValue**: Input string.

**Returns**

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets or unsets the range of valid values for this property.

```
AcMapObjClass::EErrCode SetRange(
    const ACHAR* pszRange
);
```

Parameters

- *pszRange*  
  Description
  Input range of valid values, or NULL or "--" to unset the range.

Returns

Returns *AcMapObjClass::EErrCode* eOk if successful. Returns *AcMapObjClass::EErrCode* eClassNotFromCurrentSchema if the property's class is not in the current feature-definition file. Returns *AcMapObjClass::EErrCode* eOutOfRange if the current default value falls outside of the specified range. Returns *AcMapObjClass::EErrCode* eFailed if the process failed for some other reason.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at **support@toolsfactory.com**.
Base class that is notified of classification events. Custom reactors are classes derived from AcMapObjClassReactor. See the AcMapObjClassSystem class for a description of adding and registering a custom reactor.

```cpp
class AcMapObjClassReactor;
```

File

AcMapObjClassReactor.h

- Methods
  - `~AcMapObjClassReactor` Destroys an instance of this class.
  - `AcMapObjClassReactor` Constructs an instance of this class.
  - `FeatureClassDefinitionCreated` Invoked when a feature class definition is created.
  - `FeatureClassDefinitionDeleted` Invoked when a feature class definition is deleted.
  - `FeatureClassDefinitionModified` Invoked when a feature class definition is modified.
  - `FeatureClassDefinitionRenamed` Invoked when a feature class definition is renamed.
  - `FeatureDefinitionFileAttached` Invoked when a feature-definition file is attached to or detached from the current drawing.
  - `FeatureDefinitionFileModified` Invoked when a feature-definition file is modified and saved.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Invoked when a feature class definition is created.

```c
virtual void FeatureClassDefinitionCreated(
    const ACHAR * pszFileName,
    const ACHAR * pszClassName
);
```

Parameters

- `pszFileName` - Input name and full path of the feature-definition file.
- `pszClassName` - Input name of the feature class definition.

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Invoked when a feature class definition is deleted.

```cpp
virtual void FeatureClassDefinitionDeleted(
    const ACHAR * pszFileName,
    const ACHAR * pszClassName
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pszFileName</td>
<td>Input name and full path of the feature-definition file.</td>
</tr>
<tr>
<td>pszClassName</td>
<td>Input name of the feature class definition.</td>
</tr>
</tbody>
</table>

Returns

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapObjClassReactor Class, AcMapObjClassReactor Class
AcMapObjClassReactor:: FeatureClassDefinitionModified Method
AcMapObjClassReactor Class | AcMapObjClassReactor Class

Invoked when a feature class definition is modified.

```cpp
virtual void FeatureClassDefinitionModified(
    const ACHAR * pszFileName,
    const ACHAR * pszClassName
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pszFileName</td>
<td>Input name and full path of the feature-definition file.</td>
</tr>
<tr>
<td>pszClassName</td>
<td>Input name of the feature class definition.</td>
</tr>
</tbody>
</table>

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Invoked when a feature class definition is renamed.

```cpp
virtual void FeatureClassDefinitionRenamed(
    const ACHAR * pszFileName,
    const ACHAR * pszOldClassName,
    const ACHAR * pszNewClassName
);
```

**Parameters**

- **pszFileName**: Input name and full path of the feature-definition file.
- **pszOldClassName**: Input old name of the feature class definition.
- **pszNewClassName**: Input new name of the feature class definition.

**Returns**

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://toolsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Invoked when a feature-definition file is attached to or detached from the current drawing.

```cpp
virtual void FeatureDefinitionFileAttached(
    const ACHAR * pszFileName
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pszFileName</td>
<td>Input name and full path of the feature-definition file that was attached,</td>
</tr>
<tr>
<td></td>
<td>or an empty string if the file was detached.</td>
</tr>
</tbody>
</table>

Returns

Returns nothing.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Invoked when a feature-definition file is modified and saved.

```cpp
virtual void FeatureDefinitionFileModified(
    const ACHAR * pszFileName
);
```

**Parameters**

- **pszFileName**
  
  Description
  
  Input name and full path of the feature-definition file.

**Returns**

Returns nothing.

**Remarks**

Typically, notification occurs at the moment changes are committed to a file by saving it. Note that a file can be saved with no changes, which still triggers notification.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
AcMapObjClassApi.lib

Manages the registration and unregistration of classification reactors. Reactors are classes derived from `AcMapObjClassReactor`.

To add a custom reactor: Derive a custom class `AcMapObjClassMyReactor` from `AcMapObjClassReactor`:

```cpp
class AcMapObjClassMyReactor : public AcMapObjClassReactor
```

Implement events by overriding the virtual functions that you need. Create an instance of the custom reactor:

```cpp
AcMapObjClassMyReactor* pMyReactor = new AcMapObjClassMyReactor;
```

Register it so that it becomes active:

```cpp
AcMapObjClassSystem().AddObjClassReactor(pMyReactor);
```

Write some classification code. Remove the reactor from the list and delete it:

```cpp
AcMapObjClassSystem().RemoveObjClassReactor(pMyReactor);
delete pMyReactor;
```

Note that only one list of reactors exists independently of the drawing, so if you change classification information in a feature-definition file, all the drawings are notified of the event.

```cpp
class AcMapObjClassSystem;
```

File

`AcMapObjClassSystem.h`

- Methods
Destroys an instance of this class.

Constructs an instance of this class.

Adds a reactor to the list of the classification reactors.

Removes a reactor from the list of the classification reactors.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Create Centroids

Namespaces

**AcMapUtilities**

This namespace provides functions that create centroids.
This namespace provides functions that create centroids. Newly created centroids contain the same object data and link template information, if any, of the source entities for which the centroids were created. For more information, search for *centroids* in AutoCAD Map Help.

Functions

- **CreateCentroids**
  Creates centroid blocks inside a set of closed polylines or mpolygons.

- **CreateCentroids**
  Creates centroid blocks inside a set of closed polylines or mpolygons.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
CreateCentroids Function

AcMapUtilities Namespace

Creates centroid blocks inside a set of closed polylines or mpolygons.

```cpp
AcMap::EErrCode CreateCentroids(
    AcDbObjectIdArray& aCentroidIds,
    AcDbObjectIdArray& aClosedPolyIds,
    const ACHAR* pszLayerName,
    const ACHAR* pszBlockName
);
```

### Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>aCentroidIds</td>
<td>Output array that contains the identifiers of all the created centroids.</td>
</tr>
<tr>
<td>aClosedPolyIds</td>
<td>Input array that contains the identifiers of the closed polylines or mpolygons.</td>
</tr>
<tr>
<td>pszLayerName</td>
<td>Input name of the layer to create the centroids on.</td>
</tr>
<tr>
<td>pszBlockName</td>
<td>Input name of block to use for the created centroids.</td>
</tr>
</tbody>
</table>

### Returns

Returns AcMap::kOk if successful. Returns AcMap::kErrBadInput if the named block does not exist. Returns AcMap::kErr if the process failed for some other reason.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Creates centroid blocks inside a set of closed polylines or mpolygons.

AcMap::EErrCode CreateCentroids(
    ads_name& ssCentroids,
    ads_name& ssClosedPolys,
    const ACHAR* pszLayerName,
    const ACHAR* pszBlockName
);

Parameters

<table>
<thead>
<tr>
<th></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ssCentroids</td>
<td>Output name of the ADS (AutoCAD Development System) selection set that contains all the created centroids.</td>
</tr>
<tr>
<td>ssClosedPolys</td>
<td>Input name of the ADS selection set that contains the closed polylines or mpolygons.</td>
</tr>
<tr>
<td>pszLayerName</td>
<td>Input name of the layer to create the centroids on.</td>
</tr>
<tr>
<td>pszBlockName</td>
<td>Input name of block to use for the created centroids.</td>
</tr>
</tbody>
</table>

Returns

Returns AcMap::kOk if successful. Returns AcMap::kErrBadInput if the named block does not exist. Returns AcMap::kErr if the process failed for some other reason.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Custom Object Protocol Extensions

Classes

**AcMapGeometryPE**
A set of functions to analyze custom entities as collections of simple geometry elements (points, polylines, polygons).

**AcMapQueryPE**
A set of functions to analyze custom entities for querying, altering their properties when they are queried, and saving them back to their source drawings.
Links
Custom Object Protocol Extensions

Classes
Custom Object Protocol Extensions

Classes

AcMapGeometryPE

A set of functions to analyze custom entities as collections of simple geometry elements (points, polylines, polygons).

AcMapQueryPE

A set of functions to analyze custom entities for querying, altering their properties when they are queried, and saving them back to their source drawings.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
A set of functions to analyze custom entities as collections of simple geometry elements (points, polylines, polygons).

```cpp
class AcMapGeometryPE : public AcRxObject;
```

Remarks

Valid polygons are closed figures composed of a single contour only. A polygon containing another polygon is read as two polygons, not as a polygon with a hole. Polygons are processed as solid objects during a query. If a geometry element is not a point, polyline, or polygon, its type is unsupported. Unsupported types are ignored.

- **Enumerations**
  - `EGeometryType` Types for simple geometry elements.
- **Methods**
  - `beginRead` Executes before reading an entity begins.
  - `endRead` Executes after reading an entity ends.
  - `getGeometryType` Gets the type of a simple element.
  - `getNumGeometries` Counts simple elements in an entity.
  - `getNumVertices` Counts vertices of a polyline or polygon.
  - `getPoint` Gets a point element.
  - `getVertex` Gets a vertex of a polyline or polygon.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
A set of functions to analyze custom entities for querying, altering their properties when they are queried, and saving them back to their source drawings.

class AcMapQueryPE : public AcRxObject;

File

AcMapQueryPE.h

Enumerations

- ELocationType: Location-query process types.
- EPropType: Property types for AutoCAD Map entities.

Methods

- canQuery: Can an entity be queried?
- canSaveBack: Can the entity be saved back?
- getPointsOnEntity: Get any physical point on the entity to check whether it is inside location boundary or outside.
- getPropertyValue: Gets a property value of an entity.
- hasProperty: Does the entity have a given property?
- isClosed: One more property alteration functions telling whether we need to apply hatch alteration to the entity or not.
- isPropertyReadOnly: Can an entity be changed by property alteration?
- locationQueryType: Specifies how the entity will be processed in a location query.
- onClosedSet: Sends a notification to the entity at the time of closed set calculation (query and save back).
- setPropertyValue: Performs a property alteration on an entity.
- swapIdWith: Swap ID with the prototype entity during saving back.
- transformBy: Transforms the entity (projection transformation during query and save back, and rubber-sheeting).

Created with a commercial version of Doc-O-Matic. In order to make this
message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Can an entity be queried?.

**virtual bool** canQuery(
    AcDbEntity* pEnt = NULL,
    AcDbDatabase* toDb = NULL
);

**Parameters**

- **pEnt**: The entity.
- **toDb**: Database to query to.

**Returns**

Returns true if the entity can be queried.

**Remarks**

entity can be NULL, meaning you are being asked about all entities of this type. database will be also NULL in this case.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Data Sources

- Classes

**AcMapDataSources**

This class provides methods that handle data sources.
This class provides methods that handle data sources. For more information, search for *data sources* in AutoCAD Map Help.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
This class provides methods that handle data sources. For more information, search for *data sources* in AutoCAD Map Help.

```cpp
class AcMapDataSources;
```

File

MapDataSources.h

## Methods

- **~AcMapDataSources** Destroys an instance of this class.
- **AcMapDataSources** Constructs an instance of this class.
- **AttachDataSource** Attaches a data source to the current AutoCAD Map work session.
- **ConnectDataSource** Connects a data source to the current AutoCAD Map work session.
- **DetachAllDataSources** Detaches all data sources from the current AutoCAD Map work session.
- **DetachDataSource** Detaches a data source from the current AutoCAD Map work session.
- **DisconnectAllDataSources** Disconnects all data sources from the current AutoCAD Map work session.
- **DisconnectDataSource** Disconnects a data source from the current AutoCAD Map work session.
- **GetAttachedDataSources** Retrieves the names of the attached data sources in the current AutoCAD Map work session.
- **GetAttachedDataSourcesCount** Counts the number of attached data sources in the current AutoCAD Map work session.
- **GetConnectedDataSources** Retrieves the names of the connected data sources in the current AutoCAD Map work session.
**GetConnectedDataSourcesCount**  Counts the number of connected data sources in the current AutoCAD Map work session.

**GetDisconnectedDataSources**  Retrieves the names of the disconnected data sources in the current AutoCAD Map work session.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Attaches a data source to the current AutoCAD Map work session.

```c
bool AttachDataSource(  
    ACHAR * pszDataSourceName,  
    const ACHAR * pszDataSourcePath
);
```

**Parameters**

- `pszDataSourceName`  
  Description: Output name of the data source that was attached.

- `pszDataSourcePath`  
  Description: Input absolute pathname of the data source to attach. Relative pathnames are not allowed.

**Returns**

Returns true if the data source is attached, or false if it is not attached.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Connects a data source to the current AutoCAD Map work session.

```cpp
bool ConnectDataSource(
    const ACHAR* pszDataSourceName
);
```

**Parameters**

- **pszDataSourceName**: Input name of the data source to connect.

**Returns**

Returns true if the data source is connected, or false if it not connected.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Detaches a data source from the current AutoCAD Map work session.

```cpp
void DetachDataSource(
    const ACHAR* pszDataSourceName
);
```

**Parameters**
- **pszDataSourceName**: Input name of the data source to detach.

**Returns**
- Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Disconnects a data source from the current AutoCAD Map work session.

```cpp
void DisconnectDataSource(
    const ACHAR* pszDataSourceName
);
```

### Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pszDataSourceName</td>
<td>Input name of the data source to disconnect.</td>
</tr>
</tbody>
</table>

### Returns

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Display Manager

### Classes

**AcMapDMAAllDrawOrderItemsIterator**

An iterator over the draw order of elements (AcMapDMElement). The iterator will step on every element, including those which are currently in a proxied state.

**AcMapDMAAllItemsIterator**

An iterator over a collection of display-management items. This iterator will step on every item, including those which may be in a proxy state.

**AcMapDMAAllStyleReferencesIterator**

An iterator over a collection of style references (AcMapDMStyleReferer) within one style set. The iterator may return a StyleId() that points to an AcDbProxyObject.

**AcMapDMAAttachedDwgsQueryDataSourceDescriptor**

Describes a display-management ADE query and provides functions to define queries and manage drawing lists.

**AcMapDMAAttachedDwgsQueryElement**

Represents a display-management ADE query element in an attached drawing.

**AcMapDMCurrentDwgQueryElement**

Represents a query element in the current drawing.

**AcMapDMDataSourceDescriptor**

Represents a query definition's data-source descriptor.

**AcMapDMDefaultStyle**

Represents the default
An abstraction of an AutoCAD Map element that manipulates associated styles. A concrete element provides its own implementation for object selection.

Encapsulates properties that can be applied to entities.

Represents a feature-class data source.

Represents a feature-class element.

Represents a group of display-management items.

Base class of all display-management items.

Represents a layer data source.

Represents a layer element.

Represents a legend object.

Represents the Display Manager map, the root object from which all other Display Manager objects can be obtained by using an iterator.

An iterator over a collection of display-management map objects.

Manages display-management map objects (AcMapDMMap).

Base class used to notify
**AcMapDMMapReactor**

application of display-management events with a display-management map.

Base class used to notify an application of display-management events with an AutoCAD Map project.

**AcMapDMProjectReactor**

**AcMapDMQueryDataSourceDescriptor**

Describes a base data source for querying.

**AcMapDMRasterDataSourceDescriptor**

Defines a raster data source.

**AcMapDMRasterElement**

Represents a raster element that selects raster images.

**AcMapDMRasterStyle**

Encapsulates raster-image-related properties.

Provides an annotation alteration style for stylization.

**AcMapDMSEAnnotationStyle**

Provides a hatch alteration style for stylization.

**AcMapDMSEHatchStyle**

Provides a text alteration style for stylization.

**AcMapDMSETextStyle**

**AcMapDMStyle**

Base class for display-management styles.

**AcMapDMStyleCategory**

Represents a category of styles.

**AcMapDMStyleLibrary**

Represents a library of categories of styles.

**AcMapDMStyleReference**

Represents a reference to existing style.

**AcMapDMStylizationEntityStyle**

Represents a style of a stylization entity.

**AcMapDMThematicStyle**

Represents a thematic stylization.

Defines a display-
AcMapDMThematicTable

management thematic table and its items.

Represents an annotation cell of a thematic definition table.

AcMapDMThematicTableItemAnnotation

Represents a block cell of a thematic definition table.

AcMapDMThematicTableItemBlock

Represents a color cell of a thematic definition table.

AcMapDMThematicTableItemColor

Represents a text/value-pair cell of a thematic definition table.

AcMapDMThematicTableItemDataValue

Represents a hatch cell of a thematic definition table.

AcMapDMThematicTableItemHatch

Represents a legend text cell of a thematic definition table.

AcMapDMThematicTableItemLegendText

Represents a linestyle cell of a thematic definition table.

AcMapDMThematicTableItemLinestyle

Represents a linetype cell of a thematic definition table.

AcMapDMThematicTableItemLinetype

Class representing lineweight cell of thematic definition table.

AcMapDMThematicTableItemLineweight

Represents a plotstyle cell of a thematic definition table.

AcMapDMThematicTableItemPlotstyle

Represents a text cell of a thematic definition table.

AcMapDMThematicTableItemText

Utility class that defines a table row and allows easier access to its column items.

AcMapDMThematicTableRow

AcMapDMTopoElement

Defines a query for all topological elements in the current drawing.
AcMapDMTopoElementDataSourceDescriptor

Defines a topology data-source.

AcMapDMTopoQueryDataSourceDescriptor

Defines a topology data-source.

AcMapDMTopoQueryElement

Defines a query element that acquires topology objects in the source drawing.
Links
Display Manager
AcMapDMScaleFactor Namespace
Display Manager

Provides scale-related information and reactor methods.

- **Functions**
  - `DMAAddScaleFactorReactor` Adds a scale-factor reactor to monitor scale-factor changes.
  - `DMDisplayIsMetric` Determines the display measurement units.
  - `DMDwgUnitsAreMetric` Determines the dwg measurement units.
  - `DMGetCurrentScaleFactor` Retrieves the cached current scale factor.
  - `DMGetUnCachedCurrentScaleFactor` Immediately computes the current scale factor.
  - `DMRefreshScale` Forces the scale factor to update.
  - `DMRemoveScaleFactorReactor` Removes a scale-factor reactor.

- **Classes**
  - `DMScaleFactorReactor` Reactor class for scale-factor changes.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Immediately computes the current scale factor.

\[
\text{double} \hspace{1em} \text{DMGetUnCachedCurrentScaleFactor}(\hspace{1em}
\text{AcDbDatabase} \ast \hspace{0.5em} \text{pDb}, \hspace{1em}
\text{double} \hspace{1em} \text{override_dpi} = 0.0, \hspace{1em}
\text{long} \hspace{1em} \text{override_windnumber} = -1 \hspace{1em});
\]

File

DmScaleFactor.h

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pDb</td>
<td>Input AcDbDatabase to compute scale factor.</td>
</tr>
<tr>
<td>override_dpi</td>
<td>Input value to override internal dpi calculation. Default value of 0.0 indicates to the function to get the screen dpi from the operating system.</td>
</tr>
<tr>
<td>override_windnumber</td>
<td>Input value to override selected viewport number for a layout. Default value of -1 indicates to the function to use the current paperspace layout viewport to compute the scale factor</td>
</tr>
</tbody>
</table>

Returns

Returns the current scale factor.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Namespace for display-management utility functions.

- **Enumerations**
  - **EDMStatus**
    - Enumerates the types of display-management active status.

- **Functions**
  - **AddProjectReactor**
    - Adds a display-management reactor to monitor project-level activities such as creation and deletion.
    - Used to commit uncommitted edits to objects the Element queried in from external data sources.
  - **CommitEdits**
    - Used to commit uncommitted edits to objects the Elements within the specified Map queried in from external data sources.
    - Used to discard uncommitted edits to objects the Element queried in from external data sources.
  - **DiscardEdits**
    - Used to discard uncommitted edits to objects the Elements within the specified Map queried in from external data sources.
    - Retrieves the ID of the display-management map manager object for a database.
  - **DMMapManagerId**
    - Retrieves the ID of the display-management map manager object
for a project.

DMStyleLibraryId
Retrieves the ID of display-management style library object for a database.

DMStyleLibraryId
Retrieves the ID of display-management style library object for a project.

DMStyleLibraryId
Retrieves the ID of display-management style library object for a project.

DMStyleLibraryId
Retrieves the ID of display-management style library object for a database.

DMStyleLibraryId
Retrieves the ID of display-management style library object for a project.

GetCurrentMapBaseElementInvisibleEntities
Retrieves the visibility status of the map base element and the IDs of entities made invisible when it was unchecked.

GetDMMapManager
Retrieves the display-management map manager object for the specified database.

GetDMMapManager
Retrieves the display-management map manager object for a specified AutoCAD Map project.

GetDMMapManager
Retrieves the display-management map manager object for the specified database.

GetDMMapManager
Retrieves the display-management map manager object for a specified AutoCAD Map project.

GetDMStyleLibrary
Retrieves the display-management style library object for a database.

GetDMStyleLibrary
Retrieves the display-management style library object for a project.

GetDMStyleLibrary
Retrieves the display-management style library object for a database.

GetDMStyleLibrary
Retrieves the display-management style library object for a project.

GetEntityIdForFeatureOnLayer
This function will search the current Map in the provided database to see if there is a FeatureEntity for a given GWS FeatureId currently in existence.

GetGWS
This function opens the GWS that the map refers to.

GetGWS
This function will search the current Map in the provided database to see if there is a VectorElement for the given Layer name.
HasDMData

Determines whether a project has display-management data. Used to determine if an Element has uncommitted edits to objects it queried in from external data sources.

HasUncommittedEdits

Used to determine if a Map has uncommitted edits to objects queried in from external data sources by any of the Map's Elements.

HasUncommittedEdits

OnBeginWritingToDataStore

Notifies the display manager that an application is about to write a drawing database to an external data store.

OnEndWritingToDataStore

Notifies the display manager that an application is finished writing a drawing database to an external data store.

RemoveProjectReactor

Removes a display-management reactor from a project.

Status

Retrieves the display-management current active status.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
This function will search the current Map in the provided database to see if there is a FeatureEntity for a given GWS FeatureId currently in existence.

```cpp
Acad::ErrorStatus GetEntityIdForFeatureOnLayer(
    AcDbObjectId& entId,
    AcDbDatabase* pDb,
    const TCHAR* layerSourceName,
    int featureId
);
```

File

DmDisplayManagement.h

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>entId</td>
<td>Output the FeatureEntity id. May be kNull.</td>
</tr>
<tr>
<td>pDb</td>
<td>The database to search.</td>
</tr>
<tr>
<td>layerSourceName</td>
<td>The name of the GWS LayerSource in which the featureId resides.</td>
</tr>
<tr>
<td>featureId</td>
<td>The featureId of the GWS feature instance. This should be a GWS 'cache id'.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful, otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
This function opens the GWS that the map refers to.

```cpp
Acad::ErrorStatus GetGWS(
    IGWS** pGWS,
    AcDbDatabase* pDb
);
```

### Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pGWS</td>
<td>Output opened IGWS.</td>
</tr>
<tr>
<td>pDatabase</td>
<td>Input database.</td>
</tr>
</tbody>
</table>

### Returns

Returns Acad::eOk if successful, Acad::eKeyNotFound if the dictionary entry with the file name does not exist otherwise, returns a different error code.

### Remarks

If the GWS does not exist yet it will create it and assign it to the Map. Once it has been created the name of the XML file is stored in the dictionary; this function retrieves the name from the dictionary to nopen the GWS.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
This function will search the current Map in the provided database to see if there is a VectorElement for the given Layer name.

```cpp
Acad::ErrorStatus GetVectorElementIdForFeatureLayer(
    AcDbObjectId& elementId,
    AcDbDatabase* pDb,
    const TCHAR* layerSourceName
);
```

File

DmDisplayManagement.h

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pDb</td>
<td>The database to search.</td>
</tr>
<tr>
<td>layerSourceName</td>
<td>The name of the GWS LayerSource in which the featureId resides.</td>
</tr>
<tr>
<td>entId</td>
<td>Output the VectorElement id. May be kNull.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful, otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
- Display Manager
- AcMapDMThematicUpdateFlags Enumeration
- Display Manager

This is record AcMapDMThematicUpdateFlags.

```cpp
enum AcMapDMThematicUpdateFlags {
    kUpdateWithoutText = 0x00,
    kUpdateTextOnly = 0x04
};
```

File

DmThematicStyle.h

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Enumerates defined column indexes for a thematic definition table.

```cpp
enum AcMapThematicTableColumnKey {
    kColkey_DataValue = 0,
    kColkey_LegendText = 1,
    kColkey_Color = 2,
    kColkey_Linestyle = 3,
    kColkey_Block = 4,
    kColkey_Text = 5,
    kColkey_Hatch = 6,
    kColkey_Linetype = 7,
    kColkey_Linewidth = 8,
    kColkey_Plotstyle = 9,
    kColkey_Annotation = 10
};
```

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Enumerates the insertion point for thematic table items.

```c
enum AcMapThematicTableItemInsertionPoint {
    kCenterIns,
    kCentroidIns,
    kLabelptIns,
    kX1Ins,
    kX2Ins,
    kX3Ins,
    kX4Ins
};
```

Parameters

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>kCenterIns</td>
<td>Center.</td>
</tr>
<tr>
<td>kCentroidIns</td>
<td>Centroid.</td>
</tr>
<tr>
<td>kLabelptIns</td>
<td>Label.</td>
</tr>
<tr>
<td>kX1Ins</td>
<td>X1.</td>
</tr>
<tr>
<td>kX2Ins</td>
<td>X2.</td>
</tr>
<tr>
<td>kX3Ins</td>
<td>X3.</td>
</tr>
<tr>
<td>kX4Ins</td>
<td>X4.</td>
</tr>
</tbody>
</table>

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Enumerates justification types for thematic table items.

```c
enum AcMapThematicTableItemJustificationType {
    kLeftJust,
    kCenterJust,
    kMiddleJust,
    kRightJust
};
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>kLeftJust</td>
<td>Left justification.</td>
</tr>
<tr>
<td>kCenterJust</td>
<td>Center justification.</td>
</tr>
<tr>
<td>kMiddleJust</td>
<td>Middle justification.</td>
</tr>
<tr>
<td>kRightJust</td>
<td>Right justification.</td>
</tr>
</tbody>
</table>

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
Display Manager
AcMapDMThematicBuildRangesErrorCode Enumeration
Display Manager

Error codes that are returned when a build-ranges operation fails in thematic mapping. See also AcMapDMThematicStyle::BuildRangeTables().

```cpp
enum AcMapDMThematicBuildRangesErrorCode {
    keUnknownError = 0,
    keOk = 1,
    keCouldNotOpenThemeTable,
    keMapSessionIsNull,
    keGetProjectFailed,
    keGetRangeLibFailed,
    keAddRangeTabFailed,
    keTableGetHatchIsNull,
    keTableGetDataIsNull,
    keTableGetColorIsNull,
    keTableGetLinestyleIsNull,
    keTableGetBlockIsNull,
    keTableGetTextIsNull,
    keTableGetLineweightIsNull,
    keTableGetLinetypeIsNull,
    keTableGetPlotstyleIsNull,
    keTableGetAnnotationIsNull
};
```

File
DmThematicStyle.h

Parameters Description
keUnknownError Unknown error.
keOk Completed successfully.
keCouldNotOpenThemeTable Cannot open the theme table.
keMapSessionIsNull Null AutoCAD Map session.
keGetProjectFailed Cannot get AutoCAD Map project.
keGetRangeLibFailed Cannot get range library.
keAddRangeTabFailed Cannot add range table.
keTableGetHatchIsNull Null hatch.
keTableGetDataIsNull Null data.
<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>keTableGetColorIsNull</code></td>
<td>Null color.</td>
</tr>
<tr>
<td><code>keTableGetLinestyleIsNull</code></td>
<td>Null linestyle.</td>
</tr>
<tr>
<td><code>keTableGetBlockIsNull</code></td>
<td>Null block.</td>
</tr>
<tr>
<td><code>keTableGetTextIsNull</code></td>
<td>Null text.</td>
</tr>
<tr>
<td><code>keTableGetLineweightIsNull</code></td>
<td>Null linewidth.</td>
</tr>
<tr>
<td><code>keTableGetLinetypeIsNull</code></td>
<td>Null linetype.</td>
</tr>
<tr>
<td><code>keTableGetPlotstyleIsNull</code></td>
<td>Null plotstyle.</td>
</tr>
<tr>
<td><code>keTableGetAnnotationIsNull</code></td>
<td>Null annotation.</td>
</tr>
</tbody>
</table>

Created with a commercial version of [Doc-O-Matic](#). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
Display Manager
Classes
Display Manager

Classes

AcMapDMAAllDrawOrderItemsIterator
An iterator over the draw order (AcMapDMElemnet). This iterator will step on every element, including those in a proxied state.

AcMapDMAAllItemsIterator
An iterator over a collection of display-management items. This iterator will step on every item, including those which may be in a proxy state. (AcMapDMDmsterRefefer)

AcMapDMAAllStyleReferencesIterator
An iterator over a collection of style references (AcMapDMStyleReference). This iterator may return a StyleId() that points to an AcDbProxyObject.

AcMapDMAAttachedDwgsQueryDataSourceDescriptor
Describes a display-management ADE query and provides functions that define queries and manage drawing lists.

AcMapDMAAttachedDwgsQueryElement
Represents a display-management ADE query element in an attached drawing.

AcMapDMCurrentDwgQueryElement
Represents a query element in the current drawing.

AcMapDMDataSourceDescriptor
Represents a query definition's data-source descriptor.

AcMapDMDefaultStyle
Represents the default stylization.

AcMapDMElement
An abstraction of an AutoCAD Map element that manipulates associated styles.

AcMapDMEntityStyle
Encapsulates properties that can be applied to entities.

AcMapDMFeatureDataSourceDescriptor
Represents a feature-class data source.

AcMapDMFeatureElement
Represents a feature-class element.

AcMapDMLayerDataSourceDescriptor
Represents a layer data source.

AcMapDMLayerElement
Represents a layer element.

AcMapDMGroup
Represents a group of display-management items (AcMapDMItem).

AcMapDMItem
Base class of all display-management items.

AcMapDMLayerDataSourceDescriptor
Represents a layer data source.

AcMapDMLayerElement
Represents a layer element.

Represents a legend object. A legend has the following characteristics: A legend is a table.
AcMapDMLegend
display-manager tree. The order of items in the legend matches the order of items in the display-manager tree. Each "space" (modelspace, layout1, layoutN, and so on) can have zero or one legend, which is built from the current Map and Scale when a legend object (AcMapDMLegend) is inserted or updated.

AcMapDMMap
Represents the Display Map, from which all other Display Manager objects can be obtained using an iterator.

AcMapDMMapIterator
An iterator over a collection of display-management map objects (AcMapDMMap).

AcMapDMMapManager
Manages display-management map objects (AcMapDMMap).

AcMapDMMapReactor
Base class used to notify an application of display-management events within a display-management map. Derive custom reactors from the AcMapDMMapReactor class.

To add a reactor: Derive your class from AcMapDMMapReactor and override the virtual functions of this reactor base class. Use AcMapDMMap::AddReactor to register an instance of the reactor. To unregister a reactor, use AcMapDMMap::RemoveReactor.

AcMapDMProjectReactor
Base class used to notify an application of display-management events within an AutoCAD Map project. Derive custom reactors from the AcMapDMProjectReactor class.

To add a reactor: Derive your class from AcMapDMProjectReactor and override the virtual functions of this reactor base class. Use AcMapDMDisplayManagement::AddProjectReactor to register an instance of the reactor. To unregister a reactor, use AcMapDMDisplayManagement::RemoveProjectReactor.

AcMapDMQueryDataSourceDescriptor
Describes a base data source for querying.

AcMapDMRasterDataSourceDescriptor
Defines a raster data source.
**AcMapDMRasterElement**

Represents a raster element.

**AcMapDMRasterStyle**

Encapsulates raster-image-related properties.

**AcMapDMSEAnnotationStyle**

Provides an annotation alteration style for stylization. Several properties can be represented as either a concrete type or as an ADE expression (in such cases, an API is provided for each representation). Setting a property by expression clears any value previously set by a concrete type. Likewise, setting any property by concrete type clears any value previously set in an expression. An API, `XxxIsExpression()` is provided to test which representation is currently in effect for each property. To represent the &quot;state&quot;, set the expression to `NULL` for that property. Using the wrong `GetXxx` function for the current representation of... *more*

**AcMapDMSEHatchStyle**

Provides a hatch alteration style for stylization. Several properties can be represented as either a concrete type or as an ADE expression (in such cases, an API is provided for each representation). Setting any property by expression Clears any value previously set in an expression. An API is provided to test which representation is currently in effect for each property. To represent the &quot;state&quot;, set the expression to `NULL` for that property. Using the wrong `GetXxx` function for the current representation of... *more*

**AcMapDMSETextStyle**

Provides a text alteration style for stylization. Several properties can be represented as either a concrete type or as an ADE expression (in such cases, an API is provided for each representation). Setting any property by expression clears any value previously set in an expression. An API is provided to test which representation is currently in effect for each property. To represent the &quot;state&quot;, set the expression to `NULL` for that property. Using the wrong `GetXxx` function for the current representation of... *more*

**AcMapDMStyle**

Base class for display-management styles.

**AcMapDMStyleCategory**

Represents a category of styles.
AcMapDMStyleLibrary
Represents a library of categories of styles.

AcMapDMStyleReference
Represents a reference to an existing style.

AcMapDMStylizationEntityStyle
Represents a style of a stylization entity.

AcMapDMThematicStyle
Represents a thematic stylization.

AcMapDMThematicTable
Defines a display-management thematic table and its items.

AcMapDMThematicTableItemAnnotation
Represents an annotation cell of a thematic definition table.

AcMapDMThematicTableItemBlock
Represents a block cell of a thematic definition table.

AcMapDMThematicTableItemColor
Represents a color cell of a thematic definition table.

AcMapDMThematicTableItemDataValue
Represents a text/value-pair cell of a thematic definition table.

AcMapDMThematicTableItemHatch
Represents a hatch cell of a thematic definition table.

AcMapDMThematicTableItemLegendText
Represents a legend text cell of a thematic definition table.

AcMapDMThematicTableItemLinestyle
Represents a linestyle cell of a thematic definition table.

AcMapDMThematicTableItemLinetype
Represents a linetype cell of a thematic definition table.

AcMapDMThematicTableItemLineweight
Class representing lineweight cell of thematic definition table.

AcMapDMThematicTableItemPlotstyle
Represents a plotstyle cell of a thematic definition table.

AcMapDMThematicTableItemText
Represents a text cell of a thematic definition table.

AcMapDMThematicTableRow
Utility class that defines a table row and allows easier access to its column items.

AcMapDMTopoElement
Defines a query for all topological elements in the current drawing.

AcMapDMTopoElementDataSourceDescriptor
Defines a topology data-source.

AcMapDMTopoQueryDataSourceDescriptor
Defines a query element of the source drawing.

AcMapDMTopoQueryElement

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
An iterator over the draw order of elements (AcMapDMElement). This iterator will step on every element, including those which are currently in a proxied state.

```c
class AcMapDMAAllDrawOrderItemsIterator;
```

File

DmDisplayItem.h

Methods

- `~AcMapDMAAllDrawOrderItemsIterator` Destroys an instance of this class.
- `Done` Determines whether the iterator has reached the end of the collection.
- `Next` Advances to the next element in the iteration.
- `ObjectId` Retrieves the ID of the current object in the iteration.
- `Rewind` Moves to the first element in the iteration.
Links
AcMapDMAAllDrawOrderItemsIterator Class,
AcMapDMAAllDrawOrderItemsIterator Class
AcMapDMAAllDrawOrderItemsIterator:: Rewind Method
AcMapDMAAllDrawOrderItemsIterator Class |
AcMapDMAAllDrawOrderItemsIterator Class

Moves to the first element in the iteration.

**virtual void** Rewind() = 0;

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
An iterator over a collection of display-management items. This iterator will step on every item, including those which may be in a proxy state. (AcMapDMItem).

```cpp
class AcMapDMAAllItemsIterator;
```

### Methods

- **~AcMapDMAAllItemsIterator** Destroys an instance of this class.
- **Done** Determines whether the iterator has reached the end of the collection.
- **Next** Advances to the next element in the iteration.
- **ObjectId** Retrieves the ID of the current object in the iteration.
- **Rewind** Moves to the first element in the iteration.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Moves to the first element in the iteration.

```cpp
virtual void Rewind() = 0;
```

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
An iterator over a collection of style references (`AcMapDMStyleReference`) within one style set. The iterator may return a `StyleId()` that points to an `AcDbProxyObject`.

```c
class AcMapDMApAllStyleReferencesIterator;
```

File

`DmDisplayElement.h`

Methods

- **~AcMapDMApAllStyleReferencesIterator**  
  Destroys an instance of this class.
- **Done**  
  Determines whether the iterator has reached the end of the collection.
- **Next**  
  Advances to the next element in the iteration.
- **ObjectId**  
  Retrieves the ID of the current object in the iteration.
- **Rewind**  
  Moves to the first element in the iteration.
- **StyleId**  
  Retrieves the ID of the style that the current object points to.
- **ThresholdScale**  
  Retrieves the scale threshold of the current object.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at **support@toolsfactory.com**.
Describes a display-management ADE query and provides functions that define queries and manage drawing lists.

```java
class AcMapDMAttachedDwgQueryDataSourceDescriptor : public AcMapDMQueryDataSourceDescriptor
```

File

DmAdeQueryElement.h

Remarks

This class is an abstraction of an "envelope" to communicate selection definition between an element and the corresponding user-interface component.

Methods

- `~AcMapDMAttachedDwgQueryDataSourceDescriptor` Destroys an instance of this class.
- `AcMapDMAttachedDwgQueryDataSourceDescriptor` Constructs an instance of this class.
- `GetAcquisitionStatement` Retrieves the query's string representation.
- `GetDrawingList` Lists the drawings used as the scope of the query.
- `GetQuery` Retrieves the query definition.
- `SetAcquisitionStatement` Sets the query string.
- `SetDrawingList` Sets the list of drawings to use as the scope of the query.
- `setQuery` Sets the query definition.

Created with a commercial version of Doc-O-Matic. In order to make this
message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the query's string representation.

```cpp
virtual Acad::ErrorStatus GetAcquisitionStatement(
    ACHAR*& pszStatement
) const;
```

Parameters

- `pszStatement`: Output query string.

Returns

- Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the query string.

```cpp
virtual Acad::ErrorStatus SetAcquisitionStatement(
    const ACHAR* pszStatement);
```

**Parameters**

- **pszStatement**: Input query string.

**Description**

- Input query string.

**Returns**

- Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links

Classes
AcMapDMAttachedDwgsQueryElement Class

Classes

Represents a display-management ADE query element in an attached drawing.

class AcMapDMAttachedDwgsQueryElement : public AcMapDMCurrentDwgQueryElement

File

DmAdeQueryElement.h

Remarks

Use this class to specify query criteria and acquire objects that satisfy them.

Methods

- ~AcMapDMAttachedDwgsQueryElement
  Destroys an instance of this class.

- AcMapDMAttachedDwgsQueryElement
  Constructs an instance of this class.

  AcquireEntities
  Runs an ADE query against the drawing scope defined in the data-source descriptor.

  ClonesObjectsFromExternalSource
  Clones objects from the source drawings.

  DismissEntities
  Erases entities that are part of this query element.

  dwgInFields
  Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

  dwgOutFields
  Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is
called by the system as needed; it is unlikely that you will need to call it directly.

Let's this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

Let's this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

Evaluates two expressions against the objects that meet the acquisition criteria.

Evaluates expression(s) against the element selection and returns values for the selected entities.

Retrieves a query definition's data-source descriptor.

Retrieves the query definition's data-source descriptor.

Sets the query definition's data-source descriptor.

Makes the entities that are part of this element visible or invisible.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Evaluates two expressions against the objects that meet the acquisition criteria.

```cpp
virtual Acad::ErrorStatus EvaluateExpressionValues(
    AcArray<AcMapValue*>& arValues1,
    AcArray<AcMapValue*>& arValues2,
    const ACHAR* kpszExpression1,
    AcMap::EDataType kDataType1,
    const ACHAR* kpszExpression2,
    AcMap::EDataType kDataType2
);
```

**Parameters**

- `arValues1`: Output array of evaluated values.
- `arValues2`: Output array of evaluated values.
- `kpszExpression1`: Input expression to evaluate.
- `kDataType1`: Input return data type expected.
- `kpszExpression2`: Input expression to evaluate.
- `kDataType2`: Input return data type expected.

**Returns**

- Returns Acad::eOk if successful; otherwise, returns a different error code.

**Remarks**

In the case of query elements, objects are not cloned and the expressions are evaluated against the original objects. If the expressions are not applicable to an entity (the .AREA for a line, for example), the result for this entity has type AcMap::kUnknownType.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Evaluates expression(s) against the element selection and returns values for the selected entities.

```cpp
virtual Acad::ErrorStatus EvaluateExpressionValues(
    AcArray<AcMapValue*>& arValues,
    const ACHAR* kpszExpression,
    AcMap::EDataType kDataType
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>arValues</td>
</tr>
<tr>
<td>kpszExpression</td>
</tr>
<tr>
<td>kDataType</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

**Remarks**

If entities have not yet been acquired, this function acquires them.

Created with a commercial version of [Doc-O-Matic](https://www.docsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves a query definition's data-source descriptor.

```cpp
virtual Acad::ErrorStatus GetAcquisitionCriteria(
    ACHAR*& pszString
) const;
```

**Parameters**

<table>
<thead>
<tr>
<th>pszString</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
</tr>
<tr>
<td>Output pointer to the data-source descriptor, as a string representation.</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://toolsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links

Classes
AcMapDMCurrentDwgQueryElement Class

Classes

Represents a query element in the current drawing.

class AcMapDMCurrentDwgQueryElement : public AcMapDMElement;

File
DmProjectQueryElement.h

Methods

~AcMapDMCurrentDwgQueryElement
Destroys an instance of this class.

AcMapDMCurrentDwgQueryElement
Constructs an instance of this class.

AcquireEntities
Runs the query against the current drawing.

ClonesObjectsFromExternalSource
Clones objects from the source drawings.

dwgInFields
Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

dwgOutFields
Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that

dxfInFields
dxfoutFields

EntityWrite

Evaluates two expressions against the objects that meet the acquisition criteria.

EvaluateExpressionValues

Evaluates expression(s) against the element selection and returns values for the selected entities.

GetAcquisitionCriteria

Retrieves a query definition's data-source descriptor.

SetAcquisitionCriteria

Sets the query definition's data-source descriptor.

OnMapProjectInitialized

Invoked when an AutoCAD Map project is initialized.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMCurrentDwgQueryElement Class,
AcMapDMCurrentDwgQueryElement Class
AcMapDMCurrentDwgQueryElement:: EvaluateExpressionValues Method
AcMapDMCurrentDwgQueryElement Class |
AcMapDMCurrentDwgQueryElement Class

Evaluates two expressions against the objects that meet the acquisition criteria.

virtual Acad::ErrorStatus EvaluateExpressionValues(
    AcArray<AcMapValue*>& arValues1,
    AcArray<AcMapValue*>& arValues2,
    const ACHAR* kpszExpression1,
    AcMap::EDataType kDataType1,
    const ACHAR* kpszExpression2,
    AcMap::EDataType kDataType2
);

Parameters                  Description
arValues1               Output array of evaluated values.
arValues2               Output array of evaluated values.
kpszExpression1        Input expression to evaluate.
kDataType1             Input return data type expected.
kpszExpression2        Input expression to evaluate.
kDataType2             Input return data type expected.

Returns
Returns Acad::eOk if successful; otherwise, returns a different error code.

Remarks

In the case of query elements, objects are not cloned and the expressions are evaluated against the original objects. If the expressions are not applicable to an entity (the .AREA for a line, for example), the result for this entity has type AcMap::kUnknownType.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Evaluates expression(s) against the element selection and returns values for the selected entities.

```cpp
virtual Acad::ErrorStatus EvaluateExpressionValues(
    AcArray<AcMapValue*>& arValues,
    const ACHAR* kpszExpression,
    AcMap::EDataType kDataType
);
```

Parameters
- **arValues**: Output array of evaluated values.
- **kpszExpression**: Input expression to evaluate.
- **kDataType**: Input return data type expected.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Remarks

If entities have not yet been acquired, this function acquires them.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves a query definition's data-source descriptor.

```cpp
virtual Acad::ErrorStatus GetAcquisitionCriteria(
    ACHAR*& pszString
) const;
```

**Parameters**
- `pszString` 
  - Description: Output pointer to the data-source descriptor, as a string representation.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](http://docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links

Classes

AcMapDMDataSourceDescriptor Class

Classes

Represents a query definition's data-source descriptor.

class AcMapDMDataSourceDescriptor : public AcRxObject;

File

DmDisplayElement.h

Methods

GetAcquisitionStatement  Retrieves the string representation of the query.
SetAcquisitionStatement  Sets the query string.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the string representation of the query.

```cpp
virtual Acad::ErrorStatus GetAcquisitionStatement(
    ACHAR*& pszStatement
) const;
```

**Parameters**

<table>
<thead>
<tr>
<th>Description</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pszStatement</td>
<td>Output query string.</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the query string.

```cpp
virtual Acad::ErrorStatus SetAcquisitionStatement(const ACHAR* pszStatement);
```

**Parameters**
- *pszStatement*: Input query string.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
AcMapDMDefaultStyle Class

Represents the default stylization.

class AcMapDMDefaultStyle : public AcMapDMStyle;

File
DmDefaultStyle.h

Enumerations

- **DefaultStyleType** Enumerates the types of default styles.

Methods

- **~AcMapDMDefaultStyle** Destroys an instance of this class.
- **AcMapDMDefaultStyle** Constructs an instance of this class.
- **Apply** Applies the style to an entity.
- **ClearFade** Clears the fade. See also IsFadeCleared().
- **clone** Clones the object. See also clone() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.
- **copyFrom** Copies the contents of an object into the messaged object, if feasible. See also copyFrom() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.
- **dwgInFields** Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide.
- **dwgOutFields** This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.
- **Lets this object write its data. See also**
**dwgOutFields**

Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

Lets this object write its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide.

**dxfInFields**

This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

Lets this object read its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide.

**dxfOutFields**

Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

**GetFade**

Retrieves the fade property. See also SetFade().

**GetStyleType**

Retrieves the style type. This function is intended for use in future versions of AutoCAD Map, when styles other than DefaultStyleType::kEntityFade may be available. See also SetStyleType().

**IsFadeCleared**

Determines whether the fade is cleared. See also ClearFade().

**Preview**

Applies style traits to yield a preview of the specified entity.

**SetFade**

Sets the fade property. In this release of AutoCAD Map, this function simply turns the target entity to ACI color 9 (light gray). Future releases may permit other fade colors. See also GetFade().

**SetStyleType**

Sets the style type. This function is intended for use in future versions of AutoCAD Map, when styles other than DefaultStyleType::kEntityFade may be available. See also GetStyleType().

**UnApply**

Builds stylization information for later regeneration.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Applies style traits to yield a preview of the specified entity.

```cpp
virtual Acad::ErrorStatus Preview(
    AcDbObjectIdArray& createdEntities,
    AcDbObjectId targetEntId,
    void* internalUse
);
```

### Parameters

- `createdEntities`: Output array of created alteration entities.
- `targetEntId`: Input ID of preview entity.
- `internalUse`: Reserved. For internal use only.

### Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

### Remarks

For stylization entity styles such as text, hatching, and annotation, this function returns an array of created entities.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
An abstraction of an AutoCAD Map element that manipulates associated styles.

```cpp
class AcMapDMElement : public AcMapDMItem;
```

Remarks

A concrete element provides its own implementation for object selection.

- **Enumerations**
  - `EPreviewEntityType` Enumerates the preview entity types.

- **Methods**
  - `~AcMapDMElement` Destroys an instance of this class.
  - `AcMapDMElement` Constructs an instance of this class.
  - `AcquireEntities` Runs the query and retrieves and stores the IDs of objects that meet the selection criteria.
  - `AddStyle` Adds a new style to this element.
  - `Audit` This function is called by AutoCAD when the AUDIT command is executed.
  - `ClonesObjectsFromExternalSource` Clones objects that are from the external source at AcquireEntities() time.
  - `DismissEntities` Clears the acquired entities that are stored in this element.
  - `DismissStylization` Clears the current stylization.
  - `GetAnAlternative,ConstantName` Gets an alternative, constant name
**DisplayName**

for this type of element in certain situations where the instance name may be unavailable.

Lets this object read its data. See also `dwgInFields()` in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

**dwgInFields**

**dwgOutFields**

Lets this object write its data. See also `dwgOutFields()` in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

**dxfInFields**

Lets this object read its data. See also `dxfInFields()` in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

**dxfOutFields**

**EnableStyle**

Enables the style reference's style.

**EvaluateExpressionValues**

Evaluates two expressions against the objects that meet the acquisition criteria.

**EvaluateExpressionValues**

Evaluates expression(s) against the element selection and returns values for the selected entities.

This function may be called by derived classes during their
<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ExcludeEntitiesFromStylization</td>
<td>AcquireEntities() implementation, if the Element wishes to explicitly exclude certain entities from being stylized under all circumstances.</td>
</tr>
<tr>
<td>FilterOutStylizationEntities</td>
<td>Filters out those entities from the selection that, during stylization, acquired stylization entities of another element.</td>
</tr>
<tr>
<td>GetAcquiredEntities</td>
<td>Retrieves the current selection (defined by calling AcquireEntities()).</td>
</tr>
<tr>
<td>GetAcquisitionCriteria</td>
<td>Retrieves a string representation of the acquisition criteria.</td>
</tr>
<tr>
<td>GetAcquisitionCriteria</td>
<td>Retrieves the acquisition criteria for this element.</td>
</tr>
<tr>
<td>GetAllStyleReferencesIterator</td>
<td>Retrieves an iterator over a collection of style references (for the corresponding style set) applied to this element.</td>
</tr>
<tr>
<td>GetCustomPreviewEntityName</td>
<td>Retrieves the custom preview entity name. See also SetCustomPreviewEntityName().</td>
</tr>
<tr>
<td>GetPreviewBlockId</td>
<td>Retrieves the AcDbObjectId of the block used for preview, if preview is of type kPreviewBlock. See also SetPreviewBlockId().</td>
</tr>
<tr>
<td>GetPreviewEntityType</td>
<td>Retrieves the preview entity type of this element. See also SetPreviewEntityType().</td>
</tr>
<tr>
<td>GetTitle</td>
<td>Retrieves the name of this element. See also setTitle().</td>
</tr>
<tr>
<td>HasStyleReference</td>
<td>Determines whether this element has a specified style reference.</td>
</tr>
<tr>
<td>IsStyleApplied</td>
<td>Determines whether this element has a style associated with it at a specified scale.</td>
</tr>
</tbody>
</table>
IsStylized
Determines whether this element is stylized.

IsTopElement
Determines if this element is the top element in the map.

IsUniqueStyleSet
Determines whether the style set on this element is unique. See also MakeStyleSetUnique().

IsVisible
Determines whether this element is visible at the specified scale. See also SetVisible().

MakeStyleSetUnique
Makes unique the style set associated with this element. See also IsUniqueStyleSet().

MoveStyle
Moves a style within an element.

OnMapProjectInitialized
Invoked when an AutoCAD Map project is initialized.

OnObjectAppended
Invoked when an object is appended to the database.

OrderEntitiesByDrawOrder
Order an array of objects based on their current draw order.

PlaceEntitiesInFrontOfDrawOrder
Place this array of objects at the top of draw order.

QueueObjectsForRegenerate
Puts all queried objects that are part of this element on the regeneration queue.

RemoveEntitiesFromStylizationExclusion
This function may be called by derived classes which used the excludeEntityFromStylization() method.

RemoveEntityFromAcquisition
Removes the indicated entityId from the Element's acquisition set.

RemoveStyle
Removes a style from this element.

RemoveStyle
Removes a style from this element.

RemoveStyle
Removes a style from this element.

Render
Stylize and Render layer contents to the specified renderer.
SetAcquiredEntities
Sets the current selection (defined by calling AcquireEntities()).

SetAcquisitionCriteria
Sets the acquisition criteria for this element.

SetCustomPreviewEntityName
Sets the name for kPreviewBlock and kPreviewCustom preview entity types. See also GetCustomPreviewEntityName().

SetPreviewBlockId
Sets the AcDbObjectId of the block for kPreviewBlock preview entity type. See also GetPreviewBlockId().

SetPreviewEntityType
Sets the preview entity type. See also GetPreviewEntityType().

SetTitle
Sets the name of this element. See also GetTitle().

SetVisible
Sets the element visibility at the specified scale. See also IsVisible().

subErase
Invoked from within erase() before the erase actually occurs. See also subErase() in the AutoCAD ObjectARX Developer's Guide.

UpdateStylization
Stylizes the current or updated selection at the current scale.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMElement Class, AcMapDMElement Class
AcMapDMElement:: AddStyle Method
AcMapDMElement Class | AcMapDMElement Class

Adds a new style to this element.

```cpp
virtual Acad::ErrorStatus AddStyle(
    AcDbObjectId& styleRefId,
    AcMapDMStyle* pStyle,
    AcMapDMAllStyleReferencesIterator& Position
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>styleRefId</td>
<td>Output ID of the added style reference.</td>
</tr>
<tr>
<td>pStyle</td>
<td>Input pointer to a AcMapDMStyle object.</td>
</tr>
<tr>
<td>Position</td>
<td>Input position at which to add the style. See also AcMapDMAllStyleReferencesIterator</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Remarks

The style object and the owning AcMapDMMap object must be closed for this function to succeed.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMElement Class, AcMapDMElement Class
AcMapDMElement:: AddStyle Method
AcMapDMElement Class | AcMapDMElement Class

Adds a new style to this element.

```cpp
virtual Acad::ErrorStatus AddStyle(
    AcDbObjectId& styleRefId,
    const AcDbObjectId& styleId,
    AcMapDMAAllStyleReferencesIterator& Position
);
```

Parameters

- **styleRefId**
  - **Output ID of the added style reference.**

- **styleId**
  - **Input ID of the style.**

- **Position**
  - **Input position at which to add the style. See also AcMapDMAAllStyleReferencesIterator**

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Remarks

The style object and the owning AcMapDMMMap object must be closed for this function to succeed.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMElement Class, AcMapDMElement Class
AcMapDMElement:: DisplayName Method
AcMapDMElement Class | AcMapDMElement Class

Gets an alternative, constant name for this type of element in certain situations where the instance name may be unavailable.

```
virtual const ACHAR* DisplayName() const;
```

Returns

Returns a const ACHAR* alternative name for certain display purposes.

Remarks

Derived classes should return a constant, generally descriptive name.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Evaluates two expressions against the objects that meet the acquisition criteria.

```cpp
virtual Acad::ErrorStatus EvaluateExpressionValues(
    AcArray<AcMapValue*>& arValues1,
    AcArray<AcMapValue*>& arValues2,
    const ACHAR* kpszExpression1,
    AcMap::EDataType kDataType1,
    const ACHAR* kpszExpression2,
    AcMap::EDataType kDataType2
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Description</th>
<th>Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Output array of evaluated values.</td>
<td>arValues1</td>
</tr>
<tr>
<td>Output array of evaluated values.</td>
<td>arValues2</td>
</tr>
<tr>
<td>Input expression to evaluate.</td>
<td>kpszExpression1</td>
</tr>
<tr>
<td>Input return data type expected.</td>
<td>kDataType1</td>
</tr>
<tr>
<td>Input expression to evaluate.</td>
<td>kpszExpression2</td>
</tr>
<tr>
<td>Input return data type expected.</td>
<td>kDataType2</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

**Remarks**

In the case of query elements, objects are not cloned and the expressions are evaluated against the original objects. If the expressions are not applicable to an entity (the .AREA for a line, for example), the result for this entity has type AcMap::kUnknownType.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Evaluates expression(s) against the element selection and returns values for the selected entities.

```cpp
virtual Acad::ErrorStatus EvaluateExpressionValues(
    AcArray<AcMapValue*>& arValues,
    const ACHAR* kpszExpression,
    AcMap::EDataType kDataType
);
```

Parameters

- **arValues**: Output array of evaluated values.
- **kpszExpression**: Input expression to evaluate.
- **kDataType**: Input return data type expected.

Returns

Returns `Acad::eOk` if successful; otherwise, returns a different error code.

Remarks

If entities have not yet been acquired, this function acquires them.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
This function may be called by derived classes during their AcquireEntities() implementation, if the Element wishes to explicitly exclude certain entities from being stylized under all circumstances.

\[
\text{Acad::ErrorStatus ExcludeEntitiesFromStylization(}
\text{\textbf{const AcDbObjectIdArray& ids}}
\text{)};
\]

**Parameters**

- **ids**: This is the AcDbObjectIdArray of the entities that the Element wishes to exclude from stylization.

**Returns**

Returns Acad::eOk if successful, otherwise, returns a different error code.

**Remarks**

This is rare, but may be appropriate if the Element (for example) creates temporary entities meant for a purpose other than stylization.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
AcMapDMElement Class, AcMapDMElement Class

AcMapDMElement:: GetAcquisitionCriteria Method

AcMapDMElement Class | AcMapDMElement Class

Retrieves a string representation of the acquisition criteria.

**virtual** Acad::ErrorStatus GetAcquisitionCriteria(
    ACHAR*& pszString
) const;

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pszString</td>
<td>Output string representation of the query.</td>
</tr>
</tbody>
</table>

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the name of this element. See also setTitle().

```cpp
Acad::ErrorStatus getTitle(
    const ACHAR*& kpszTitle
) const;
```

**Parameters**
- `kpszTitle`: Output element name.

**Returns**
Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determines if this element is the top element in the map.

```cpp
bool IsTopElement() const;
```

Returns true if the element is the top element; false on any error or if the element is not the top element.

Remarks

The element must exist in the map.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Moves a style within an element.

**Acad::ErrorStatus** MoveStyle(

```cpp
    const AcDbObjectId& styleRefId,
    AcMapDMAAllStyleReferencesIterator& newPosition

);```

**Parameters**

- **styleRefId**
  - Input ID of the style reference.
- **newPosition**
  - Input new position to move to. See also `AcMapDMAAllStyleReferencesIterator`

**Returns**

Returns Acad::eOk if successful. Returns Acad::eInvalidOwnerObject if the specified style reference belongs to another element. Returns Acad::eInvalidIndex if newPosition is invalid. Returns Acad::eUnrecoverableErrors if a fatal error occurred.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links

AcMapDMElement Class, AcMapDMElement Class
AcMapDMElement:: RemoveEntitiesFromStylizationExclusion Method
AcMapDMElement Class | AcMapDMElement Class

This function may be called by derived classes which used the excludeEntityFromStylization() method.

Acad::ErrorStatus RemoveEntitiesFromStylizationExclusion(const AcDbObjectIdArray& ids);

Parameters

ids

This is the AcDbObjectIdArray of the entities that the Element no longer wishes to exclude from stylization.

Returns

Returns Acad::eOk if successful, otherwise, returns a different error code.

Remarks

When the Element is done excluding the Entity from stylization, it can call this method to alert the system that this particular entity no longer requires specialized handling.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMElement Class, AcMapDMElement Class
AcMapDMElement:: RemoveEntityFromAcquisition Method
AcMapDMElement Class | AcMapDMElement Class

Removes the indicated entityId from the Element's acquisition set.

virtual Acad::ErrorStatus RemoveEntityFromAcquisition(
    AcDbObjectId entityId
);

Parameters          Description
entityId            Input AcDbObjectId of entity to remove from
                     Acquisition Set.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Removes a style from this element.

Acad::ErrorStatus RemoveStyle(
    AcMapDMAAllStyleReferencesIterator & Position
);

Parameters
Position

Description
Input position of the style to remove. See also
AcMapDMAAllStyleReferencesIterator

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this
message disappear you need to register this software. If you have problems
registering this software please contact us at support@toolsfactory.com.
Stylize and Render layer contents to the specified renderer.

```cpp
virtual Acad::ErrorStatus Render(
    Renderer & renderer
);
```

**Parameters**
- `renderer`: The Renderer onto which the element will render all layer graphics.

**Returns**
- Returns Acad::eOk if successful; otherwise, returns a different error code.

**Remarks**

This is used in various places including creating a bitmap representing the layer.

Created with a commercial version of [Doc-O-Matic](#). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the name for kPreviewBlock and kPreviewCustom preview entity types. See also GetCustomPreviewEntityName().

```cpp
virtual Acad::ErrorStatus SetCustomPreviewEntityName(const ACHAR* pName);
```

**Parameters**
- **pName**: Input name of the block.

**Returns**
- Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the name of this element. See also GetTitle().

```cpp
Acad::ErrorStatus SetTitle(
    const ACHAR* kpszTitle
);
```

**Parameters**

- **Description**
  - `kpszTitle`: Input element name.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Encapsulates properties that can be applied to entities.

class AcMapDMEntityStyle : public AcMapDMStyle;

File

DmEntityStyle.h

Methods

~AcMapDMEntityStyle Destroys an instance of this class.

AcMapDMEntityStyle Constructs an instance of this class.

Apply Applies the style to an entity.

ClearColor Clears the color. See also IsColorCleared().

ClearLinetype Clears the linetype. See also IsLinetypeCleared().

ClearLinetypeScale Clears the linetype scale. See also IsLinetypeScaleCleared().

ClearLineWeight Clears the lineweight. See also IsLineWeightCleared().

ClearPlotstyleName Clears the plotstyle. See also IsPlotstyleNameCleared().

clone Clones the object. See also clone() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

copyFrom Copies the contents of an object into the messaged object, if feasible. See also copyFrom() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

Dismiss Clears the stylization of a topology.

Dismiss Clears the stylization on an entity.
Let this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

Let this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

Let this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

Let this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

Retrieves the color. See also SetColor().

Retrieves the linetype id. See also SetLinetype().

Retrieves the linetype name. See also SetLinetype().

Retrieves the linetype scale. See also SetLinetypeScale().

Retrieves the lineweight. See also SetLineWeight().

Retrieves the plotstyle. See also SetPlotstyleName().

Determines whether the color is cleared. See also ClearColor().

Determines whether the linetype is cleared. See also ClearLinetype().

Determines whether the linetype scale is cleared. See also ClearLinetypeScale().

Determines whether the lineweight is cleared. See also ClearLineWeight().

Determines whether the plotstyle is cleared. See also ClearPlotstyleName().

Applies style traits to yield a preview of the specified entity.
<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SetColor</td>
<td>Sets the color. See also GetColor().</td>
</tr>
<tr>
<td>SetLinetype</td>
<td>Sets the linetype. See also GetLinetype().</td>
</tr>
<tr>
<td>SetLinetype</td>
<td>Sets the linetype. See also GetLinetype().</td>
</tr>
<tr>
<td>SetLinetypeScale</td>
<td>Sets the linetype scale. See also GetLinetypeScale().</td>
</tr>
<tr>
<td>SetLineWeight</td>
<td>Sets the linewidth. See also GetLineWeight().</td>
</tr>
<tr>
<td>SetPlotstyleName</td>
<td>Sets the plotstyle. See also GetPlotstyleName().</td>
</tr>
<tr>
<td>UnApply</td>
<td>Removes the style from a specified entity.</td>
</tr>
<tr>
<td>Update</td>
<td>Retrieves and stylizes a topology.</td>
</tr>
<tr>
<td>Update</td>
<td>Retrieves and stylizes an entity.</td>
</tr>
</tbody>
</table>

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Clears the stylization of a topology.

```cpp
template virtutal Acad::ErrorStatus Dismiss(
    void* & pCookie,
    const ACHAR* pTopoName,
    Adesk::IntDbId lTopoElemId,
    Adesk::UInt32 flag = 0
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pCookie</td>
<td>Input memory value used by styles to transfer cached data across time from one stylization method to another. Cookie memory is considered to be opaque for existing styles, and should not be modified by the caller.</td>
</tr>
<tr>
<td>pTopoName</td>
<td>Input topology name.</td>
</tr>
<tr>
<td>ITopoElemId</td>
<td>Input topology ID.</td>
</tr>
<tr>
<td>flag</td>
<td>Reserved. For internal use only.</td>
</tr>
</tbody>
</table>

**Returns**

Returns `Acad::eOk` if successful; otherwise, returns a different error code.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at **support@toolsfactory.com**.
Retrieves the linetype name. See also SetLinetype().

```cpp
Acad::ErrorStatus GetLinetype(
    const ACHAR*& pLinetypeName
) const;
```

Parameters

- `pLinetypeName`: Output linetype name.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the plotstyle. See also SetPlotstyleName().

```cpp
Acad::ErrorStatus GetPlotstyleName(
    const ACHAR*& pPlotStyleName
) const;
```

**Parameters**
- `pPlotStyleName` Output plotstyle name.

**Returns**
- Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](#). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Applies style traits to yield a preview of the specified entity.

```cpp
virtual Acad::ErrorStatus Preview(
  AcDbObjectIdArray& createdEntities,
  AcDbObjectId targetEntId,
  void* internalUse
);
```

**Parameters**

- `createdEntities` Output array of created alteration entities.
- `targetEntId` Input ID of the preview entity.
- `internalUse` Reserved. For internal use only.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

**Remarks**

For stylization entity styles such as text, hatch, and annotation, this function returns an array of created entities.

Created with a commercial version of [Doc-O-Matic](https://www.docsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the linetype. See also GetLinetype().

\[
\text{Acad::ErrorStatus SetLinetype(}
\text{    const ACHAR* pLinetypeName}
\text{)}
\]

Parameters  Description
pLinetypeName  Input linetype name.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the plotstyle. See also GetPlotstyleName().

```
Acad::ErrorStatus SetPlotstyleName(
    const ACHAR* pPlotStyleName
);
```

**Parameters**

- `pPlotStyleName`: Input plotstyle name.

**Description**

Input plotstyle name.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Retrieves and stylizes a topology.

```cpp
virtual Acad::ErrorStatus Update(
    void*& pCookie,
    const ACHAR* pTopoName,
    Adesk::IntDbId lTopoId,
    Adesk::UInt32 flag = 0
);
```

### Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pCookie</td>
<td>Output memory value used by styles to transfer cached data across time from one stylization method to another. Cookie memory is considered to be opaque for existing styles, and should not be modified by the caller.</td>
</tr>
<tr>
<td>pTopoName</td>
<td>Input topology name.</td>
</tr>
<tr>
<td>lTopoId</td>
<td>Input topology ID.</td>
</tr>
<tr>
<td>flag</td>
<td>Reserved. For internal use only.</td>
</tr>
</tbody>
</table>

### Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Represents a feature-class data source.

```cpp
class AcMapDMFeatureDataSourceDescriptor : public AcMapDMDataSourceDescriptor
```

File

DmFeatureElement.h

Methods

- `~AcMapDMFeatureDataSourceDescriptor` Destroys an instance of this class.
- `AcMapDMFeatureDataSourceDescriptor` Constructs an instance of this class.
- `GetAcquisitionStatement` Retrieves the query's string representation.
- `SetAcquisitionStatement` Sets the query string.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the query's string representation.

```cpp
virtual Acad::ErrorStatus GetAcquisitionStatement(
    ACHAR*& pszStatement
) const;
```

**Parameters**

<table>
<thead>
<tr>
<th>pszStatement</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Output query string.</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](#). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the query string.

```cpp
virtual Acad::ErrorStatus SetAcquisitionStatement(
    const ACHAR* pszStatement);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pszStatement</td>
<td>Input query string.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links

Classes
AcMapDMFeatureElement Class

**Classes**

Represents a feature-class element.

```c++
class AcMapDMFeatureElement : public AcMapDMCurrentDwgQueryElement;
```

**File**

DmFeatureElement.h

**Methods**

- `~AcMapDMFeatureElement` ➧ Destroys an instance of this class.
- `AcMapDMFeatureElement` ➧ Constructs an instance of this class.
- `AcquireEntities` ➧ Runs an ADE query against the drawing scope defined in the data-source descriptor.
- `ClonesObjectsFromExternalSource` ➧ Clones objects from the source drawings.
- `dwgInFields` ➧ Lets this object read its data. See also `dwgInFields()` in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.
- `dwgOutFields` ➧ Lets this object write its data. See also `dwgOutFields()` in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.
- `dxfInFields` ➧ Lets this object read its data. See also `dxfInFields()` in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.
dxfoutFields

Lets this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

EvaluateExpressionValues

Evaluates two expressions against the objects that meet the acquisition criteria.

EvaluateExpressionValues

Evaluates expression(s) against the element selection and returns values for the selected entities.

GetAcquisitionCriteria

Retrieves a query definition's data-source descriptor.

GetAcquisitionCriteria

Retrieves the query definition's data-source descriptor.

OnMapProjectInitialized

Invoked when an AutoCAD Map project is initialized.

SetAcquisitionCriteria

Sets the query definition's data-source descriptor.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
virtual Acad::ErrorStatus EvaluateExpressionValues(
    AcArray<AcMapValue*>& arValues1,
    AcArray<AcMapValue*>& arValues2,
    const ACHAR* kpszExpression1,
    AcMap::EDataType kDataType1,
    const ACHAR* kpszExpression2,
    AcMap::EDataType kDataType2
);

Parameters

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>arValues1</td>
</tr>
<tr>
<td>arValues2</td>
</tr>
<tr>
<td>kpszExpression1</td>
</tr>
<tr>
<td>kDataType1</td>
</tr>
<tr>
<td>kpszExpression2</td>
</tr>
<tr>
<td>kDataType2</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Remarks

In the case of query elements, objects are not cloned and the expressions are evaluated against the original objects. If the expressions are not applicable to an entity (the .AREA for a line, for example), the result for this entity has type AcMap::kUnknownType.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Evaluates expression(s) against the element selection and returns values for the selected entities.

```
virtual Acad::ErrorStatus EvaluateExpressionValues(
    AcArray<AcMapValue*>& arValues,
    const ACHAR* kpszExpression,
    AcMap::EDataType kDataType
);  
```

Parameters

- **arValues**: Output array of evaluated values.
- **kpszExpression**: Input expression to evaluate.
- **kDataType**: Input return data type expected.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Remarks

If entities have not yet been acquired, this function acquires them.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves a query definition's data-source descriptor.

```cpp
virtual Acad::ErrorStatus GetAcquisitionCriteria(
    ACHAR*& pszString
) const;
```

**Parameters**

<table>
<thead>
<tr>
<th>pszString</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
</tr>
<tr>
<td>Output pointer to a data-source descriptor, as a string representation.</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
Classes
AcMapDMGroup Class

Represents a group of display-management items (AcMapDMItem).

```
class AcMapDMGroup : public AcMapDMItem;
```

File
DmGroup.h

- **Methods**
  - `~AcMapDMGroup` - Destroys an instance of this class.
  - `AcMapDMGroup` - Constructs an instance of this class.
  - `AddItem` - Adds a new item to this group.
  - `AddItem` - Adds a new item to this group.
  - `audit` - This function is called by AutoCAD when the AUDIT command is executed.
  - `DismissStylization` - Clears the current stylization.
  - `dwgInFields` - Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.
  - `dwgOutFields` - Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.
  - `dxfInFields` - Lets this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.
  - `dxfOutFields` - Lets this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.
dxfoutFields

Directly.

Lets this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

erased

Lets this object listen to erase-notifications from items that it owns. See also erased() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

HasChildrenInSpecifiedVisibleState

Determines whether children are visible or invisible at a specified scale.

IsEmpty

Determines whether this group is empty.

IsVisible

Determines whether this group is visible at the specified scale. See also SetVisible() and HasChildrenInSpecifiedVisibleState().

MoveItem

Moves an item to a different position in this group.

NewAllItemsIterator

Retrieves a new iterator over the contents of this group.

OnMapCSChanged

Invoked when an AutoCAD Map changes project coordinate system.

OnMapProjectInitialized

Invoked when an AutoCAD Map project is initialized.

OnObjectAppended

Invoked when an object is appended to the database.

RemoveItem

Removes an item from this group.

RemoveItem

Removes an item from this group.

SelectElementsWithStyleApplied

Retrieves the IDs of the elements that have references to the specified style at the specified scale.

Sets the group visibility at the specified
SetVisible scale. See also IsVisible() and HasChildrenInSpecifiedVisibleState().

Invoked from within erase() before the erase actually occurs. See also subErase() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

subErase

UpdateStylization Stylizes the current or updated selection at the current scale.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Adds a new item to this group.

```
virtual Acad::ErrorStatus AddItem(
    AcDbObjectId& Id,
    AcMapDMItem* pItem,
    AcMapDMAllItemsIterator* pPosition = NULL
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Id</td>
<td>Output ID of the added item.</td>
</tr>
<tr>
<td>pItem</td>
<td>Input AcMapDMItem to add.</td>
</tr>
<tr>
<td>pPosition</td>
<td>Input position at which to add the item. The default value, NULL, adds the item to the end of the group. See also AcMapDMAllItemsIterator.</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.doc-omatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMGroup Class, AcMapDMGroup Class
AcMapDMGroup:: AddItem Method
AcMapDMGroup Class | AcMapDMGroup Class

Adds a new item to this group.

virtual Acad::ErrorStatus AddItem(
    const AcDbObjectId& Id,
    AcMapDMAAllItemsIterator* pPosition = NULL
);  
Parameters   Description
Id           Input ID of the item to add.
              Input position at which to add the item. The default
pPosition    value, NULL, adds the item to the end of the group.
              See also AcMapDMAAllItemsIterator.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this
message disappear you need to register this software. If you have problems
registering this software please contact us at support@toolsfactory.com.
Moves an item to a different position in this group.

```cpp
virtual Acad::ErrorStatus MoveItem(
    const AcDbObjectId& Id,
    AcMapDMAAllItemsIterator& Position
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Id</td>
<td>Input ID of the item to move.</td>
</tr>
<tr>
<td>Position</td>
<td>Input position to move the item to. See also</td>
</tr>
<tr>
<td></td>
<td>AcMapDMAAllItemsIterator.</td>
</tr>
</tbody>
</table>

Returns Acad::eOk if successful. Returns Acad::eInvalidIndex if the iterator position is invalid.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Invoked when an AutoCAD Map changes project coordinate system.

```cpp
virtual void OnMapCSChanged(
    const ACHAR * oldCS,
    const ACHAR * newCS
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>oldCS</td>
<td>Old CS code</td>
</tr>
<tr>
<td>newCS</td>
<td>New CS code</td>
</tr>
</tbody>
</table>

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Removes an item from this group.

```cpp
virtual Acad::ErrorStatus RemoveItem(
    Acad::MapDMAllItemsIterator& Position
);
```

**Parameters**
- **Position**
  - Input position of the item to remove. See also `AcMapDMAllItemsIterator`.

**Returns**
- Returns Acad::eOk if successful. Returns Acad::eInvalidIndex if the iterator position is invalid.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Base class of all display-management items.

```cpp
class AcMapDMItem : public AcDbObject;
```

File

DmDisplayItem.h

- **Enumerations**
  - `ELegendDetailLevel`:
    Enumerates the levels of legend detail.

- **Methods**
  - `~AcMapDMItem`:
    Destroys an instance of this class.
  - `AcMapDMItem`:
    Constructs an instance of this class.
  - `audit`:
    This function is called by AutoCAD when the AUDIT command is executed.
  - `DismissStylization`:
    Clears the current stylization.
  - `dwgInFields`:
    Lets this object read its data. See also `dwgInFields()` in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.
  - `dwgOutFields`:
    Lets this object write its data. See also `dwgOutFields()` in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.
  - `dxfInFields`:
    Lets this object read its data. See also `dxfInFields()` in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.
  - `dxfOutFields`:
    Lets this object write its data. See also `dxfOutFields()` in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed;
<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetName</td>
<td>Retrieves the name of this item. See also SetName().</td>
</tr>
<tr>
<td>Implementation</td>
<td>Returns the implementation object.</td>
</tr>
<tr>
<td>IsVisible</td>
<td>Determines whether this item is visible at the specified scale. See also SetVisible().</td>
</tr>
<tr>
<td>LegendDetailLevel</td>
<td>Retrieves the level of legend detail of this item. See also SetLegendDetailLevel().</td>
</tr>
<tr>
<td>MapId</td>
<td>Retrieves the map ID.</td>
</tr>
<tr>
<td>MapProject</td>
<td>Retrieves the AutoCAD Map project.</td>
</tr>
<tr>
<td>OnMapCSChanged</td>
<td>Invoked when an AutoCAD Map changes project coordinate system.</td>
</tr>
<tr>
<td>OnMapProjectInitialized</td>
<td>Invoked when an AutoCAD Map project is initialized.</td>
</tr>
<tr>
<td>OnObjectAppended</td>
<td>Invoked when an object is appended to the database.</td>
</tr>
<tr>
<td>SetLegendDetailLevel</td>
<td>Sets the level of legend detail of this item. See also LegendDetailLevel().</td>
</tr>
<tr>
<td>SetName</td>
<td>Sets the name of this item. See also GetName().</td>
</tr>
<tr>
<td>SetVisible</td>
<td>Sets the item visibility at the specified scale. See also IsVisible().</td>
</tr>
<tr>
<td>subClose</td>
<td>Invoked from within close() before the close actually occurs. The default implementation of this function returns Acad::eOk. See also subClose() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.</td>
</tr>
<tr>
<td>wblockClone</td>
<td>Grants control of deep clone operations to the object. In the default implementation, the object is cloned and appended to the owner object pOwnerObject. See also wblockClone() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.</td>
</tr>
</tbody>
</table>
Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Invoked when an AutoCAD Map changes project coordinate system.

```cpp
virtual void OnMapCSChanged(
    const ACHAR * oldCS,
    const ACHAR * newCS
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>oldCS</td>
<td>Old CS code</td>
</tr>
<tr>
<td>newCS</td>
<td>New CS code</td>
</tr>
</tbody>
</table>

Created with a commercial version of [Doc-O-Matic](https://www.doc-o-matic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the name of this item. See also GetName().

```cpp
virtual Acad::ErrorStatus SetName(
    const ACHAR* pszName
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pszName</td>
<td>Input item name.</td>
</tr>
</tbody>
</table>

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Represents a layer data source.

```cpp
class AcMapDMLayerDataSourceDescriptor : public AcMapDMDataSourcesDes
```

File

DmLayerElement.h

- Methods
  - `~AcMapDMLayerDataSourceDescriptor`: Destroys an instance of this class.
  - `AcMapDMLayerDataSourceDescriptor`: Constructs an instance of this class.
  - `GetAcquisitionStatement`: Retrieves the query's string representation.
  - `SetAcquisitionStatement`: Sets the query string.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the query's string representation.

```cpp
virtual Acad::ErrorStatus GetAcquisitionStatement(
    ACHAR*& pszStatement
) const;
```

Parameters

- **pszStatement**
  - Description: Output query string.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the query string.

```cpp
virtual Acad::ErrorStatus SetAcquisitionStatement(
    const ACHAR* pszStatement
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pszStatement</td>
<td>Input query string.</td>
</tr>
</tbody>
</table>

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links

Classes

AcMapDMLayerElement Class

Classes

Represents a layer element.

```
class AcMapDMLayerElement : public AcMapDMCurrentDwgQueryElement;
```

File

DmLayerElement.h

Methods

- **~AcMapDMLayerElement**
  - Destroys an instance of this class.

- **AcMapDMLayerElement**
  - Constructs an instance of this class.

**AcquireEntities**

Clones objects from the source drawings.

**ClonesObjectsFromExternalSource**

Let this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

**dwgInFields**

Let this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

**dwgOutFields**

**dxfInFields**

Let this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.
dxfOutFields

Lets this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

EvaluateExpressionValues

Evaluates two expressions against the objects that meet the acquisition criteria.

EvaluateExpressionValues

Evaluates expression(s) against the element selection and returns values for the selected entities.

GetAcquisitionCriteria

Retrieves a query definition's data-source descriptor.

GetAcquisitionCriteria

Retrieves the query definition's data-source descriptor.

OnMapProjectInitialized

Invoked when an AutoCAD Map project is initialized.

SetAcquisitionCriteria

Sets the query definition's data-source descriptor.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Evaluates two expressions against the objects that meet the acquisition criteria.

```cpp
virtual Acad::ErrorStatus EvaluateExpressionValues(
   AcArray<AcMapValue*>& arValues1,
   AcArray<AcMapValue*>& arValues2,
   const ACHAR* kpszExpression1,
   AcMap::EDataType kDataType1,
   const ACHAR* kpszExpression2,
   AcMap::EDataType kDataType2
);
```

### Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>arValues1</td>
<td>Output array of evaluated values.</td>
</tr>
<tr>
<td>arValues2</td>
<td>Output array of evaluated values.</td>
</tr>
<tr>
<td>kpszExpression1</td>
<td>Input expression to evaluate.</td>
</tr>
<tr>
<td>kDataType1</td>
<td>Input return data type expected.</td>
</tr>
<tr>
<td>kpszExpression2</td>
<td>Input expression to evaluate.</td>
</tr>
<tr>
<td>kDataType2</td>
<td>Input return data type expected.</td>
</tr>
</tbody>
</table>

### Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

### Remarks

In the case of query elements, objects are not cloned and the expressions are evaluated against the original objects. If the expressions are not applicable to an entity (the .AREA for a line, for example), the result for this entity has type AcMap::kUnknownType.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Evaluates expression(s) against the element selection and returns values for the selected entities.

```cpp
virtual Acad::ErrorStatus EvaluateExpressionValues(
    AcArray<AcMapValue*>& arValues,
    const ACHAR* kpszExpression,
    AcMap::EDataType kDataType
);
```

Parameters

- `arValues` Output array of evaluated values.
- `kpszExpression` Input expression to evaluate.
- `kDataType` Input return data type expected.

Returns

Returns `Acad::eOk` if successful; otherwise, returns a different error code.

Remarks

If entities have not yet been acquired, this function acquires them.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves a query definition's data-source descriptor.

```cpp
virtual Acad::ErrorStatus GetAcquisitionCriteria(
    ACHAR*& pszString
) const;
```

### Parameters

<table>
<thead>
<tr>
<th>pszString</th>
</tr>
</thead>
</table>
| Output pointer to the data-source descriptor, as a string representation.

### Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Represents a legend object. A legend has the following characteristics: A legend is a table. A legend will display the active (checked) Themes, Elements, and Groups as in the display-manager tree. The order of items in the legend matches the order of items in the display-manager tree. Each "space" (modelspace, layout1, layoutN, and so on) can have zero or one legend, which is built from the current Map and Scale when a legend is inserted or updated. A legend object does not store information on which layout or map it is associated with.

```cpp
class AcMapDMLegend : public AcDbTable;
```

### File
DmLegend.h

### Methods
- `~AcMapDMLegend` Destroys an instance of this class.
- `AcMapDMLegend` Constructs an instance of this class.
- `DismissTableContents` Clears the contents of the legend.

- `dwgInFields` Lets this object read its data. See also `dwgInFields()` in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.
- `dwgOutFields` Lets this object write its data. See also `dwgOutFields()` in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

- `dxfInFields` Lets this object read its data. See also `dxfInFields()` in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.
- `dxfOutFields` Lets this object write its data. See also `dxfOutFields()` in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed;
it is unlikely that you will need to call it directly.

**Implementation** Returns the implementation object.

**UpdateTableContents** Clears the contents of the legend and updates it with current data from the display manager.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
Classes
AcMapDMMap Class
Classes

Represents the Display Manager map, the root object from which all other Display Manager objects can be obtained by using an iterator.

```cpp
class AcMapDMMap : public AcMapDMGroup;
```

File

DmMap.h

- Enumerations
  - `DOMode` This is record AcMapDMMap::DOMode.

- Methods
  - `~AcMapDMMap` Destroys an instance of this class.
  - `AcMapDMMap` Constructs an instance of this class.
  - `AddElement` Adds an element to this map.
  - `AddReactor` Adds a reactor to this map.
  - `AddScaleThreshold` Adds a scale threshold to this map - this function is not yet supported in AutoCAD Map.
  - `audit` This function is called by AutoCAD when the AUDIT command is executed.
  - `CopyScaleThreshold` Copies a scale threshold to this map - this function is not yet supported in AutoCAD Map.
  - `CreateLegend` Creates a new legend for the specified layout.
  - `deepCloneObject` Clones this object and all the objects that it refers to.
  - `deepCloneObjects` Clones this object and all the objects that it refers to.
  - `DeleteLegend` Deletes the contents of the legend of the specified layout.
<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>DismissStylization</strong></td>
<td>Clears the current stylization.</td>
</tr>
<tr>
<td><strong>dwgInFields</strong></td>
<td>Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide.</td>
</tr>
<tr>
<td><strong>dwgOutFields</strong></td>
<td>Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide.</td>
</tr>
<tr>
<td><strong>dxfInFields</strong></td>
<td>Lets this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide.</td>
</tr>
<tr>
<td><strong>dxfOutFields</strong></td>
<td>Lets this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide.</td>
</tr>
<tr>
<td><strong>erased</strong></td>
<td>Lets this object listen to erase-notifications from items that it owns. See also erased() in the AutoCAD ObjectARX Developer's Guide.</td>
</tr>
<tr>
<td><strong>Get3dGridPercent</strong></td>
<td>Retrieves the current 3d grid percent of this map exactly as last input to Set3dGridPercent().</td>
</tr>
<tr>
<td><strong>GetAllDrawOrderItemsIterator</strong></td>
<td>Retrieves the draw-order iterator. The first function call iterates elements in the order that they are arranged in the map tree. This iterator may return Items that are in a proxy state, unless the bSkipProxies argument is passed as true. See also IsDrawOrderDefined().</td>
</tr>
</tbody>
</table>
**GetCurrentScaleThreshold**
Retrieves the current scale threshold of this map, as calculated from the last input to SetCurrentScale().

**GetCurrentUserScale**
Retrieves the current scale of this map exactly as last input to SetCurrentScale().

**GetDOMode**
Gets the draw order mode currently in use for the map.

**GetExaggeration**
Retrieves the current exaggeration of this map exactly as last input to SetExaggeration().

**GetExtent**
Retrieves the current extent of this map exactly as last input to SetExtent().

**GetLegend**
Retrieves the legend of the specified layout.

**GetLegendId**
Retrieves the legend of the specified layout.

**GetLineTypeScale**
Retrieves the linetype scale applied to the stylization. The linetype scale, a positive value, is assigned to the AutoCAD system variable LTSCALE at update time. See also SetLineTypeScale().

**GetLinkedFileName**
Retrieves the linked output filename. The linked output file contains the stylized objects of this map. By default, the current scale is used in output. See also SetLinkedFileName().

**GetName**
Retrieves the name of this map. See also SetName().

**GetScaleThreshold**
Retrieves the scale threshold at a specified index.

**GetSun**
Get sun vector setting for current map.

**GetTableStyle**
Retrieves the table style. See also SetTableStyle().

**GetThresholdFor**
Retrieves an upper-bound map threshold scale for the specified scale.

**GetThumbnailDimensions**
Retrieves the thumbnail width and height factors. These width and height factors are multiplied by the text height in the legend to
<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>GetThumbnailFraming</code></td>
<td>Determines whether a box is drawn around the thumbnail graphic. See also <code>SetThumbnailFraming()</code></td>
</tr>
<tr>
<td><code>InfinityThreshold</code></td>
<td>Returns the value used as the Infinity threshold scale in the Map.</td>
</tr>
<tr>
<td><code>IsDrawOrderDefined</code></td>
<td>Determines whether the draw order is defined. Draw order is defined on demand when <code>GetDrawOrderIterator()</code> is first called. Draw-order functionality is available in Display Manager user interface in the list view.</td>
</tr>
<tr>
<td><code>IsInfinityThreshold</code></td>
<td>Reports whether the input value is equal to the value used as the Infinity threshold scale in the Map.</td>
</tr>
<tr>
<td><code>IsLinkToFileEnabled</code></td>
<td>Determines whether linked-file capability is enabled.</td>
</tr>
<tr>
<td><code>IsStylized</code></td>
<td>Determines whether this map is stylized.</td>
</tr>
<tr>
<td><code>IsVisible</code></td>
<td>Determines whether this map is visible at the specified scale. See also <code>SetVisible()</code></td>
</tr>
<tr>
<td><code>MapId</code></td>
<td>Retrieves the ID of this map.</td>
</tr>
<tr>
<td><code>ModifyScaleThreshold</code></td>
<td>Changes the scale threshold value of an existing scale threshold.</td>
</tr>
<tr>
<td><code>MoveElement</code></td>
<td>Changes the position of an element within this map.</td>
</tr>
<tr>
<td><code>NumScaleThresholds</code></td>
<td>Counts the number of scale thresholds defined in this map.</td>
</tr>
<tr>
<td><code>OnMapProjectInitialized</code></td>
<td>Invoked when an AutoCAD Map project is initialized.</td>
</tr>
<tr>
<td><code>OnObjectAppended</code></td>
<td>Invoked when an object is appended to the database.</td>
</tr>
<tr>
<td><code>OnStyleModified</code></td>
<td>Makes the system to refresh the contents of map when the given style is modified.</td>
</tr>
<tr>
<td><code>RemoveElement</code></td>
<td>Removes an element from this map.</td>
</tr>
<tr>
<td><code>RemoveExtent</code></td>
<td>Removes the extent of this map.</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>RemoveReactor</td>
<td>Removes a reactor from this map.</td>
</tr>
<tr>
<td>RemoveScaleThreshold</td>
<td>Removes a scale threshold from this map.</td>
</tr>
<tr>
<td>RemoveScaleThreshold</td>
<td>Removes a scale threshold from this map.</td>
</tr>
<tr>
<td>Requery</td>
<td>Requeries objects belonging to the map.</td>
</tr>
<tr>
<td>ResumeFixingDO</td>
<td>Resumes fixing of the draw order of entities.</td>
</tr>
<tr>
<td></td>
<td>The draw order is fixed during this call. Any change that affects the draw</td>
</tr>
<tr>
<td></td>
<td>order after this method is called until the next SuspendFixingDO method</td>
</tr>
<tr>
<td></td>
<td>will be reflected immediately.</td>
</tr>
<tr>
<td></td>
<td>This method must be used in conjunction with SuspendFixingDO() and these</td>
</tr>
<tr>
<td></td>
<td>two methods must always be paired.</td>
</tr>
<tr>
<td>SaveLinkedFile</td>
<td>Saves the current database in its current state of stylization at the</td>
</tr>
<tr>
<td></td>
<td>current scale to the filename specified in a previous call to AcMapDMMap::</td>
</tr>
<tr>
<td></td>
<td>SetLinkedFileName(). The linked file contains a stylized map that can be</td>
</tr>
<tr>
<td></td>
<td>plotted or published to DWF.</td>
</tr>
<tr>
<td></td>
<td>Clicking the Update button in the AutoCAD Map user interface invokes</td>
</tr>
<tr>
<td></td>
<td>SaveLinkedFile(). SaveLinkedFile() does not call AcMapDMMap::UpdateStyli-</td>
</tr>
<tr>
<td></td>
<td>zation(); it is your application's responsibility to prepare the stylized</td>
</tr>
<tr>
<td></td>
<td>model. SaveLinkedFile() calls AcMapDMMap::GetCurrentScale(), followed by</td>
</tr>
<tr>
<td></td>
<td>AcMapDMMap::GetLinkedFileName(), with the current scale to obtain the target</td>
</tr>
<tr>
<td></td>
<td>filename.</td>
</tr>
<tr>
<td>SelectElementsWithStyleApplied</td>
<td>Retrieves elements that reference the specified style at the specified scale</td>
</tr>
<tr>
<td></td>
<td>threshold.</td>
</tr>
<tr>
<td>Set3dGridPercent</td>
<td>Sets the 3d grid percent of this map.</td>
</tr>
<tr>
<td>SetCurrentScale</td>
<td>Sets the current scale of this map, affects BOTH the arbitrary scale and</td>
</tr>
<tr>
<td></td>
<td>the classic Threshold.</td>
</tr>
<tr>
<td></td>
<td>Sets the current scale threshold of this map at</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><code>SetCurrentScaleThreshold</code></td>
<td>Sets the specified index.</td>
</tr>
<tr>
<td><code>SetDOMode</code></td>
<td>Sets the draw order mode for the map.</td>
</tr>
<tr>
<td><code>SetExaggeration</code></td>
<td>Sets the current exaggeration of this map.</td>
</tr>
<tr>
<td><code>SetExtent</code></td>
<td>Sets the extent of this map.</td>
</tr>
<tr>
<td><code>SetLineTypeScale</code></td>
<td>Sets the linetype scale for a specified scale. See also <code>GetLineTypeScale()</code>.</td>
</tr>
<tr>
<td><code>SetLinkedFileName</code></td>
<td>Sets the linked output filename. The linked output file contains the stylized objects of this map. By default, current scale is used in output. See also <code>GetLinkedFileName()</code>.</td>
</tr>
<tr>
<td><code>SetLinkToFileEnabled</code></td>
<td>Sets the status of the linked file capability.</td>
</tr>
<tr>
<td><code>SetName</code></td>
<td>Sets the name of this map. See also <code>GetName()</code>.</td>
</tr>
<tr>
<td><code>SetSun</code></td>
<td>Sets sun vector setting for current map.</td>
</tr>
<tr>
<td><code>SetTableStyle</code></td>
<td>Sets the table style. The table style is used for the legend. See also <code>GetTableStyle()</code>.</td>
</tr>
<tr>
<td><code>SetThumbnailDimensions</code></td>
<td>Sets the thumbnail width and height factors. These width and height factors are multiplied by the text height in the legend to determine the size of the thumbnail width and height. See also <code>GetThumbnailDimensions()</code>.</td>
</tr>
<tr>
<td><code>SetThumbnailFraming</code></td>
<td>Sets whether a box is drawn around the thumbnail graphic. See also <code>GetThumbnailFraming()</code>.</td>
</tr>
<tr>
<td><code>SetVisible</code></td>
<td>Sets the map visibility at the specified scale. See also <code>IsVisible()</code>. Invoked from within <code>erase()</code> before the erase actually occurs. See also <code>subErase()</code> in the AutoCAD ObjectARX Developer's Guide.</td>
</tr>
<tr>
<td><code>subErase</code></td>
<td>This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.</td>
</tr>
<tr>
<td><code>SuspendFixingDO</code></td>
<td>Suspends fixing of the draw order of entities.</td>
</tr>
<tr>
<td><code>UpdateStylization</code></td>
<td>Stylizes the current or updated selection at the current scale.</td>
</tr>
</tbody>
</table>

*subErase* Invoked from within `erase()` before the erase actually occurs. See also `subErase()` in the AutoCAD ObjectARX Developer's Guide.
Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
AcMapDMMap Class, AcMapDMMap Class

AcMapDMMap:: AddElement Method
AcMapDMMap Class | AcMapDMMap Class

Adds an element to this map.

```cpp
Acad::ErrorStatus AddElement(
    AcDbObjectId& Id,
    AcMapDMElement* pElement,
    AcMapDMAllDrawOrderItemsIterator& Position
);
```

Parameters Description
---
Id Output ID of the map.
pElement Input AcMapDMElement to add.
Position Input position at which to add the element. See also `AcMapDMAllDrawOrderItemsIterator`.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMMap Class, AcMapDMMap Class
AcMapDMMap:: AddScaleThreshold Method
AcMapDMMap Class | AcMapDMMap Class

Adds a scale threshold to this map - this function is not yet supported in AutoCAD Map.

```cpp
Acad::ErrorStatus AddScaleThreshold(
    double dNewScale,
    double dSourceScale = 0.
);
```

Parameters
- **dNewScale**: Input new scale threshold.
- **dSourceScale**: Input source scale threshold. The default value is zero.

Returns
- Returns Acad::eOk if successful; otherwise, returns a different error code.

Remarks

For every element, this function adds a new scale and associates it with the style set and with the source scale where dSourceScale is defined on the map. If dSourceScale is zero (the default), a new empty style set is created.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Copies a scale threshold to this map - this function is not yet supported in AutoCAD Map.

```cpp
Acad::ErrorStatus CopyScaleThreshold(
    double dNewScale,
    double dSourceScale
);
```

**Parameters**

- **dNewScale**: Input new scale threshold.
- **dSourceScale**: Input source scale threshold. The default value is zero.

**Returns**

- Returns Acad::eOk if successful; otherwise, returns a different error code.

**Remarks**

For every element, this function copies a new scale and associates it with the style set, and associates with the source scale where dSourceScale is defined on the map. If dSourceScale is zero (the default), a new empty style set is created.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Retrieves the current 3d grid percent of this map exactly as last input to Set3dGridPercent().

```cpp
int Get3dGridPercent() const;
```

Returns the 3d grid percent.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the current scale threshold of this map, as calculated from the last input to SetCurrentScale().

```cpp
double GetCurrentScaleThreshold() const;
```

Returns

Returns the scale.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at **support@toolsfactory.com**.
Retrieves the current scale of this map exactly as last input to SetCurrentScale().

```cpp
double GetCurrentUserScale() const;
```

Returns the scale.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Retrieves the current exaggeration of this map exactly as last input to SetExaggeration().

```cpp
double GetExaggeration() const;
```

Returns the exaggeration.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the current extent of this map exactly as last input to SetExtent().

```cpp
Acad::ErrorStatus GetExtent(
    double& dMinX,
    double& dMinY,
    double& dMaxX,
    double& dMaxY
) const;
```

Parameters

- **dMinX**: Output the extent min x value.
- **dMinY**: Output the extent min y value.
- **dMaxX**: Output the extent max x value. Output the extent max Y value.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the linked output filename. The linked output file contains the stylized objects of this map. By default, the current scale is used in output. See also SetLinkedFileName.

\begin{verbatim}
Acad::ErrorStatus GetLinkedFileName(
    const ACHAR*& kpszFileName,
    double dScale = 0.
) const;
\end{verbatim}

Parameters

- \textbf{kpszFileName} Output linked filename.
- \textbf{dScale} Input threshold scale value.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of \textit{Doc-O-Matic}. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at \texttt{support@toolsfactory.com}.  

Links

- \texttt{AcMapDMMAP Class}, \texttt{AcMapDMMAP Class}
- \texttt{AcMapDMMAP:: GetLinkedFileName Method}
- \texttt{AcMapDMMAP Class | AcMapDMMAP Class}
Retrieves the scale threshold at a specified index.

```cpp
Acad::ErrorStatus GetScaleThreshold(
    double& dScale,
    size_t iIndex
) const;
```

**Parameters**  
- `dScale`  
  - Output scale threshold.
- `iIndex`  
  - Input index of the desired scale threshold.

**Returns**  
Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Get sun vector setting for current map.

\begin{verbatim}
Acad::ErrorStatus GetSun(
    double& dAzimuth,
    double& dAltitude
) const;
\end{verbatim}

Parameters Description
\begin{itemize}
  \item dAzimuth Azimuth value of sun vector
  \item dAltitude Altitude value of sun vector
\end{itemize}

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves an upper-bound map threshold scale for the specified scale.

```cpp
double GetThresholdFor(
    double dDisplayScale
) const;
```

Parameters | Description
--- | ---
dDisplayScale | Input display scale.

Returns

Returns the map scale.

Created with a commercial version of [Doc-O-Matic](https://www.toolsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns the value used as the Infinity threshold scale in the Map.

```java
static double InfinityThreshold();
```

Returns the value used as the Infinity threshold scale in the Map.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Reports whether the input value is equal to the value used as the Infinity threshold scale in the Map.

```cpp
static bool IsInfinityThreshold(double dScale);
```

**Parameters**

- **dScale**: Input scale to compare to DMMap's "infinity" threshold.

**Returns**

Returns true if the input value is the Infinty threshold, otherwise returns false.
Changes the scale threshold value of an existing scale threshold.

```cpp
Acad::ErrorStatus ModifyScaleThreshold(
    double dOldScale,
    double dNewScale
);
```

Parameters

- **dOldScale**: Input old scale threshold.
- **dNewScale**: Input new scale threshold.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Changes the position of an element within this map.

\[
\text{Acad::ErrorStatus } \text{MoveElement(}
\text{const AcDbObjectId& } \text{Id},
\text{AcMapDMAllDrawOrderItemsIterator& } \text{Position})
\]

Parameters

- **Id**: Input ID of the element to move.
- **Position**: Input position to move the element to. See also \text{AcMapDMAllDrawOrderItemsIterator}.

Returns

Returns Acad::eOk if successful. Returns Acad::eInvalidIndex if the iterator position is invalid.

Remarks

Moving an element affects the draw order of entities that are part of element.

Created with a commercial version of \text{Doc-O-Matic}. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Counts the number of scale thresholds defined in this map.

```cpp
size_t NumScaleThresholds() const;
```

Returns the number of scales.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
AcMapDMMap:: RemoveElement Method

Removes an element from this map.

```
Acad::ErrorStatus RemoveElement(
    AcMapDMAllDrawOrderItemsIterator& Position
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Position</td>
<td>Input position of the element to remove. See also <code>AcMapDMAllDrawOrderItemsIterator</code></td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful. Returns Acad::eInvalidIndex if the iterator position is invalid.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMMap Class, AcMapDMMap Class
AcMapDMMap:: RemoveExtent Method
AcMapDMMap Class | AcMapDMMap Class

Removes the extent of this map.

Acad::ErrorStatus RemoveExtent();

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMMap Class, AcMapDMMap Class
AcMapDMMap:: RemoveScaleThreshold Method
AcMapDMMap Class | AcMapDMMap Class

Removes a scale threshold from this map.

Acad::ErrorStatus RemoveScaleThreshold(
    double dScale
);  
Parameters                Description
dScale                    Input value of the scale threshold to remove.

Returns

Returns Acad::eOk if successful. Returns Acad::eInvalidInput if the scale has not been defined. Returns Acad::eNotApplicable if the scale is DBL_MAX.

Remarks

If the scale is the current scale, this function dismisses stylization.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Remove a scale threshold from this map.

```cpp
Acad::ErrorStatus RemoveScaleThreshold(
    size_t iIndex
);
```

Parameters

- `iIndex`: Input index of the scale threshold to remove.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the 3d grid percent of this map.

```
Acad::ErrorStatus Set3dGridPercent(
    int iValue
);
```

**Parameters**

- `iValue`  
  Input percent value. The valid value is from 10 to 100.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the current scale of this map, affects BOTH the arbitrary scale and the classic Threshold.

```
Acad::ErrorStatus SetCurrentScale(
  double dScale,
  bool bRegen = true
);
```

Parameters

- **dScale**: Input arbitrary scale.
- **bRegen**: Input whether to have Elements regen their acquired entities, regardless of whether a threshold change is occurring.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Remarks

This method sets two internal values, an arbitrary "user scale" and the appropriate "map threshold scale", the latter being a threshold value corresponding to scale thresholds available for classic DM Elements. If the change in arbitrary scale does not require a threshold change, no threshold change will occur. Changing the current threshold is not allowed if stylization is on. If the Threshold will change, the caller is required to bracket this call with DismissSylization/UpdateStylization, which will requery the elements of the map.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the current scale threshold of this map at the specified index.

\[
\text{Acad::ErrorStatus SetCurrentScaleThreshold(}
\begin{array}{l}
\text{size_t iIndex)}
\end{array}
\]

Parameters | Description
--- | ---
iIndex | Input index of the scale threshold.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Remarks

Sets the current User scale to match the specified threshold. Setting the current threshold requeries the elements of the map and applies stylization if stylization is on.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the current exaggeration of this map.

```cpp
Acad::ErrorStatus SetExaggeration(
    double dValue
);
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dValue</td>
<td>Input exaggeration value.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the extent of this map.

```cpp
Acad::ErrorStatus SetExtent(
    double dMinX,
    double dMinY,
    double dMaxX,
    double dMaxY
);
```

Parameters

**dMinX**
- Input the extent min x value.

**dMinY**
- Input the extent min y value.

**dMaxX**
- Input the extent max x value.
- Input the extent max Y value.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the linked output filename. The linked output file contains the stylized objects of this map. By default, current scale is used in output. See also GetLinkedFileName().

```cpp
Acad::ErrorStatus SetLinkedFileName(
    const ACHAR* kpszFileName,
    double dScale = 0.
);
```

Parameters

- **kpszFileName**
  - Input linked filename.
- **dScale**
  - Input threshold scale value.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the name of this map. See also GetName().

```cpp
virtual Acad::ErrorStatus SetName(
    const ACHAR* pszName
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Description</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pszName</td>
<td>Input name of the map.</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at **support@toolsfactory.com**.
Sets sun vector setting for current map.

```
Acad::ErrorStatus SetSun(
    double dAzimuth,
    double dAltitude
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dAzimuth</td>
<td>Azimuth value of sun vector</td>
</tr>
<tr>
<td>dAltitude</td>
<td>Altitude value of sun vector</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the table style. The table style is used for the legend. See also GetTableStyle().

```cpp
Acad::ErrorStatus SetTableStyle(
    const ACHAR* pszTableStyleName
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pszTableStyleName</td>
<td>Input name of the table style.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if a table style with the specified name exists. Returns Acad::eInvalidInput if pszTableStyleName is a NULL pointer or empty string.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
An iterator over a collection of display-management map objects (AcMapDMMMap).

class AcMapDMMMapIterator;

File

DmMapManager.h

Methods

~AcMapDMMMapIterator  Destroys an instance of this class.

Done  Determines whether the iterator has reached the end of the collection.

GetObj  Retrieves the current object in the iteration.

Next  Advances to the next element in the iteration.

ObjectId  Retrieves the ID of the current object in the iteration.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Class AcMapDMMapManager Class

Manages display-management map objects (AcMapDMMap).

```c
class AcMapDMMapManager : public AcDbDictionaryWithDefault;
```

File DmMapManager.h

- Methods
  - ~AcMapDMMapManager: Destroys an instance of this class.
  - AcMapDMMapManager: Constructs an instance of this class.
  - AddDefaultMap: Creates a default map object.
  - Append: Adds a map.
  - CreateNewMap: Adds a new map with the default settings.
    - `dwgInFields`: Lets this object read its data. See also `dwgInFields()` in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.
    - `dwgOutFields`: Lets this object write its data. See also `dwgOutFields()` in the AutoCAD ObjectARX Developer's Guide.
  - `dxfInFields`: This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.
    - `dxfOutFields`: Lets this object write its data. See also `dxfOutFields()` in the AutoCAD ObjectARX Developer's Guide.
Export

Exports an XML string representation of a map. See also Import().

GetAt

Retrieves the ID of the named map.

GetAt

Retrieves the named map.

GetCurrent

Retrieves the ID of the current map.

GetCurrent

Retrieves the name of the current map.

Has

Determines whether a map with the specified ID exists.

Has

Determines whether a named map exists.

Implementation

Returns the implementation object.

Import

Imports an XML string to form a map - this function is not yet implemented in AutoCAD Map. See also Export().

NewIterator

Returns a new map iterator.

NumMaps

Counts the number of maps.

Remove

Removes the map with the specified ID.

Remove

Removes the named map.

ResetName

 Renames the named map.

ResetName

Renames the map with the specified ID.

SetCurrent

Sets the named map to be the current map.

SetCurrent

Sets the map with the specified ID to be the current map.

Clones the specified map to the destination database. See also wblockClone() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

Deep clones the specified objects and appends them to the specified container. The objects can come from multiple source databases, and must match the type of owner specified, but must be from a different database than the ownerId object. See also wblockCloneObjects() in the AutoCAD ObjectARX
Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
**Links**

- [AcMapDMMapManager Class](#)
- [AcMapDMMapManager Class](#)

**AcMapDMMapManager: Append Method**

- [AcMapDMMapManager Class](#) | [AcMapDMMapManager Class](#)

**Adds a map.**

```cpp
Acad::ErrorStatus Append(
    AcDbObjectId& Id,
    const ACHAR* pszName,
    AcMapDMMap* pMap
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Id</td>
<td>Output ID of the added map.</td>
</tr>
<tr>
<td>pszName</td>
<td>Input name of the map.</td>
</tr>
<tr>
<td>pMap</td>
<td>Input AcMapDMMap.</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](#). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Exports an XML string representation of a map. See also Import().

```cpp
Acad::ErrorStatus Export(
    ACHAR*& pszXMLString
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pszXMLString</td>
<td>Output XML string.</td>
</tr>
</tbody>
</table>

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapDMMapManager Class, AcMapDMMapManager Class
AcMapDMMapManager:: GetAt Method
AcMapDMMapManager Class | AcMapDMMapManager Class

Retrieves the ID of the named map.

Acad::ErrorStatus GetAt(
    AcDbObjectId& Id,
    const ACHAR* pszName
) const;

Parameters Description
Id Output ID of the map.
pszName Input name of the map.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the named map.

\[
\text{Acad::ErrorStatus GetAt(}
\quad \text{AcMapDMMMap}\&\ pMap,
\quad \text{const ACHAR}\*\ \text{pszName},
\quad \text{AcDb::OpenMode}\ \text{mode}
\quad ) \text{ const;}
\]

Parameters

- **pMap**
  - Output: AcMapDMMMap.
- **pszName**
  - Input: name of the map.
- **mode**
  - Input mode for opening the map.

Returns

- Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Retrieves the name of the current map.

```cpp
Acad::ErrorStatus GetCurrent(
    const ACHAR*& pszName
) const;
```

**Parameters**

- `pszName`: Output name of the current map.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determines whether a named map exists.

```cpp
bool Has(
    const ACHAR* pszName
) const;
```

### Parameters

**pszName**

Input name of the map to search for.

### Returns

Returns true if the map exists.

Created with a commercial version of [Doc-O-Matic](#). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Imports an XML string to form a map - this function is not yet implemented in AutoCAD Map. See also Export().

```cpp
Acad::ErrorStatus Import(
    const ACHAR* pszXMLString
);
```

### Parameters

- **pszXMLString**: Input XML string.

### Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Removes the named map.

```c
Acad::ErrorStatus Remove(
    const ACHAR* pszName
);
```

**Parameters**

<table>
<thead>
<tr>
<th>pszName</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input name of the map to remove.</td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful. Returns Acad::eNotApplicable if the specified map is the current map.

**Remarks**

The default map cannot be removed. If the current map is removed, the default map becomes the current map.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapDMMapManager Class, AcMapDMMapManager Class
AcMapDMMapManager:: ResetName Method
AcMapDMMapManager Class | AcMapDMMapManager Class

Renames the named map.

`Acad::ErrorStatus ResetName(
    const ACHAR* pszOldName,
    const ACHAR* pszNewName
);`

Parameters Description
pszOldName Input name of the map to rename.
pszNewName Input new map name.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Renames the map with the specified ID.

```cpp
Acad::ErrorStatus ResetName(
    const AcDbObjectId& mapId,
    const ACHAR* pszNewName
);
```

Parameters Description
---
mapId Input ID of the map to rename.
pszNewName Input new map name.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the named map to be the current map.

\[
\text{Acad::ErrorStatus SetCurrent(}
\text{\hspace{1cm} \textbf{const} ACHAR* pszName} 
\text{\hspace{1cm})};
\]

Parameters | Description  
--- | ---  
pszName | Input name of the map.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of \textbf{Doc-O-Matic}. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Base class used to notify an application of display-management events within a display-management map. Derive custom reactors from the AcMapDMMapReactor class.

To add a reactor: Derive your class AcMapDMMMyReactor from AcMapDMMapReactor. Override the virtual functions of this reactor base class. Use AcMapDMMap::AddReactor() to register an instance of the reactor. To unregister a reactor, use AcMapDMMap::RemoveReactor()..

class AcMapDMMMapReactor;

File
DMReactor.h

Methods
- **CurrentScaleModified** Invoked when the current scale is modified.
- **DismissStylizationBegin** Invoked just before stylization is dismissed.
- **DismissStylizationCancel** Invoked when a stylization dismissal is cancelled.
- **DismissStylizationEnd** Invoked just after stylization is dismissed.
- **ItemAppended** Invoked when a display-management item is added to the map.
- **ItemErased** Invoked when a display-management item is erased from the map.
- **ItemModified** Invoked when a display-management item is modified.
- **ScaleAdded** Invoked when a scale is added.
- **ScaleErased** Invoked when a scale is erased.
- **ScaleModified** Invoked when a scale is modified.
- **StyleAppended** Invoked when a style is added.
- **StyleErased** Invoked when a style is erased.
- **StyleModified** Invoked when a style is modified.
StyleReferenceAppended Invoked when a style reference is added.
StyleReferenceErased Invoked when a style reference is erased.
StyleReferenceModified Invoked when a style reference is modified.
UpdateStylizationBegin Invoked just before stylization occurs.
UpdateStylizationCancel Invoked when a stylization is cancelled.
UpdateStylizationEnd Invoked just after stylization occurs.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Base class used to notify the an application of display-management events within an AutoCAD Map project. Derive custom reactors from the AcMapDMProjectReactor class.

To add a reactor: Derive your class AcMapDMMyReactor from AcMapDMProjectReactor. Override the virtual functions of this reactor base class. Use AcMapDMDisplayManagement::AddProjectReactor() to register an instance of the reactor. To unregister a reactor, use AcMapDMDisplayManagement::RemoveProjectReactor()..

class AcMapDMProjectReactor;

File
DMReactor.h

Methods
- CategoryAppended: Invoked when a category is added to the style library.
- CategoryModified: Invoked when a category is modified.
- CategoryUnAppended: Invoked when a category is detached from the style library.
- MapAppended: Invoked when a new map is appended or after a reactor is attached to a project that has map(s) defined.
- MapGoodBye: Invoked when a map is unloaded.
- MapSetCurrentBegin: Invoked just before a map is set as the current map.
- MapSetCurrentEnd: Invoked just after a map is set as the current map.
- MapSetCurrentFails: Invoked when setting a map as current fails.
- MapUnAppended: Invoked when a map is detached.
- StyleAppended: Invoked when a style is added to the category.
- StyleModified: Invoked when a style is modified.
- StyleUnAppended: Invoked when a style is detached.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this
message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links

Classes

AcMapDMQueryDataSourceDescriptor Class

Classes

Describes a base data source for querying.

class AcMapDMQueryDataSourceDescriptor : public AcMapDMDataSourceDes

File

DmProjectQueryElement.h

Methods

~AcMapDMQueryDataSourceDescriptor

AcMapDMQueryDataSourceDescriptor

GetAcquisitionStatement

GetQuery

SetAcquisitionStatement

SetQuery

Destroy an instance of this class.

Constructs an instance of this class.

Retrieves the query's string representation.

Retrieves the query definition.

Sets the query string.

Sets the query definition.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the query's string representation.

```cpp
def virtual Acad::ErrorStatus GetAcquisitionStatement(
    ACHAR*& pszStatement
) const;
```

**Parameters**

- **pszStatement**: Output query string.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the query string.

```cpp
virtual Acad::ErrorStatus SetAcquisitionStatement(
    const ACHAR* pszStatement);
```

**Parameters**

**Description**

- **pszStatement**: Input query string.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.doc-omatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Defines a raster data source.

```cpp
class AcMapDMRasterDataSourceDescriptor : public AcMapDMDataSourceDescriptor
```

File

DmRasterElement.h

Methods

- `~AcMapDMRasterDataSourceDescriptor` Destroys an instance of this class.
- `AcMapDMRasterDataSourceDescriptor` Constructs an instance of this class.
- `GetAcquisitionStatement` Retrieves the query's string representation.
- `SetAcquisitionStatement` Sets the query string.

Created with a commercial version of [Doc-O-Matic](https://example.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the query's string representation.

```cpp
class AcMapDMRasterDataSourceDescriptor
{
public:
    virtual Acad::ErrorStatus GetAcquisitionStatement(
        ACHAR*& pszStatement
    ) const;
}
```

**Parameters**

- `pszStatement`: Output query string.

**Description**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the query string.

```cpp
virtual Acad::ErrorStatus SetAcquisitionStatement(
    const ACHAR* pszStatement
);
```

Parameters

- **pszStatement**: Input query string.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](#). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links

Classes

AcMapDMRasterElement Class

Represents a raster element that selects raster images.

class AcMapDMRasterElement : public AcMapDMElement;

File

DmRasterElement.h

Methods

~AcMapDMRasterElement

Destroys an instance of this class.

AcMapDMRasterElement

Constructs an instance of this class.

AcquireEntities

Runs the query against the current drawing.

ClonesObjectsFromExternalSource

Clones objects from the source drawings.

Let this object read its data. See also
dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded
function is called by the system as needed; it is unlikely that you will need to call it directly.

dwgInFields

Let this object write its data. See also
dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded
function is called by the system as needed; it is unlikely that you will need to call it directly.

dwgOutFields

dxfInFields

Let this object read its data. See also
dxfInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded
function is called by the system as needed; it is unlikely that you will need to call it directly.

dxfOutFields
**dxfOutFields**

dxfOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

**EvaluateExpressionValues**

Evaluates two expressions against the objects that meet the acquisition criteria.

**EvaluateExpressionValues**

Evaluates expression(s) against the element selection and returns values for the selected entities.

**GetAcquisitionCriteria**

Retrieves a query definition's data-source descriptor.

**GetAcquisitionCriteria**

Retrieves the query definition's data-source descriptor.

**SetAcquisitionCriteria**

Sets the query definition's data-source descriptor.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at **support@toolsfactory.com**.
virtual Acad::ErrorStatus EvaluateExpressionValues(
    AcArray<AcMapValue*>& arValues1,
    AcArray<AcMapValue*>& arValues2,
    const ACHAR* kpszExpression1,
    AcMap::EDataType kDataType1,
    const ACHAR* kpszExpression2,
    AcMap::EDataType kDataType2
);

Parameters

- arValues1: Output array of evaluated values.
- arValues2: Output array of evaluated values.
- kpszExpression1: Input expression to evaluate.
- kDataType1: Input return data type expected.
- kpszExpression2: Input expression to evaluate.
- kDataType2: Input return data type expected.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Remarks

In the case of query elements, objects are not cloned and the expressions are evaluated against the original objects. If the expressions are not applicable to an entity (the .AREA for a line, for example), the result for this entity has type AcMap::kUnknownType.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Evaluates expression(s) against the element selection and returns values for the selected entities.

```cpp
virtual Acad::ErrorStatus EvaluateExpressionValues(
    AcArray<AcMapValue*>& arValues,
    const ACHAR* kpszExpression,
    AcMap::EDataType kDataType
);
```

**Parameters**
- `arValues` Output array of evaluated values.
- `kpszExpression` Input expression to evaluate.
- `kDataType` Input return data type expected.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

**Remarks**

If entities have not yet been acquired, this function acquires them.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves a query definition's data-source descriptor.

```cpp
virtual Acad::ErrorStatus GetAcquisitionCriteria(
    ACHAR*& pszString
) const;
```

**Parameters**

- `pszString`  
  Description: Output pointer to the data-source descriptor, as a string representation.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Encapsulates raster-image-related properties.

```cpp
class AcMapDMRasterStyle : public AcMapDMStyle;
```

File

DmRasterStyle.h

Methods

- `~AcMapDMRasterStyle` Destroys an instance of this class.
- `AcMapDMRasterStyle` Constructs an instance of this class.
- `Apply` Applies a style to a raster.
- `ClearBrightness` Clears image brightness. See also `IsBrightnessCleared()`.
- `ClearContrast` Clears image contrast. See also `IsContrastCleared()`.
- `ClearFade` Clears image fade. See also `IsFadeCleared()`.
- `ClearTransparency` Clears image transparency. See also `IsTransparencyCleared()`.
- `clone` Clones the object. See also `clone()` in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.
- `copyFrom` Copies the contents of an object into the messaged object, if feasible. See also `copyFrom()` in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.
- `DeleteCookie` Deletes the style's cookie during database destruction.
- `Dismiss` Clears the stylization of a raster.
- `Lets this object read its data. See also dwgInFields()` in the AutoCAD ObjectARX Developer's Guide. This
<table>
<thead>
<tr>
<th></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>dwgInFields</strong></td>
<td>overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.</td>
</tr>
<tr>
<td><strong>dwgOutFields</strong></td>
<td>Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.</td>
</tr>
<tr>
<td><strong>dxfInFields</strong></td>
<td>Lets this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.</td>
</tr>
<tr>
<td><strong>dxfOutFields</strong></td>
<td>Lets this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.</td>
</tr>
<tr>
<td><strong>Enable</strong></td>
<td>Enables or disables the style of a raster.</td>
</tr>
<tr>
<td><strong>GetBrightness</strong></td>
<td>Retrieves image brightness. See also SetBrightness().</td>
</tr>
<tr>
<td><strong>GetContrast</strong></td>
<td>Retrieves image contrast. See also SetContrast().</td>
</tr>
<tr>
<td><strong>GetFade</strong></td>
<td>Retrieves image fade. See also SetFade().</td>
</tr>
<tr>
<td><strong>GetTransparency</strong></td>
<td>Retrieves image transparency. See also SetTransparency().</td>
</tr>
<tr>
<td><strong>IsBrightnessCleared</strong></td>
<td>Determines whether image brightness is cleared. See also ClearBrightness().</td>
</tr>
<tr>
<td><strong>IsContrastCleared</strong></td>
<td>Determines whether image contrast is cleared. See also ClearContrast().</td>
</tr>
<tr>
<td><strong>IsFadeCleared</strong></td>
<td>Determines whether image fade is cleared. See also ClearFade().</td>
</tr>
<tr>
<td><strong>IsTransparencyCleared</strong></td>
<td>Determines whether image transparency is cleared. See also ClearTransparency().</td>
</tr>
<tr>
<td><strong>Preview</strong></td>
<td>Applies style traits to yield a preview of the specified raster.</td>
</tr>
<tr>
<td><strong>SetBrightness</strong></td>
<td>Sets image brightness. See also GetBrightness().</td>
</tr>
<tr>
<td><strong>SetContrast</strong></td>
<td>Sets image contrast. See also GetContrast().</td>
</tr>
<tr>
<td><strong>SetFade</strong></td>
<td>Sets image fade. See also GetFade().</td>
</tr>
<tr>
<td><strong>SetTransparency</strong></td>
<td>Sets image transparency. See also GetTransparency().</td>
</tr>
</tbody>
</table>
UnApply
Removes the style from a raster.

Update
Retrieves and stylizes a raster.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMRasterStyle Class, AcMapDMRasterStyle Class
AcMapDMRasterStyle:: Preview Method
AcMapDMRasterStyle Class | AcMapDMRasterStyle Class

Applies style traits to yield a preview of the specified raster.

**virtual Acad::ErrorStatus Preview**

```cpp
    Acad::ErrorStatus Preview(
        AcDbObjectIdArray& createdEntities,
        AcDbObjectId targetEntId,
        void* internalUse
    );
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>createdEntities</td>
<td>Output array of created alteration entities.</td>
</tr>
<tr>
<td>targetEntId</td>
<td>Input ID of the preview entity.</td>
</tr>
<tr>
<td>internalUse</td>
<td>Reserved. For internal use only.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Remarks

For stylization entity styles such as text, hatch, and annotation, this function returns an array of created entities.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Provides an annotation alteration style for stylization. Several properties can be represented as either a concrete type or as an ADE expression (in such cases, an API is provided for each representation). Setting a property by expression clears any value previously set by a concrete type. Likewise, setting any property by concrete type clears any value previously set in an expression. An API, XxxIsExpression() is provided to test which representation is currently in effect for each property. To represent the """" state, set the expression to NULL for that property. Using the wrongGetXxx function for the current representation of a property will return Acad::eWrongObjectType. The const ACHAR * & output parameters are valid immediately upon return and point directly to member data. The caller must not delete the pointer. If the caller desires to keep the value for an indefinite time, the caller must copy the output const ACHAR * & to its own memory and manage its lifetime.

```cpp
class AcMapDMSEAnnotationStyle : public AcMapDMStylizationEntityStyl
```

File
DmSEAnnotationStyle.h

Methods
- ~AcMapDMSEAnnotationStyle
  Destroys an instance of this class.
- AcMapDMSEAnnotationStyle
  Constructs an instance of this class.
- clone
  Clones the object. See also clone() in the AutoCAD ObjectARX Developer's Guide.
  This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.
- CoExistenceType
  Styles may declare preferences for how they will co-exist with other styles under a single Element.
- ColorIsExpression
  Determines whether the color is an expression.
**copyFrom**
Copies the contents of an object into the messaged object, if feasible. See also `copyFrom()` in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

**DeleteCookie**
Deletes the style's cookie during database destruction.

**Dismiss**
Removes topology stylization information that is used during regeneration.

**Dismiss**
Removes stylization information that is used during regeneration.

**dwgInFields**
Lets this object read its data. See also `dwgInFields()` in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

**dwgOutFields**
Lets this object write its data. See also `dwgOutFields()` in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

**dxfInFields**
Lets this object read its data. See also `dxfInFields()` in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

**dxfOutFields**
Lets this object write its data. See also `dxfOutFields()` in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

**Enable**
Enables or disables the style for the specified topology.

**Enable**
Enables or disables the style for the specified entity.

**GetAnnotationTemplate**
Retrieves the annotation template ID.
<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>GetColor</strong></td>
<td>Retrieves the color.</td>
</tr>
<tr>
<td><strong>GetColorExpression</strong></td>
<td>Retrieves the color expression.</td>
</tr>
<tr>
<td><strong>GetLayer</strong></td>
<td>Retrieves the layer id.</td>
</tr>
<tr>
<td><strong>GetLayerExpression</strong></td>
<td>Retrieves the layer expression.</td>
</tr>
<tr>
<td><strong>GetLinetype</strong></td>
<td>Retrieves the linetype id.</td>
</tr>
<tr>
<td><strong>GetLinetypeExpression</strong></td>
<td>Retrieves the linetype expression.</td>
</tr>
<tr>
<td><strong>GetLineWeight</strong></td>
<td>Retrieves the lineweight.</td>
</tr>
<tr>
<td><strong>GetLineWeightExpression</strong></td>
<td>Retrieves the lineweight expression.</td>
</tr>
<tr>
<td><strong>GetLocationExpression</strong></td>
<td>Retrieves the location expression.</td>
</tr>
<tr>
<td><strong>GetRotation</strong></td>
<td>Retrieves the rotation.</td>
</tr>
<tr>
<td><strong>GetRotationExpression</strong></td>
<td>Retrieves the rotation expression.</td>
</tr>
<tr>
<td><strong>GetScale</strong></td>
<td>Retrieves the scale.</td>
</tr>
<tr>
<td><strong>GetScaleExpression</strong></td>
<td>Retrieves the scale expression.</td>
</tr>
<tr>
<td><strong>GetStylizationEntities</strong></td>
<td>Retrieves the AcDbObjectId of any stylization entities created during an Update().</td>
</tr>
<tr>
<td><strong>IsSymbolStyle</strong></td>
<td>Indicates that this style is a symbol style instead of an annotation style.</td>
</tr>
<tr>
<td><strong>LayerIsExpression</strong></td>
<td>Determines whether the layer is an expression.</td>
</tr>
<tr>
<td><strong>LinetypeIsExpression</strong></td>
<td>Determines whether the linetype is an expression.</td>
</tr>
<tr>
<td><strong>LineWeightIsExpression</strong></td>
<td>Determines whether the lineweight is an expression.</td>
</tr>
<tr>
<td><strong>Preview</strong></td>
<td>Applies style traits to yield a preview of the specified entity.</td>
</tr>
<tr>
<td><strong>RotationIsExpression</strong></td>
<td>Determines whether the rotation is an expression.</td>
</tr>
<tr>
<td><strong>ScaleIsExpression</strong></td>
<td>Determines whether the scale is an expression.</td>
</tr>
<tr>
<td><strong>SetAnnotationTemplate</strong></td>
<td>Sets the annotation template ID or block ID.</td>
</tr>
<tr>
<td><strong>SetColor</strong></td>
<td>Sets the color.</td>
</tr>
<tr>
<td><strong>SetColorExpression</strong></td>
<td>Sets the color expression.</td>
</tr>
<tr>
<td><strong>SetLayer</strong></td>
<td>Sets the layer id.</td>
</tr>
<tr>
<td><strong>SetLayerExpression</strong></td>
<td>Sets the layer expression.</td>
</tr>
</tbody>
</table>
SetLinetype
Sets the linetype id.

SetLinetypeExpression
Sets the linetype expression.

SetLineWeight
Sets the lineweight.

SetLineWeightExpression
Sets the lineweight expression.

SetLocationExpression
Sets the location expression.

SetRotation
Sets the rotation.

SetRotationExpression
Sets the rotation expression.

SetScale
Sets the scale.

SetScaleExpression
Sets the scale expression.

Update
Builds topology stylization information for later regeneration.

Update
Builds stylization information for later regeneration.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Styles may declare preferences for how they will co-exist with other styles under a single Element.

```cpp
virtual AcMapDMStyle::CoExistenceFlags CoExistenceType() const;
```

Returns Style's CoExistence preferences.

Remarks

They may coexist with all other styles, only other styles of the same type, or may require being the only style under an Element.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Removes topology stylization information that is used during regeneration.

```cpp
virtual Acad::ErrorStatus Dismiss(
    void*& pCookie,
    const ACHAR* pTopoName,
    Adesk::IntDbId lTopoId,
    Adesk::UInt32 flag = 0
);
```

**Parameters**

- **pCookie**: Output cookie stored by Update().
- **pTopoName**: Input topology name.
- **lTopoId**: Input topology element ID.
- **flag**: Reserved. For internal use only.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Enables or disables the style for the specified topology.

```cpp
virtual Acad::ErrorStatus Enable(
    void*& pCookie,
    const ACHAR* pTopoName,
    Adesk::IntDbId lTopoId,
    bool bEnable,
    Adesk::UInt32 flag = 0
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pCookie</td>
<td>Input opaque memory value.</td>
</tr>
<tr>
<td>pTopoName</td>
<td>Input topology name.</td>
</tr>
<tr>
<td>lTopoId</td>
<td>Input topology element ID.</td>
</tr>
<tr>
<td>bEnable</td>
<td>Input true to enable the style.</td>
</tr>
<tr>
<td>flag</td>
<td>Reserved. For internal use only.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMSEAnnotationStyle Class, AcMapDMSEAnnotationStyle Class
AcMapDMSEAnnotationStyle:: getColorExpression Method
AcMapDMSEAnnotationStyle Class | AcMapDMSEAnnotationStyle Class

Retrieves the color expression.

Acad::ErrorStatus getColorExpression(
    const ACHAR*& pColorExp
) const;

Parameters Description
pColorExp Output color expression.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the layer expression.

```cpp
Acad::ErrorStatus GetLayerExpression(
    const ACHAR*& pLayerExp
) const;
```

**Parameters**
- `pLayerExp`: Output layer expression.

**Returns**
- Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the linetype expression.

```cpp
Acad::ErrorStatus GetLinetypeExpression(
    const ACHAR*& pLinetypeExp
) const;
```

**Parameters**
- `pLinetypeExp` Output linetype expression.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Retrieves the lineweight expression.

```
Acad::ErrorStatus GetLineWeightExpression(
    const ACHAR*& pLineWeightExp
) const;
```

**Parameters**  
**Description**

- `pLineWeightExp`  
  Output lineweight expression.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Retrieves the location expression.

```cpp
Acad::ErrorStatus GetLocationExpression(
    const ACHAR*& pLocationExp
) const;
```

**Parameters**
- `pLocationExp`: Output location expression.

**Returns**
- Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the rotation expression.

```cpp
Acad::ErrorStatus GetRotationExpression(
    const ACHAR*& pRotationExp
) const;
```

**Parameters**
- `pRotationExp`: Output rotation expression.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the scale expression.

```cpp
Acad::ErrorStatus GetScaleExpression(
    const ACHAR*& pScaleExp
) const;
```

**Parameters**
- `pScaleExp`: Output scale expression.

**Description**
Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Applies style traits to yield a preview of the specified entity.

```cpp
virtual Acad::ErrorStatus Preview(
    AcDbObjectIdArray& createdEntities,
    AcDbObjectId targetEntId,
    void* internalUse
);
```

**Parameters**

- `createdEntities` Output array of IDs of created alteration entities.
- `targetEntId` Input ID of the preview entity.
- `internalUse` Reserved. For internal use only.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

**Remarks**

For stylization entity styles such as text, hatch, and annotation, this function returns an array of created entities.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the color expression.

```
Acad::ErrorStatus SetColorExpression(
    const ACHAR* pColorExp
);
```

Parameters | Description
--- | ---
pColorExp | Input color expression.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the layer expression.

```
Acad::ErrorStatus SetLayerExpression(
    const ACHAR* pLayerExp
);
```

**Parameters**
- `pLayerExp`  
  Description: Input layer expression.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the linetype expression.

```cpp
Acad::ErrorStatus SetLinetypeExpression(
    const ACHAR* pLinetypeExp
);
```

Parameters | Description
---|---
pLinetypeExp | Input linetype expression.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the lineweight expression.

```cpp
Acad::ErrorStatus SetLineWeightExpression(
    const ACHAR* pLineWeightExp
);
```

Parameters

- **pLineWeightExp**: Input lineweight expression.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the location expression.

```cpp
Acad::ErrorStatus SetLocationExpression(
    const ACHAR* pLocationExp
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pLocationExp</td>
<td>Input location expression.</td>
</tr>
</tbody>
</table>

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.doc-omatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the rotation expression.

```cpp
Acad::ErrorStatus SetRotationExpression(
    const ACHAR* pRotationExp
);```

**Parameters**

- `pRotationExp`: Input rotation expression.

**Description**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the scale expression.

```cpp
Acad::ErrorStatus SetScaleExpression(
    const ACHAR* pScaleExp
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pScaleExp</td>
<td>Input scale expression.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Builds topology stylization information for later regeneration.

```cpp
virtual Acad::ErrorStatus Update(
    void* & pCookie,
    const ACHAR* pTopoName,
    Adesk::IntDbId lTopoId,
    Adesk::UInt32 flag = 0
);
```

### Parameters

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pCookie</td>
</tr>
<tr>
<td>Input opaque memory value for later access by Dismiss().</td>
</tr>
<tr>
<td>pTopoName</td>
</tr>
<tr>
<td>Input topology name.</td>
</tr>
<tr>
<td>lTopoId</td>
</tr>
<tr>
<td>Input topology element ID.</td>
</tr>
<tr>
<td>flag</td>
</tr>
<tr>
<td>Reserved. For internal use only.</td>
</tr>
</tbody>
</table>

### Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

### Remarks

Derived classes can perform any one-time Update()-time actions here. At Dismiss() time, the implementation can do any required work to clean up storage created at Update() time.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Provides a hatch alteration style for stylization. Several properties can be represented as either a concrete type or as an ADE expression (in such cases, an API is provided for each representation). Setting a property by expression clears any value previously set by a concrete type. Likewise, setting any property by concrete type clears any value previously set in an expression. An API, XxxIsExpression() is provided to test which representation is currently in effect for each property. To represent the """" state, set the expression to NULL for that property. Using the wrong GetXxx function for the current representation of a property will return Acad::eWrongObjectType. The const ACHAR * & output parameters are valid immediately upon return and point directly to member data. The caller must not delete the pointer. If the caller desires to keep the value for an indefinite time, the caller must copy the output const ACHAR* & to its own memory and manage its lifetime.

class AcMapDMSEHatchStyle : public AcMapDMStylizationEntityStyle;

DmSEHatchStyle.h

Methods

~AcMapDMSEHatchStyle
Destroys an instance of this class.

AcMapDMSEHatchStyle
Constructs an instance of this class.

clone
Clones the object. See also clone() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

ColorIsExpression
Determines whether the color is an expression.

copyFrom
Copies the contents of an object into the messaged object, if feasible. See also copyFrom() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it
**DeleteCookie**

Deletes the style's cookie during database destruction.

**Dismiss**

Removes topology stylization information that is used during regeneration.

**Dismiss**

Removes stylization information that is used during regeneration.

**dwgInFields**

Lets this object read its data. See also `dwgInFields()` in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

**dwgOutFields**

Lets this object write its data. See also `dwgOutFields()` in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

**dxfInFields**

This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

**dxfOutFields**

This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

**Enable**

Enables or disables the style for the specified topology.

**Enable**

Enables or disables the style for the specified entity.

**GetColor**

Retrieves the color.

**GetColorExpression**

Retrieves the color expression.

**GetHatchPatternName**

Retrieves the hatch pattern name.

**GetLayer**

Retrieves the layer id.

**GetLayer**

Retrieves the layer name.
GetLayerExpression
Retrieves the layer expression.

GetRotation
Retrieves the rotation.

GetRotationExpression
Retrieves the rotation expression.

GetScale
Retrieves the scale.

GetScaleExpression
Retrieves the scale expression.

GetStylizationEntities
Retrieves the AcDbObjectIds of any stylization entities created during an Update().

LayerIsExpression
Determines whether the layer is an expression.

Preview
Applies style traits to yield a preview of the specified entity.

RotationIsExpression
Determines whether the rotation is an expression.

ScaleIsExpression
Determines whether the scale is an expression.

SetColor
Sets the color.

SetColorExpression
Sets the color expression.

SetHatchPattern
Sets the hatch pattern name.

SetLayer
Sets the layer name.

SetLayer
Sets the layer id.

SetLayerExpression
Sets the layer expression.

SetRotation
Sets the rotation.

SetRotationExpression
Sets the rotation expression.

SetScale
Sets the scale.

SetScaleExpression
Sets the scale expression.

Update
Builds topology stylization information for later regeneration.

Update
Builds stylization information for later regeneration.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Removes topology stylization information that is used during regeneration.

```cpp
virtual Acad::ErrorStatus Dismiss(
    void* & pCookie,
    const ACHAR* pTopoName,
    Adesk::IntDbId lTopoId,
    Adesk::UInt32 flag = 0
);
```

**Parameters**

- **pCookie**: Output cookie stored by Update().
- **pTopoName**: Input topology name.
- **lTopoId**: Input topology element ID.
- **flag**: Reserved. For internal use only.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](http://example.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Enables or disables the style for the specified topology.

```cpp
virtual Acad::ErrorStatus Enable(
    void*& pCookie,
    const ACHAR* pTopoName,
    Adesk::IntDbId lTopoId,
    bool bEnable,
    Adesk::UInt32 flag = 0
);
```

**Parameters**

- `pCookie` Input opaque memory value.
- `pTopoName` Input topology name.
- `lTopoId` Input topology element ID.
- `bEnable` Input true to enable the style.
- `flag` Reserved. For internal use only.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the color expression.

```cpp
Acad::ErrorStatus GetColorExpression(
    const ACHAR*& pColorExp
) const;
```

**Parameters**
- pColorExp: Output color expression.

**Returns**
Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMSEHatchStyle Class, AcMapDMSEHatchStyle Class
AcMapDMSEHatchStyle:: GetHatchPatternName Method
AcMapDMSEHatchStyle Class | AcMapDMSEHatchStyle Class

Retrieves the hatch pattern name.

```cpp
Acad::ErrorStatus GetHatchPatternName(
    const ACHAR*& pHatchPatternName
) const;
```

**Parameters**

| pHatchPatternName | Output hatch pattern name. |

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.doc-o-matic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Retrieves the layer name.

Acad::ErrorStatus GetLayer(
    const ACHAR*& pLayerName
) const;

Parameters
    pLayerName
Description
    Output layer name.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the layer expression.

```cpp
Acad::ErrorStatus GetLayerExpression(
    const ACHAR*& pLayerExp
) const;
```

**Parameters**
- `pLayerExp`: Output layer expression.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the rotation expression.

```cpp
Acad::ErrorStatus GetRotationExpression(
    const ACHAR*& pRotationExp
) const;
```

**Parameters**

- `pRotationExp`: Output rotation expression.

**Returns**

Returns `Acad::eOk` if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the scale expression.

```cpp
Acad::ErrorStatus GetScaleExpression(
    const ACHAR*& pScaleExp
) const;
```

**Parameters**
- `pScaleExp` Output scale expression.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://toolsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Applies style traits to yield a preview of the specified entity.

```cpp
virtual Acad::ErrorStatus Preview(
    AcDbObjectIdArray& createdEntities,
    AcDbObjectId targetEntId,
    void* internalUse
);
```

### Parameters

- **createdEntities**
  - **Description**: Output array of IDs of created alteration entities.

- **targetEntId**
  - **Description**: Input ID of the preview entity.

- **internalUse**
  - **Description**: Reserved. For internal use only.

### Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

### Remarks

For stylization entity styles such as text, hatch, and annotation, this function returns an array of created entities.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the color expression.

```cpp
Acad::ErrorStatus SetColorExpression(
    const ACHAR* pColorExp
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pColorExp</td>
<td>Input color expression.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the hatch pattern name.

```cpp
Acad::ErrorStatus SetHatchPattern(
    const ACHAR* pHatchPatternName
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pHatchPatternName</td>
<td>Input hatch pattern name.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the layer name.

```cpp
Acad::ErrorStatus SetLayer(
    const ACHAR* pLayerName
);
```

Parameters

- **pLayerName**: Input layer name.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.
Sets the layer expression.

`Acad::ErrorStatus SetLayerExpression(`
```
    const ACHAR* pLayerExp
```
`);

Parameters Description
--|---------------
pLayerExp | Input layer expression.

Returns

Returns `Acad::eOk` if successful; otherwise, returns a different error code.

Created with a commercial version of `Doc-O-Matic`. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at `support@toolsfactory.com`.
Sets the rotation expression.

```cpp
Acad::ErrorStatus SetRotationExpression(
    const ACHAR* pRotationExp
);
```

Parameters

<table>
<thead>
<tr>
<th>Description</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>pRotationExp</td>
<td>Input rotation expression.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.toolsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the scale expression.

    Acad::ErrorStatus SetScaleExpression(
        const ACHAR* pScaleExp
    );

Parameters

    pScaleExp Input scale expression.

Returns

    Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Builds topology stylization information for later regeneration.

```
virtual Acad::ErrorStatus Update(
    void* & pCookie,
    const ACHAR* pTopoName,
    Adesk::IntDbId lTopoId,
    Adesk::UInt32 flag = 0
);
```

Parameters

- **pCookie**: Input opaque memory value for later access by `Dismiss()`.
- **pTopoName**: Input topology name.
- **lTopoId**: Input topology element ID.
- **flag**: Reserved. For internal use only.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Remarks

Derived classes can perform any one-time `Update()`-time actions here. At `Dismiss()` time, the implementation can do any required work to clean up storage created at `Update()` time.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Provides a text alteration style for stylization. Several properties can be represented as either a concrete type or as an ADE expression (in such cases, an API is provided for each representation). Setting a property by expression clears any value previously set by a concrete type. Likewise, setting any property by concrete type clears any value previously set in an expression. An API, XxxIsExpression() is provided to test which representation is currently in effect for each property. To represent the "" state, set the expression to NULL for that property. Using the wrong GetXxx function for the current representation of a property will return Acad::eWrongObjectType. The const ACHAR *& output parameters are valid immediately upon return and point directly to member data. The caller must not delete the pointer. If the caller desires to keep the value for an indefinite time, the caller must copy the output const ACHAR*& to its own memory and manage its lifetime.

class AcMapDMSETextStyle : public AcMapDMStylizationEntityStyle;

File

DmSETextStyle.h

Enumerations

- Justification
  Enumerates the types of text justification.

Methods

- ~AcMapDMSETextStyle
  Destroys an instance of this class.
- AcMapDMSETextStyle
  Constructs an instance of this class.
- clone
  Clones the object. See also clone() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.
- ColorIsExpression
  Determines whether the color is an expression. Copies the contents of an object into the messaged object, if feasible. See also copyFrom() in the
copyFrom

AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

DeleteCookie

Deletes the style's cookie during database destruction.

Dismiss

Removes topology stylization information that is used during regeneration.

Dismiss

Removes stylization information that is used during regeneration.

Let this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide.

dwgInFields

This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

Let this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide.

dwgOutFields

Let this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide.

dxfInFields

This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

Let this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide.

dxfOutFields

Enables or disables the style for the specified topology.

Enable

Enables or disables the style for the specified entity.

Enable

Retrieves the text color.

GetColor

Retrieves the color expression.

GetColorExpression

Retrieves the text height.

GetHeight
GetHeightExpression
Retrieves the height expression.

GetJustification
Retrieves the text justification.

GetJustificationExpression
Retrieves the justification expression.

GetLayer
Retrieves the text layer id.

GetLayer
Retrieves the text layer.

GetLayerExpression
Retrieves the layer expression.

GetLocationExpression
Retrieves the location expression.

GetRotation
Retrieves the text rotation.

GetRotationExpression
Retrieves the rotation expression.

GetStyle
Retrieves the text style id.

GetStyle
Retrieves the text style name.

GetStyleExpression
Retrieves the style expression.

GetStylizationEntities
Retrieves the AcDbObjectIds of any stylization entities created during an Update().

GetTextStringExpression
Retrieves the text string expression.

HeightIsExpression
Determines whether the height is an expression.

JustificationIsExpression
Determines whether justification is an expression.

LayerIsExpression
Determines whether the layer is an expression.

Preview
Applies style traits to yield a preview of the specified entity.

RotationIsExpression
Determines whether the rotation is an expression.

SetColor
Sets the text color.

SetColorExpression
Sets the color expression.

SetHeight
Sets the text height.

SetHeightExpression
Sets the height expression.

SetJustification
Sets the text justification.

SetJustificationExpression
Sets the justification expression.

SetLayer
Sets the layer by name.

SetLayer
Sets the layer by id.

SetLayerExpression
Sets the layer expression.

SetLocationExpression
Sets the location expression.

SetRotation
Sets the text rotation.

SetRotationExpression
Sets the rotation expression.
<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SetStyle</td>
<td>Sets the text style by name.</td>
</tr>
<tr>
<td>SetStyle</td>
<td>Sets the text style by id.</td>
</tr>
<tr>
<td>SetStyleExpression</td>
<td>Sets the style expression.</td>
</tr>
<tr>
<td>SetTextStringExpression</td>
<td>Sets the text string expression.</td>
</tr>
<tr>
<td>StyleIsExpression</td>
<td>Determines whether the style is an expression.</td>
</tr>
<tr>
<td>Update</td>
<td>Builds topology stylization information for later regeneration.</td>
</tr>
<tr>
<td>Update</td>
<td>Builds stylization information for later regeneration.</td>
</tr>
</tbody>
</table>

Created with a commercial version of [Doc-O-Matic](http://www.toolsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMSETextStyle Class, AcMapDMSETextStyle Class
AcMapDMSETextStyle::Dismiss Method
AcMapDMSETextStyle Class | AcMapDMSETextStyle Class

Removes topology stylization information that is used during regeneration.

```cpp
virtual Acad::ErrorStatus Dismiss(
    void*& pCookie,
    const ACHAR* pTopoName,
    Adesk::IntDbId lTopoId,
    Adesk::UInt32 flag = 0
);
```

Parameters

- **pCookie**: Output cookie stored by Update().
- **pTopoName**: Input topology name.
- **lTopoId**: Input topology element ID.
- **flag**: Reserved. For internal use only.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Enables or disables the style for the specified topology.

```cpp
virtual Acad::ErrorStatus Enable(
    void* & pCookie,
    const ACHAR* pTopoName,
    Adesk::IntDbId lTopoId,
    bool bEnable,
    Adesk::UInt32 flag = 0
);
```

### Parameters

- **pCookie**: Input opaque memory value.
- **pTopoName**: Input topology name.
- **lTopoId**: Input topology element ID.
- **bEnable**: Input true to enable the style.
- **flag**: Reserved. For internal use only.

### Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Retrieves the color expression.

```cpp
Acad::ErrorStatus GetColorExpression(
    const ACHAR*& pColorExp
) const;
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pColorExp</td>
<td>Output color expression.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the height expression.

```
Acad::ErrorStatus GetHeightExpression(
    const ACHAR*& pHeightExp
) const;
```

**Parameters**

- `pHeightExp`: Output height expression.

**Returns**

Returns Acad::eOk if successful.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the justification expression.

```cpp
Acad::ErrorStatus GetJustificationExpression(
    const ACHAR*& pJustificationExp
) const;
```

Parameters | Description
---|---

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.doc-omatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Retrieves the text layer.

```cpp
Acad::ErrorStatus GetLayer(
    const ACHAR*& pLayerName
) const;
```

Parameters

- `pLayerName`: Output layer name.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the layer expression.

```
Acad::ErrorStatus GetLayerExpression(
    const ACHAR*& pLayerExp
) const;
```

Parameters

- `pLayerExp` Output layer expression.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the location expression.

```cpp
Acad::ErrorStatus GetLocationExpression(
    const ACHAR*& pLocationExpression
) const;
```

Parameters |
---|---
pLocationExpression | Output location expression.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the rotation expression.

```cpp
Acad::ErrorStatus GetRotationExpression(
    const ACHAR*& pRotationExp
) const;
```

Parameters:  
- `pRotationExp`  

Description:  
Output rotation expression.

Returns:  
- Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the text style name.

```cpp
Acad::ErrorStatus GetStyle(
    const ACHAR*& pStyleName
) const;
```

**Parameters**
- `pStyleName`: Output style name.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the style expression.

```cpp
Acad::ErrorStatus GetStyleExpression(
    const ACHAR*& pStyleExp
) const;
```

**Parameters**

- `pStyleExp`
  
  - **Description**: Output style expression.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the text string expression.

```cpp
Acad::ErrorStatus GetTextStringExpression(
    const ACHAR*& pTextExpression
) const;
```

**Parameters**
- `pTextExpression`: Output text string expression.

**Returns**
- Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Applies style traits to yield a preview of the specified entity.

```cpp
virtual Acad::ErrorStatus Preview(
    AcDbObjectIdArray& createdEntities,
    AcDbObjectId targetEntId,
    void* internalUse
);
```

**Parameters**

- `createdEntities`: Output array of IDs of created alteration entities.
- `targetEntId`: Input ID of the preview entity.
- `internalUse`: Reserved. For internal use only.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

**Remarks**

For stylization entity styles such as text, hatch, and annotation, this function returns an array of created entities.

Created with a commercial version of [Doc-O-Matic](#). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the color expression.

```cpp
Acad::ErrorStatus SetColorExpression(
    const ACHAR* pColorExp
);
```

Parameters

<table>
<thead>
<tr>
<th>pColorExp</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Input color expression.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the height expression.

```
Acad::ErrorStatus SetHeightExpression(
    const ACHAR* pHeightExp
);
```

Parameters

- **pHeightExp**: Input height string expression.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](http://www.doc-o-matic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the justification expression.

```cpp
Acad::ErrorStatus SetJustificationExpression(
    const ACHAR* pJustificationExp
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pJustificationExp</td>
<td>Input justification expression.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the layer by name.

```cpp
Acad::ErrorStatus SetLayer(
    const ACHAR* pLayerName
);
```

Parameters  | Description
-------------|-------------
pLayerName   | Input layer name.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the layer expression.

\begin{verbatim}
Acad::ErrorStatus SetLayerExpression(
    const ACHAR* pLayerExp
);
\end{verbatim}

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pLayerExp</td>
<td>Input layer expression.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of \texttt{Doc-O-Matic}. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at \texttt{support@toolsfactory.com}.
Sets the location expression.

`Acad::ErrorStatus SetLocationExpression(
    const ACHAR* pLocationExp
);`

Parameters  Description

pLocationExp  Input location expression.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the rotation expression.

```cpp
Acad::ErrorStatus SetRotationExpression(
    const ACHAR* pRotationExp
);
```

Parameters

<table>
<thead>
<tr>
<th>pRotationExp</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input rotation expression.</td>
<td></td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the text style by name.

Acad::ErrorStatus SetStyle(
    const ACHAR* pStyleName // Parameters Description
);

pStyleName // Input style name.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the style expression.

```cpp
Acad::ErrorStatus SetStyleExpression(
    const ACHAR* pStyleExp
);
```

**Parameters**

<table>
<thead>
<tr>
<th>pStyleExp</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Input style expression.</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the text string expression.

`Acad::ErrorStatus SetTextStringExpression(
    const ACHAR* pTextExp
);`

Parameters | Description
--- | ---
pTextExp | Input string expression.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Builds topology stylization information for later regeneration.

```cpp
virtual Acad::ErrorStatus Update(
    void* & pCookie,
    const ACHAR* pTopoName,
    Adesk::IntDbId lTopoId,
    Adesk::UInt32 flag = 0
);
```

Parameters

- **pCookie**
  - Input: opaque memory value for later access by Dismiss().

- **pTopoName**
  - Input: topology name.

- **lTopoId**
  - Input: topology element ID.

- **flag**
  - Reserved. For internal use only.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Remarks

Derived classes can perform any one-time Update()-time actions here. At Dismiss() time, the implementation can do any required work to clean up storage created at Update() time.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Base class for display-management styles.

class AcMapDMStyle : public AcDbObject;

File

DmDisplayStyle.h

Remarks

This class includes utility functions that stylize, unstylize, cache style values, preview, and so on.

Enumerations

- **CachableProperties**: Enumerates ADE non-geometric DOT variables.
- **CoExistenceFlags**: Enumerates CoExistance preferences that Styles may declare.

Methods

- **Apply**: Applies this style to a specified entity.
- **CacheStylizedPropValue**: Caches the stylized string property value.
- **CacheStylizedPropValue**: Caches the stylized color property value.
- **CacheStylizedPropValue**: Caches the stylized integer property value.
- **CacheStylizedPropValue**: Caches the stylized topology string property value.
- **CacheStylizedPropValue**: Caches the stylized topology color property value.
- **CacheStylizedPropValue**: Caches the stylized topology integer property value.
- **CoExistenceType**: Styles may declare preferences for how they will co-exist with other styles under a single Element.
- **copyFrom**: Copies the contents of an object into the messaged object, if feasible. See also copyFrom() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly. Decrements the reference count to indicate there is
DecrementRef

one fewer client using this style. If the reference count becomes zero, the style erases itself. You do not need to call this function directly unless you are implementing an AcMapDMStyleReference class. See also IncrementRef().

deepClone

Clones this style. See also deepClone() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

DeleteCookie

Deletes the style's cookie during database destruction.

Dismiss

Removes the stylization information that is used during regeneration.

Dismiss

Removes the stylization information that is used during regeneration.

DisplayName

Gets an alternative, constant name for this type of style in certain situations where the instance name may be unavailable.

dwgInFields

Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

dwgOutFields

Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

dxfInFields

Lets this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

dxfOutFields

Lets this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

Enable

Enables or disables the style for the specified
Enable
Enables or disables the style for the specified entity.

GetName
Retrieves the name of this style. See also SetName().

GetStylizationEntities
Retrieves the AcDbObjectId of any stylization entities created during an Update().

Implementation
Returns the implementation object.

IncrementRef
Increments the reference count to indicate that there is one more client using this style. You do not need to call this function directly unless you are implementing an AcMapDMStyleReference class. The caller must call DecrementRef() when finished with this style.

IsMultiplyReferenced
Determines whether this style has multiple references.

OnMapProjectInitialized
Invoked when the an AutoCAD Map project is initialized.

OnObjectAppended
Invoked after a style has been appended to the database, allowing the appended style to do any necessary post-append initialization.

Preview
Stylizes an entity.

Project
Retrieves the AutoCAD Map project.

Refresh
Refreshes stylization information for later regeneration.

Refresh
Refreshes topology stylization information for later regeneration.

SetName
Sets the name of this style. See also GetName().

Invoked from within close() before the close actually occurs. The default implementation of this function returns Acad::eOk. See also subClose() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

subClose
Invoked from within erase() before the erase actually occurs. See also subErase() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.
<table>
<thead>
<tr>
<th><strong>UnApply</strong></th>
<th>Removes the style from a specified entity.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Update</strong></td>
<td>Builds topology stylization information for later regeneration.</td>
</tr>
<tr>
<td><strong>Update</strong></td>
<td>Builds stylization information for later regeneration.</td>
</tr>
</tbody>
</table>

Created with a commercial version of *Doc-O-Matic*. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at *support@toolsfactory.com*. |
Enumerates CoExistence preferences that Styles may declare.

```cpp
text
enum CoExistenceFlags {
    kNotApplicable = 0,
    kAllStyles,
    kSingleton,
    kLikeKind
};
```

### Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>kNotApplicable</td>
<td>A default/unitialized value.</td>
</tr>
<tr>
<td>kAllStyles</td>
<td>This style will exist under an Element with all other style types.</td>
</tr>
<tr>
<td>kSingleton</td>
<td>This style must be the only style under an Element.</td>
</tr>
<tr>
<td>kLikeKind</td>
<td>This style can only exist under an Element with other styles of the same kind.</td>
</tr>
</tbody>
</table>

### Remarks

This state is considered by the Element during AddStyle.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Caches the stylized string property value.

```
Acad::ErrorStatus CacheStylizedPropValue(
    AcDbObjectId entityId,
    CachableProperties key,
    const ACHAR* value
) const;
```

**Parameters**
- **entityId**
  - Input ID of the entity.
- **key**
  - Input CachableProperties key.
- **value**
  - Input string value.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Caches the stylized topology string property value.

```cpp
Acad::ErrorStatus CacheStylizedPropValue(
    const ACHAR* pTopoName,
    Adesk::IntDbId lTopoId,
    CachableProperties key,
    const ACHAR* value
) const;
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pTopoName</td>
<td>Input topology name.</td>
</tr>
<tr>
<td>lTopoId</td>
<td>Input topology ID.</td>
</tr>
<tr>
<td>key</td>
<td>Input CachableProperties key.</td>
</tr>
<tr>
<td>value</td>
<td>Input string value.</td>
</tr>
</tbody>
</table>

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Caches the stylized topology color property value.

Acad::ErrorStatus CacheStylizedPropValue(
    const ACHAR* pTopoName,
    Adesk::IntDbId lTopoId,
    CachableProperties key,
    const AcCmColor& color
) const;

Parameters

pTopoName  Input topology name.
lTopoId     Input topology ID.
key         Input CachableProperties key.
color       Input color value.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Caches the stylized topology integer property value.

`Acad::ErrorStatus CacheStylizedPropValue(const ACHAR* pTopoName,
  Adesk::IntDbId lTopoId,
  CachableProperties key,
  int value
) const;`

Parameters Description
---  
`pTopoName` Input topology name.
`lTopoId` Input topology ID.
`key` Input `CachableProperties` key.
`value` Input integer value.

Returns

Returns `Acad::eOk` if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Styles may declare preferences for how they will co-exist with other styles under a single Element.

```cpp
virtual AcMapDMStyle::CoExistenceFlags CoExistenceType() const;
```

Returns Style's CoExistence preferences.

Remarks

They may coexist with all other styles, only other styles of the same type, or may require being the only style under an Element.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Clones this style. See also deepClone() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus deepClone(
    AcDbObject* pOwnerObject,
    AcDbObject*& pClonedObject,
    AcDbIdMapping& idMap,
    Adesk::Boolean isPrimary = true
) const;
```

### Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pOwnerObject</td>
<td>Input object to append the clone to.</td>
</tr>
<tr>
<td>pClonedObject</td>
<td>Returned cloned object, or NULL if not cloned.</td>
</tr>
<tr>
<td>idMap</td>
<td>Input current object map ID.</td>
</tr>
<tr>
<td>isPrimary</td>
<td>Input true if this object is primary, or false if it is owned.</td>
</tr>
</tbody>
</table>

### Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](http://www.toolsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMStyle Class, AcMapDMStyle Class
AcMapDMStyle::Dismiss Method
AcMapDMStyle Class | AcMapDMStyle Class

Removes the stylization information that is used during regeneration.

```cpp
virtual Acad::ErrorStatus Dismiss(
    void* & pCookie,
    const ACHAR* pTopoName,
    Adesk::IntDbId lTopoElemId,
    Adesk::UInt32 flag = 0
);
```

Parameters

| Description | Input memory value used by styles to transfer cached data across time from one stylization method to another. Cookie memory is considered to be opaque for existing styles, and should not be modified by the caller. |
| pCookie | Input topology name. |
| pTopoName | Input topology element ID. |
| ITopoElemId | Reserved. For internal use only. |

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMStyle Class, AcMapDMStyle Class
AcMapDMStyle:: DisplayName Method
AcMapDMStyle Class | AcMapDMStyle Class

Gets an alternative, constant name for this type of style in certain situations where the instance name may be unavailable.

**virtual const** ACHAR* DisplayName() **const**;

Returns

Returns a const ACHAR* alternative name for certain display purposes.

Remarks

Derived classes should return a constant, generally descriptive name.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Enables or disables the style for the specified topology.

```cpp
virtual Acad::ErrorStatus Enable(
    void*& pCookie,
    const ACHAR* pTopoName,
    Adesk::IntDbId lTopoElemId,
    bool bEnable,
    Adesk::UInt32 flag = 0
);
```

### Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pCookie</td>
<td>Input memory value used by styles to transfer cached data across time from one stylization method to another. Cookie memory is considered to be opaque for existing styles, and should not be modified by the caller.</td>
</tr>
<tr>
<td>pTopoName</td>
<td>Input topology name.</td>
</tr>
<tr>
<td>lTopoElemId</td>
<td>Input topology element ID.</td>
</tr>
<tr>
<td>bEnable</td>
<td>Input true to enable the style.</td>
</tr>
<tr>
<td>flag</td>
<td>Reserved. For internal use only.</td>
</tr>
</tbody>
</table>

### Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Stylizes an entity.

```cpp
virtual Acad::ErrorStatus Preview(
    AcDbObjectIdArray& createdEntities,
    AcDbObjectId targetEntId,
    void* internalUse = 0
);
```

**Parameters**
- `createdEntities`: Output array of IDs of any created entities.
- `targetEntId`: Input ID of the entity to be stylized.
- `internalUse`: Reserved. For internal use only.

**Returns**
Returns Acad::eOk if successful; otherwise, returns a different error code.

**Remarks**
If any additional entities are created as a result of stylization, as in the case of text, hatch, and annotation alterations, then `createdEntities` will contain them. The subsequent lifetime of the created entities is managed by the preview system. There is no need to do any caching or other base-class messaging for stylization of preview entities.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Refreshes stylization information for later regeneration.

```cpp
virtual Acad::ErrorStatus Refresh(
    void*& pCookie,
    AcDbObjectId entityId,
    Adesk::UInt32 flag = 0
);
```

**Parameters**

- **pCookie**
  - Input memory value used by styles to transfer cached data across time from one stylization method to another. Cookie memory is considered to be opaque for existing styles, and should not be modified by the caller.

- **entityId**
  - Input object ID.

- **flag**
  - Reserved. For internal use only.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

**Remarks**

Derived classes can reevaluate any Update()-time actions here. An example would be to rebuild a stylization entity in response to changes in the target entity. If the Style populated the cookie during Update, the appropriate cookie will be provided for the target entity. It is acceptable for this method to be a no-op.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Refreshes topology stylization information for later regeneration.

```cpp
virtual Acad::ErrorStatus Refresh(
    void* & pCookie,
    const ACHAR* pTopoName,
    Adesk::IntDbId lTopoElemId,
    Adesk::UInt32 flag = 0
);
```

### Parameters

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pCookie</td>
</tr>
<tr>
<td>pTopoName</td>
</tr>
<tr>
<td>ITopoElemId</td>
</tr>
<tr>
<td>flag</td>
</tr>
</tbody>
</table>

### Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

### Remarks

Derived classes can reevaluate any Update()-time actions here. An example would be to rebuild a stylization entity in response to changes in the target entity. If the Style populated the cookie during Update, the appropriate cookie will be provided for the target topology. It is acceptable for this method to be a no-op.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the name of this style. See also GetName().

```cpp
virtual Acad::ErrorStatus SetName(const ACHAR* pName);
```

**Parameters**

- **pName**

  Input style name.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Builds topology stylization information for later regeneration.

```cpp
virtual Acad::ErrorStatus Update(
    void*& pCookie,
    const ACHAR* pTopoName,
    Adesk::IntDbId lTopoElemId,
    Adesk::UInt32 flag = 0
);
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pCookie</td>
<td>Output memory value used by styles to transfer cached data across time from one stylization method to another. Cookie memory is considered to be opaque for existing styles, and should not be modified by the caller.</td>
</tr>
<tr>
<td>pTopoName</td>
<td>Input topology name.</td>
</tr>
<tr>
<td>lTopoElemId</td>
<td>Input topology element ID.</td>
</tr>
<tr>
<td>flag</td>
<td>Reserved. For internal use only.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Remarks

Derived classes can perform any one-time Update()-time actions here. At Dismiss() time, the implementation can do any required work to clean up storage created at Update() time.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
AcMapDMStyleCategory Class

Represents a category of styles.

class AcMapDMStyleCategory : public AcDbObject;

File

DmStyleCategory.h

Methods

~AcMapDMStyleCategory Destroys an instance of this class.

AcMapDMStyleCategory Constructs an instance of this class.

Append Adds a style to this category.

AppendStyleCopy Copies a style and appends it to this category.

AppendStyleCopy Copies multiple styles and appends them to this category.

audit This function is called by AutoCAD when the AUDIT command is executed.

Let this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

Let this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

Let this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.
dxfoutFields

Let this object write its data. See also dxfoutFields() in the AutoCAD ObjectARX
Developer's Guide. This overloaded function is
called by the system as needed; it is unlikely that
you will need to call it directly.

Erased

Let this object listen to erase-notifications from
items that it owns. See also erased() in the
AutoCAD ObjectARX Developer's Guide. This
overloaded function is called by the system as
needed; it is unlikely that you will need to call it
directly.

Find

Finds a specified style within this category.

GetAt

Retrieves the ID of a style at a specified position.

GetAt

Retrieves a style at a specified position, opened in a
specified mode.

GetAt

Retrieves a style, opened in a specified mode.

GetName

Retrieves the name of this category. See also
SetName().

Has

Determines whether this category has a specified
style.

Implementation

Returns the implementation object.

InsertAt

Inserts a style at a specified position.

InsertStyleCopyAt

Inserts a copy of a style at a specified position.

InsertStyleCopyAt

Inserts multiple styles at a specified position.

Move

Moves a style to a new position.

Move

Moves a style to a new position.

NumStyles

Counts the number of styles within this category.

Remove

Removes a style.

RemoveAt

Removes a style at a specified position.

SetName

Sets the name of this category. See also
SetName().

SubClose

Invoked from within close() before the close
actually occurs. The default implementation of this
function returns Acad::eOk. See also subClose() in
the AutoCAD ObjectARX Developer's Guide. This
overloaded function is called by the system as
subErase

Invoked from within erase() before the erase actually occurs. See also subErase() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the name of this category. See also GetName().

```cpp
virtual Acad::ErrorStatus SetName(
    const ACHAR* pszName
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pszName</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](#). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
AcMapDMStyleLibrary Class

Represents a library of categories of styles.

class AcMapDMStyleLibrary : public AcDbObject;

File
DmStyleLibrary.h

Methods

~AcMapDMStyleLibrary
Destroys an instance of this class.

AcMapDMStyleLibrary
Constructs an instance of this class.

Append
Adds a style category to this library.

audit
This function is called by AutoCAD when the AUDIT command is executed.

let this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide.

This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

let this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

let this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is
called by the system as needed; it is unlikely that you will need to call it directly.

Lets this object listen to erase-notifications from items that it owns. See also erased() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

erased

GetAt
Retrieves a category at a specified position.

GetAt
Retrieves a category at a specified position, opened in a specified mode.

GetAt
Retrieves a category, opened in a specified mode.

Has
Determines whether a specific category exists in this library.

Implementation
Returns the implementation object.

InsertAt
Inserts a category at a specified position.

Move
Moves a category from one position to another in this library.

Move
Moves a category with a specified ID to another position in this library.

NumCategories
Counts the number of categories within this library.

Remove
Removes a category from this library.

RemoveAt
Removes a category at a specified position from this library.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links

Classes
AcMapDMStyleReference Class

Classes

Represents a reference to an existing style.

class AcMapDMStyleReference : public AcDbObject;

File

DmStyleReference.h

Remarks

These references create additional entities in the database, rather than modifying the appearance of existing entities.

Methods

~AcMapDMStyleReference

Destroys an instance of this class.

AcMapDMStyleReference

Constructs an instance of this class.

audit

This function is called by AutoCAD when the AUDIT command is executed.

Let this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

Let this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

DwgInFields

Let this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

dxfInFields

Let this object read its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.
dxfoutFields

Let this object write its data. See also dxfoutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

Implementation

Returns the implementation object.

IsEnabled

Determines whether this reference is enabled. See also SetEnabled().

IsUniqueReference

Determines whether this reference is the only reference to the style that this reference refers to. See also MakeUniqueReference().

MakeUniqueReference

Determines whether this reference is a unique reference and, if not, clones the referenced style and resets the style ID to the new copy. See also IsUniqueReference().

SetEnabled

Enables or disables this reference. See also IsEnabled().

StyleId

Retrieves the ID of the style that this reference refers to.

Invoked from within close() before the close actually occurs. The default implementation of this function returns Acad::eOk. See also subClose() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

Invoked from within erase() before the erase actually occurs. See also subErase() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

Grants control of deep clone operations to the object. In the default implementation, the object is cloned and appended to the owner object pOwnerObject. See also wblockClone() in the AutoCAD ObjectARX Developer's Guide. This
overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links

Classes

AcMapDMStylizationEntityStyle Class

Classes

Represents a style of a stylization entity.

class AcMapDMStylizationEntityStyle : public AcMapDMStyle;

File

DmStyleEntStyle.h

Remarks

A stylization entity is a new entity that stylization generates and adds to the database (rather than making temporary modifications during regeneration). This abstract class is not intended for direct instantiation.

Methods

Apply

Applies the style to an entity.

Inform the display manager that a particular AcDbEntity was created as a result of applying a StylizationEntityStyle.

CacheStylizationEntity

Removes topology stylization information that is used during regeneration.

Dismiss

Removes stylization information that is used during regeneration.

Dismiss

Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

dwgInFields

Lets this object write its data. See
**dwgOutFields**
also `dwgOutFields()` in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

`dwgOutFields()`

**dxfInFields**

Lets this object read its data. See also `dxfInFields()` in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

`dxfInFields()`

**dxfOutFields**

Lets this object write its data. See also `dxfOutFields()` in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

`dxfOutFields()`

**Enable**
Enables or disables the style for the specified topology.

**Enable**
Enables or disables the style for the specified entity.

**ExcludeStylizationEntityFromHatchStyle**
Creates an island around the stylization entity.

**ExcludeStylizationEntityFromHatchStyle**
Creates an island around the stylization entity.

**GetStylizationEntities**
Retrieves the AcDbObjectId of any stylization entities created during an `Update()`.

**Preview**
Applies style traits to yield a preview of the specified entity. Builds stylization information for
UnApply

A derived class must call this function to remove a stylization entity from association with a target topology, for the purposes of AcMapDMSEHatchStyle islanding.

UnExcludeStylizationEntityFromHatchStyle

A derived class must call this function to remove a stylization entity from association with a target entity, for the purposes of AcMapDMSEHatchStyle islanding.

Builds topology stylization information for later regeneration.

Update

Derived classes can perform any one-time Update()-time actions here.

Update

Builds stylization information for later regeneration.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMStylizationEntityStyle Class, AcMapDMStylizationEntityStyle Class
AcMapDMStylizationEntityStyle::Dismiss Method
AcMapDMStylizationEntityStyle Class | AcMapDMStylizationEntityStyle Class

Removes topology stylization information that is used during regeneration.

virtual Acad::ErrorStatus Dismiss(
    void* & pCookie,
    const ACHAR* pTopoName,
    Adesk::IntDbId lTopoElemId,
    Adesk::UInt32 flag = 0
);

Parameters Description
pCookie Input memory value used by styles to transfer cached data across time from one stylization method to another. Cookie memory is considered to be opaque for existing styles, and should not be modified by the caller.
pTopoName Input topology name.
ITopoElemId Input topology element ID.
flag Reserved. For internal use only.

Returns
Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Enables or disables the style for the specified topology.

```cpp
virtual Acad::ErrorStatus Enable(
    void* & pCookie,
    const ACHAR* pTopoName,
    Adesk::IntDbId lTopoElemId,
    bool bEnable,
    Adesk::UInt32 flag = 0
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pCookie</td>
<td>Input memory value used by styles to transfer cached data across time from one stylization method to another. Cookie memory is considered to be opaque for existing styles, and should not be modified by the caller.</td>
</tr>
<tr>
<td>pTopoName</td>
<td>Input topology name.</td>
</tr>
<tr>
<td>lTopoElemId</td>
<td>Input topology element ID.</td>
</tr>
<tr>
<td>bEnable</td>
<td>Input true to enable the style.</td>
</tr>
<tr>
<td>flag</td>
<td>Reserved. For internal use only.</td>
</tr>
</tbody>
</table>

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMStylizationEntityStyle Class, AcMapDMStylizationEntityStyle Class
AcMapDMStylizationEntityStyle::
ExcludeStylizationEntityFromHatchStyle Method
AcMapDMStylizationEntityStyle Class | AcMapDMStylizationEntityStyle Class

Creates an island around the stylization entity.

```
Acad::ErrorStatus ExcludeStylizationEntityFromHatchStyle(
    const ACHAR* pTopoName,
    Adesk::IntDbId lTopoId,
    const AcDbObjectId styleEntId
) const;
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pTopoName</td>
<td>Input topology name.</td>
</tr>
<tr>
<td>lTopoId</td>
<td>Input topology ID.</td>
</tr>
<tr>
<td>styleEntId</td>
<td>Input ID of the stylization entity.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Remarks

A derived class must call this function if it creates a stylization entity, and wants the AcMapDMSEHatchStyle to attempt to create an island around the stylization entity, should a stylization hatch happen to be applied to the same target topology. In this release of AutoCAD Map, islanding works for only AcDbText entities.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Applies style traits to yield a preview of the specified entity.

```cpp
virtual Acad::ErrorStatus Preview(
    AcDbObjectIdArray& createdEntities,
    AcDbObjectId targetEntId,
    void* internalUse
);
```

**Parameters**

- `createdEntities`: Output array of IDs of created alteration entities.
- `targetEntId`: Input ID of the preview entity.
- `internalUse`: Reserved. For internal use only.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

**Remarks**

For stylization entity styles such as text, hatch, and annotation, this function returns an array of created entities.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
A derived class must call this function to remove a stylization entity from association with a target topology, for the purposes of AcMapDMSEHatchStyle islanding.

```cpp
Acad::ErrorStatus UnExcludeStylizationEntityFromHatchStyle(
    const ACHAR* pTopoName,
    Adesk::IntDbId lTopoId,
    const AcDbObjectId styleEntId
) const;
```

**Parameters**

- **pTopoName**: Input topology name.
- **lTopoId**: Input topology ID.
- **styleEntId**: Input ID of the stylization entity.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

**Remarks**

This should be done by the Style at dismiss time if the Style may reuse the same stylization entity against a different target topology. In this release of AutoCAD Map, islanding works for only AcDbText entities.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Builds topology stylization information for later regeneration. Derived classes can perform any one-time Update()-time actions here.

```cpp
virtual Acad::ErrorStatus Update(
    void*& pCookie,
    const ACHAR* pTopoName,
    Adesk::IntDbId lTopoElemId,
    Adesk::UInt32 flag = 0
);
```

### Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pCookie</td>
<td>Output memory value used by styles to transfer cached data across time from one stylization method to another. Cookie memory is considered to be opaque for existing styles, and should not be modified by the caller.</td>
</tr>
<tr>
<td>pTopoName</td>
<td>Input topology name.</td>
</tr>
<tr>
<td>lTopoElemId</td>
<td>Input topology element ID.</td>
</tr>
<tr>
<td>flag</td>
<td>Reserved. For internal use only.</td>
</tr>
</tbody>
</table>

### Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

### Remarks

At Dismiss() time, the implementation can do any required work to clean up storage created at Update() time.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links

Classes

AcMapDMThematicStyle Class

Classes

Represents a thematic stylization.

```cpp
class AcMapDMThematicStyle : public AcMapDMStyle;
```

File

DmThematicStyle.h

Methods

- `~AcMapDMThematicStyle` Destroys an instance of this class.
- `AcMapDMThematicStyle` Constructs an instance of this class.
- `Apply` Applies the style to an entity.
- `AutoRecalculateRanges` Determines whether automatic recalculation of ranges is turned on or off.
- `BuildRangeTables` Builds range tables.
- `CoExistenceType` Styles may declare preferences for how they will co-exist with other styles under a single Element.
- `DataSourceExpression` Retrieves the expression used in theming.
- `DeleteCookie` Deletes the style's cookie during database destruction.
- `Dismiss` Removes topology stylization information that is used during regeneration.
- `Dismiss` Removes stylization information that is used during regeneration.
- `dwgInFields` Lets this object read its data. See also `dwgInFields()` in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.
- `dwgInFields` Lets this object write its data. See also
**dwgOutFields**
dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

Lets this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

**dxfInFields**

**dxfOutFields**

**Enable**
Enables or disables the style for a specified topology.

**Enable**
Enables or disables the style for a specified entity.

**GetStylizationEntities**
Retrieves the IDs of any stylization entities created during an Update().

**GetThematicTable**
Retrieves the thematic table, opened in a specified mode.

**GroupCount**
Counts the number of groups for thematic ranges.

**GroupingAlgorithmName**
Retrieves the name of the grouping algorithm.

**GroupRoundingMethod**
Retrieves the name of the rounding method used.

**IsAlterAnnotationEnabled**
Determines whether annotation alteration is set to modify a thematic stylization query result.

**IsAlterBlockEnabled**
Determines whether block alteration is set to modify a thematic stylization query result.
<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IsAlterColorEnabled</td>
<td>Determines whether color alteration is set to modify a thematic stylization query result.</td>
</tr>
<tr>
<td>IsAlterHatchEnabled</td>
<td>Determines whether hatch alteration is set to modify a thematic stylization query result.</td>
</tr>
<tr>
<td>IsAlterLinestyleEnabled</td>
<td>Determines whether linestyle alteration is set to modify a thematic stylization query result.</td>
</tr>
<tr>
<td>IsAlterLinetypeEnabled</td>
<td>Determines whether linetype alteration is set to modify a thematic stylization query result.</td>
</tr>
<tr>
<td>IsAlterLineweightEnabled</td>
<td>Determines whether lineweight alteration is set to modify a thematic stylization query result.</td>
</tr>
<tr>
<td>IsAlterPlotstyleEnabled</td>
<td>Determines whether plotstyle alteration is set to modify a thematic stylization query result.</td>
</tr>
<tr>
<td>IsAlterTextEnabled</td>
<td>Determines whether text alteration is set to modify a thematic stylization query result.</td>
</tr>
<tr>
<td>IsScaleColorRamp</td>
<td>Determines whether the scale-color-ramp flag is set.</td>
</tr>
<tr>
<td>IsScaleHatchRamp</td>
<td>Determines whether the scale-hatch-ramp flag is set.</td>
</tr>
<tr>
<td>IsScaleLinestyleRamp</td>
<td>Determines whether the scale-linestyle-ramp flag is set.</td>
</tr>
<tr>
<td>IsScaleRampToFit</td>
<td>Determines whether the scale-ramp-to-fit flag is set.</td>
</tr>
<tr>
<td>IsSourceDataNumeric</td>
<td>Determines whether the data source is numeric.</td>
</tr>
<tr>
<td>LastColorRampName</td>
<td>Retrieves the name of the last-used color ramp.</td>
</tr>
<tr>
<td>LastHatchRampName</td>
<td>Retrieves the name of the last-used hatch ramp.</td>
</tr>
<tr>
<td>LastLinestyleRampName</td>
<td>Retrieves the name of the last-used linestyle ramp.</td>
</tr>
</tbody>
</table>
LinetypeScale
   Retrieves the linetype scale.
NormalizationExpression
   Retrieves the normalization expression.
OnObjectAppended
   Invoked when an object is assigned to the database.
Preview
   Stylizes an entity.
RangeTableComparisonOperator
   Retrieves the range-table comparison operator.
SetAlterAnnotation
   Sets annotation alteration.
SetAlterBlock
   Sets block alteration.
SetAlterColor
   Sets color alteration.
SetAlterHatch
   Sets hatch alteration.
SetAlterLinestyle
   Sets linestyle alteration.
SetAlterLinetype
   Sets linetype alteration.
SetAlterLineweight
   Sets lineweight alteration.
SetAlterPlotstyle
   Sets plotstyle alteration.
SetAlterText
   Sets text alteration.
SetAutoRecalculateRanges
   Turns automatic recalculation on or off.
SetDataSourceExpression
   Sets the expression to use in theming.
SetGroupCount
   Sets the number of groups for thematic ranges.
SetGroupingAlgorithmName
   Sets the grouping algorithm.
SetGroupRoundingMethod
   Sets the rounding method to use.
SetLastColorRampName
   Sets the last-used color ramp.
SetLastHatchRampName
   Sets the last-used hatch ramp.
SetLastLinestyleRampName
   Sets the last-used linestyle ramp.
SetLinetypeScale
   Sets the linetype scale.
SetNormalizationExpression
   Sets the normalization expression.
SetRangeTableComparisonOperator
   Sets the range-table comparison operator.
SetScaleColorRamp
   Sets the scale-color-ramp flag.
SetScaleHatchRamp
   Sets the scale-hatch-ramp flag.
SetScaleLinestyleRamp
   Sets the scale-linestyle-ramp flag.
SetScaleRampToFit
   Sets the scale-ramp-to-fit flag.
**SetSourceDataNumeric**  
Sets the data source of the thematic stylization to be numeric.

**SetUseThousandsSeparators**  
Sets the use of thousands separators.

**SetValuesToIgnore**  
Sets the values-to-ignore expression.

**subErase**  
Invoked from within erase() before the erase actually occurs. See also subErase() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

**UnApply**  
Builds stylization information for later regeneration.

**Update**  
Builds topology stylization information for later regeneration.

**UseThousandsSeparators**  
Determines whether thousands separators are used.

**ValuesToIgnore**  
Retrieves the values-to-ignore expression.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Styles may declare preferences for how they will co-exist with other styles under a single Element.

```cpp
virtual AcMapDMStyle::CoExistenceFlags CoExistenceType() const;
```

Returns Style's CoExistance preferences.

Remarks

The ThematicStyle must be the only style under a given Element.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Removes topology stylization information that is used during regeneration.

```cpp
virtual Acad::ErrorStatus Dismiss(
    void* & pCookie,
    const ACHAR* pTopoName,
    Adesk::IntDbId lTopoElemId,
    Adesk::UInt32 flag = 0
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pCookie</td>
<td>Input memory value used by styles to transfer cached data across time from one stylization method to another. Cookie memory is considered to be opaque for existing styles, and should not be modified by the caller.</td>
</tr>
<tr>
<td>pTopoName</td>
<td>Input topology name.</td>
</tr>
<tr>
<td>ITopoElemId</td>
<td>Input topology element ID.</td>
</tr>
<tr>
<td>flag</td>
<td>Reserved. For internal use only.</td>
</tr>
</tbody>
</table>

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Enables or disables the style for a specified topology.

```cpp
virtual Acad::ErrorStatus Enable(
    void*& pCookie,
    const ACHAR* pTopoName,
    Adesk::IntDbId lTopoElemId,
    bool bEnable,
    Adesk::UInt32 flag = 0
);
```

Parameters

- **pCookie**: Input memory value used by styles to transfer cached data across time from one stylization method to another. Cookie memory is considered to be opaque for existing styles, and should not be modified by the caller.

- **pTopoName**: Input topology name.

- **lTopoElemId**: Input topology element ID.

- **bEnable**: Input true to enable the style.

- **flag**: Reserved. For internal use only.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Stylizes an entity.

```cpp
virtual Acad::ErrorStatus Preview(
    AcDbObjectIdArray & createdEntities,
    AcDbObjectId targetEntity,
    void* rangeIndex
);
```

Parameters

- `createdEntities`: Output array of IDs of any created entities.
- `targetEntity`: Input ID of the entity to be stylized.
- `rangeIndex`: Input range index.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Remarks

If any additional entities are created as a result of stylization, as in the case of text, hatch, and annotation alterations, then `createdEntities` will contain them. The subsequent lifetime of the created entities is managed by the preview system. There is no need to do any caching or other base-class messaging for stylization of preview entities.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the expression to use in theming.

```cpp
virtual Acad::ErrorStatus SetDataSourceExpression(
    const ACHAR* pszSourceExpression);
```

Parameters

| pszSourceExpression | Input expression string. |

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the grouping algorithm.

```cpp
virtual Acad::ErrorStatus SetGroupingAlgorithmName(
    const ACHAR* pszGroupingAlgorithmName);
```

**Parameters**

- `pszGroupingAlgorithmName`  
  Input name of the grouping algorithm.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the rounding method to use.

```cpp
virtual Acad::ErrorStatus SetGroupRoundingMethod(
    const ACHAR* pszGroupRoundingMethod);
```

**Parameters**

- `pszGroupRoundingMethod`: Input name of the rounding method.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the last-used color ramp.

```cpp
virtual Acad::ErrorStatus SetLastColorRampName(
    const ACHAR* pszLastColorRampName
);
```

**Parameters**

- `pszLastColorRampName`: Input ramp name.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docOMATIC.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the last-used hatch ramp.

```cpp
virtual Acad::ErrorStatus SetLastHatchRampName(
    const ACHAR* pszLastHatchRampName)
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pszLastHatchRampName</td>
<td>Input ramp name.</td>
</tr>
</tbody>
</table>

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the last-used linestyle ramp.

```cpp
virtual Acad::ErrorStatus SetLastLinestyleRampName(
    const ACHAR* pszLastLinestyleRampName);
```

**Parameters**  
pszLastLinestyleRampName  Input ramp name.

**Returns**  
Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](http://www.doctoramic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the normalization expression.

```cpp
virtual Acad::ErrorStatus SetNormalizationExpression(const ACHAR* pszNormalizationExpression);
```

**Parameters**

- `pszNormalizationExpression`: Input normalization expression.

**Returns**

Returns `Acad::eOk` if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the values-to-ignore expression.

```cpp
virtual Acad::ErrorStatus SetValuesToIgnore(
    const ACHAR* pszValuesToIgnore
);
```

Parameters

<table>
<thead>
<tr>
<th>Description</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pszValuesToIgnore</td>
<td>Input values-to-ignore expression.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Builds topology stylization information for later regeneration.

```cpp
virtual Acad::ErrorStatus Update(
    void*& pCookie,
    const ACHAR* pTopoName,
    Adesk::IntDbId lTopoElemId,
    Adesk::UInt32 flag = 0
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pCookie</td>
<td>Output memory value used by styles to transfer cached data across time from</td>
</tr>
<tr>
<td></td>
<td>one stylization method to another. Cookie memory is considered to be opaque</td>
</tr>
<tr>
<td></td>
<td>for existing styles, and should not be modified by the caller.</td>
</tr>
<tr>
<td>pTopoName</td>
<td>Input topology name.</td>
</tr>
<tr>
<td>lTopoElemId</td>
<td>Input topology element ID.</td>
</tr>
<tr>
<td>flag</td>
<td>Reserved. For internal use only.</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

**Remarks**

Derived classes can perform any one-time Update()-time actions here. At Dismiss() time, the implementation can do any required work to clean up storage created at Update() time.

Created with a commercial version of [Doc-O-Matic](http://www.doc-o-matic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Defines a display-management thematic table and its items.

```cpp
class AcMapDMThematicTable : public AcDbDataTable;
```

File

DmThematicTable.h

Methods

- `~AcMapDMThematicTable` Destroys an instance of this class.
- `AcMapDMThematicTable` Constructs an instance of this class.
- `AppendRow` Appends a specified row to the thematic table.
- `Clear` Empties all contents of the thematic table.
- `dwgInFields` Lets this object read its data. See also `dwgInFields()` in the AutoCAD ObjectARX Developer's Guide.
- `dwgOutFields` Lets this object write its data. See also `dwgOutFields()` in the AutoCAD ObjectARX Developer's Guide.
- `dxfInFields` Lets this object read its data. See also `dxfInFields()` in the AutoCAD ObjectARX Developer's Guide.
- `dxfOutFields` Lets this object write its data. See also `dxfOutFields()` in the AutoCAD ObjectARX Developer's Guide.
- `GetCellAt` Opens and retrieves an annotation cell item.
- `GetCellAt` Opens and retrieves a block cell item.
- `GetCellAt` Opens and retrieves a color cell item.
- `GetCellAt` Opens and retrieves a data cell item.
- `GetCellAt` Opens and retrieves a hatch cell item.
- `GetCellAt` Opens and retrieves a legend-text cell item.
- `GetCellAt` Opens and retrieves a linestyle cell item.
<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetCellAt</td>
<td>Opens and retrieves a linetype cell item.</td>
</tr>
<tr>
<td>GetCellAt</td>
<td>Opens and retrieves a lineweight cell item.</td>
</tr>
<tr>
<td>GetCellAt</td>
<td>Opens and retrieves a plotstyle cell item.</td>
</tr>
<tr>
<td>GetCellAt</td>
<td>Opens and retrieves a text cell item.</td>
</tr>
<tr>
<td>GetRowAt</td>
<td>Opens and retrieves an entire row.</td>
</tr>
<tr>
<td>Implementation</td>
<td>Returns the implementation object.</td>
</tr>
<tr>
<td>SetCellAt</td>
<td>Sets the annotation style for a specified cell.</td>
</tr>
<tr>
<td>SetCellAt</td>
<td>Sets the block for a specified cell.</td>
</tr>
<tr>
<td>SetCellAt</td>
<td>Sets the item color for a specified cell.</td>
</tr>
<tr>
<td>SetCellAt</td>
<td>Sets the value for a specified cell.</td>
</tr>
<tr>
<td>SetCellAt</td>
<td>Sets the hatch for a specified cell.</td>
</tr>
<tr>
<td>SetCellAt</td>
<td>Sets the legend text for a specified cell.</td>
</tr>
<tr>
<td>SetCellAt</td>
<td>Sets the linestyle for a specified cell.</td>
</tr>
<tr>
<td>SetCellAt</td>
<td>Sets the linetype for a specified cell.</td>
</tr>
<tr>
<td>SetCellAt</td>
<td>Sets the lineweight for a specified cell.</td>
</tr>
<tr>
<td>SetCellAt</td>
<td>Sets the plotstyle for a specified cell.</td>
</tr>
<tr>
<td>SetCellAt</td>
<td>Sets the text for a specified cell.</td>
</tr>
<tr>
<td>SetRowAt</td>
<td>Sets the row at a specified position.</td>
</tr>
</tbody>
</table>

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Represented an annotation cell of a thematic definition table.

```plaintext
class AcMapDMThematicTableItemAnnotation : public AcDbObject;
```

File

DmThematicTable.h

- **Methods**

  - `~AcMapDMThematicTableItemAnnotation` Destroys an instance of this class.
  - `AcMapDMThematicTableItemAnnotation` Constructs an instance of this class.
  - `AcMapDMThematicTableItemAnnotation` by using another instance of this class.
  - `ClearColorOverride` Disables the color override.
  - `ClearLayerOverride` Disables the layer override.
  - `ClearLinetypeOverride` Disables the linetype override.
  - `ClearLineweightOverride` Disables the lineweight override.
  - `ClearRotationOverride` Disables the rotation override.
  - `ClearScaleOverride` Disables the scale override.
  - `ColorOverride` Retrieves the color override.
  - `ColorOverrideExpression` Retrieves the color override expression.
  - `dwgInFields` Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide.
  - `dwgOutFields` Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide.
dxfinFields

This function allows the object to read its data. See also dxfinFields() in the AutoCAD ObjectARX Developer's Guide.

dxfoutFields

This function allows the object to write its data. See also dxfoutFields() in the AutoCAD ObjectARX Developer's Guide.

Implementation

Returns the implementation object.

IsColorOverrideEnabled

Indicates the enable state of the color override.

IsLayerOverrideEnabled

Indicates the enable state of the layer override.

IsLinetypeOverrideEnabled

Indicates the enable state of the linetype override.

IsLineweightOverrideEnabled

Indicates the enable state of the lineweight override.

IsRotationOverrideEnabled

Indicates the enable state of the rotation override.

IsScaleOverrideEnabled

Indicates the enable state of the scale override.

LayerOverride

Retrieves the layer override.

LayerOverrideExpression

Retrieves the layer override expression.

LinetypeOverride

Retrieves the linetype override.

LinetypeOverrideExpression

Retrieves the linetype override expression.

LineweightOverride

Retrieves the lineweight override.

LineweightOverrideExpression

Retrieves the lineweight override expression.

LocationOverrideExpression

Retrieves the annotation location (insertion point) override expression.

Read

Lets this object read in its data from an input stream.
RotationOverride
RotationOverrideExpression
ScaleOverride
ScaleOverrideExpression
SetColorOverride
SetColorOverrideExpression
SetLayerOverride
SetLayerOverrideExpression
SetLinetypeOverride
SetLinetypeOverrideExpression
SetLineweightOverride
SetLineweightOverrideExpression
SetLocationOverrideExpression
SetRotationOverride
SetRotationOverrideExpression
SetScaleOverride
SetScaleOverrideExpression
SetTemplateId
TemplateId
Write

Retrieves the rotation override.
Retrieves the annotation rotation override expression.
Retrieves the scale override.
Retrieves the annotation scale override expression.
Sets the color override value.
Sets the annotation Color override expression.
Sets the layer override value.
Sets the annotation Layer override expression.
Sets the linetype override value.
Sets the annotation Linetype override expression.
Sets the lineweight override value.
Sets the annotation Lineweight override expression.
Sets the annotation location (insertion point) override expression.
Sets the rotation override value.
Sets the annotation Rotation override expression.
Sets the scale override value.
Sets the annotation Scale override expression.
Sets the template name for this annotation alteration.
Retrieves the annotation template Id for this alteration.
Writes object data to an output stream.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems
registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableItemAnnotation Class,
AcMapDMThematicTableItemAnnotation Class
AcMapDMThematicTableItemAnnotation:: Read Method
AcMapDMThematicTableItemAnnotation Class |
AcMapDMThematicTableItemAnnotation Class

Lets this object read in its data from an input stream.

```cpp
void Read(
    std::wistream& in
);```

Parameters Description
in Input filer to use to read the object's data.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the annotation Color override expression.

```cpp
virtual Acad::ErrorStatus SetColorOverrideExpression(
    const ACHAR * pszColorOverrideExpression
);
```

**Parameters**

<table>
<thead>
<tr>
<th>pszColorOverride</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input color override expression value.</td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the annotation Layer override expression.

```cpp
virtual Acad::ErrorStatus SetLayerOverrideExpression(
    const ACHAR * pszLayerOverrideExpression);
```

**Parameters**

- `pszLayerOverride`: Input layer override expression value.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the annotation Linetype override expression.

```cpp
virtual Acad::ErrorStatus SetLinetypeOverrideExpression(
    const ACHAR * pszLinetypeOverrideExpression);
```

Parameters Description
pszLinetypeOverride Input linetype override expression value.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the annotation Lineweight override expression.

**virtual** Acad::ErrorStatus SetLineweightOverrideExpression(
    const ACHAR * pszLineweightOverrideExpression
);

Parameters
pszLineweightOverride Input lineweight override expression value.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the annotation location (insertion point) override expression.

```cpp
class AcMapDMThematicTableItemAnnotation {
public:
  virtual Acad::ErrorStatus SetLocationOverrideExpression(
      const ACHAR * pszLocationOverrideExpression) = 0;
};
```

**Parameters**

- `pszLocationOverrideExpression` Input location override expression value.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the annotation Rotation override expression.

```cpp
data virtual Acad::ErrorStatus SetRotationOverrideExpression(
    const ACHAR * pszRotationOverrideExpression
);
```

Parameters Description
pszRotationOverride Input rotation override expression value.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the annotation Scale override expression.

```cpp
virtual Acad::ErrorStatus SetScaleOverrideExpression(
    const ACHAR * pszScaleOverrideExpression
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Description</th>
<th>Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input scale override expression value.</td>
<td>pszScaleOverride</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Writes object data to an output stream.

```cpp
void Write(
    std::wostream& o
) const;
```

**Parameters**

- `o` Input stream filer to use to write in the object's data.

Created with a commercial version of [Doc-O-Matic](https://www.toolsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Represents a block cell of a thematic definition table.

class AcMapDMThematicTableItemBlock : public AcDbObject;

File

DmThematicTable.h

Methods

~AcMapDMThematicTableItemBlock

Destroys an instance of this class.

AcMapDMThematicTableItemBlock

Constructs an instance of this class.

AcMapDMThematicTableItemBlock

Constructs an instance of this class by using another instance of this class.

Angle

Retrieves the angle.

BlockId

Retrieves the block AcDbObjectId.

BlockName

Retrieves the block name.

dwgInFields

Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide.

dwgOutFields

Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide.

dxfInFields

Lets this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide.

dxfOutFields

Lets this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide.

Implementation

Returns the implementation object.

LayerId

Retrieves the layer AcDbObjectId.

LayerName

Retrieves the layer name.
<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scale</td>
<td>Retrieves the scale.</td>
</tr>
<tr>
<td>SetAngle</td>
<td>Sets the angle.</td>
</tr>
<tr>
<td>SetBlockId</td>
<td>Sets the block by AcDbObjectId.</td>
</tr>
<tr>
<td>SetBlockName</td>
<td>Sets the block name.</td>
</tr>
<tr>
<td>SetLayerId</td>
<td>Sets the layer by AcDbObjectId.</td>
</tr>
<tr>
<td>SetLayerName</td>
<td>Sets the layer name.</td>
</tr>
<tr>
<td>SetScale</td>
<td>Sets the scale.</td>
</tr>
</tbody>
</table>

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the block name.

```cpp
virtual Acad::ErrorStatus SetBlockName(const ACHAR* pszBlockName);
```

**Parameters**

- `pszBlockName` - Input block name.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the layer name.

```cpp
virtual Acad::ErrorStatus SetLayerName(
    const ACHAR* pszLayerName
);
```

**Parameters**
- `pszLayerName`: Input layer name.

**Returns**
- Returns `Acad::eOk` if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
Classes
AcMapDMThematicTableItemColor Class
Classes

Represents a color cell of a thematic definition table.

class AcMapDMThematicTableItemColor : public AcDbObject;

File
DmThematicTable.h

Methods

~AcMapDMThematicTableItemColor Destroys an instance of this class.

AcMapDMThematicTableItemColor Constructs an instance of this class.

AcMapDMThematicTableItemColor Constructs an instance of this class by using another instance of this class.

Color Retrieves the color.

dwgInFields Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide.

dwgOutFields Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide.

dxfInFields Lets this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide.

dxfOutFields Lets this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide.

Implementation Returns the implementation object.

SetColor Sets the color.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
Classes
AcMapDMThematicTableItemDataValue Class
Classes

Represents a text/value-pair cell of a thematic definition table.

class AcMapDMThematicTableItemDataValue : public AcDbObject;

File

DmThematicTable.h

Methods

~AcMapDMThematicTableItemDataValue
Destroys an instance of this class.

AcMapDMThematicTableItemDataValue
Constructs an instance of this class.

AcMapDMThematicTableItemDataValue
Constructs an instance of this class by using another instance of this class.

dwgInFields
Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide.

dwgOutFields
Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide.

dxfInFields
Lets this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide.

dxfOutFields
Lets this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide.

Implementation

SetText
Sets the text.

SetValue
Sets the value.

Text
Retrieves the text.
Value

Retrieves the value.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the text.

```cpp
virtual Acad::ErrorStatus SetText(
    const ACHAR* pText
);
```

Parameters Description

- `pText` Input text.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Represented a hatch cell of a thematic definition table.

**class** AcMapDMThematicTableItemHatch : **public** AcDbObject;

**File**

DmThematicTable.h

**Methods**

- `~AcMapDMThematicTableItemHatch` - Destroys an instance of this class.
- `AcMapDMThematicTableItemHatch` - Constructs an instance of this class.
- `AcMapDMThematicTableItemHatch` - Constructs an instance of this class by using another instance of this class.
- `Angle` - Retrieves the angle.
- `Color` - Retrieves the color.
- `dwgInFields` - Lets this object read its data. See also `dwgInFields()` in the AutoCAD ObjectARX Developer's Guide.
- `dwgOutFields` - Lets this object write its data. See also `dwgOutFields()` in the AutoCAD ObjectARX Developer's Guide.
- `dxfInFields` - Lets this object read its data. See also `dxfInFields()` in the AutoCAD ObjectARX Developer's Guide.
- `dxfOutFields` - Lets this object write its data. See also `dxfOutFields()` in the AutoCAD ObjectARX Developer's Guide.
- `Implementation` - Returns the implementation object.
- `LayerId` - Retrieves the layer `AcDbObjectId`.
- `LayerName` - Retrieves the layer name.
- `PatternName` - Retrieves the pattern name.
<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scale</td>
<td>Retrieves the scale.</td>
</tr>
<tr>
<td>SetAngle</td>
<td>Sets the angle.</td>
</tr>
<tr>
<td>SetColor</td>
<td>Sets the color.</td>
</tr>
<tr>
<td>SetLayerId</td>
<td>Sets the layer AcDbObjectId.</td>
</tr>
<tr>
<td>SetLayerName</td>
<td>Sets the layer name.</td>
</tr>
<tr>
<td>SetPatternName</td>
<td>Sets the pattern name.</td>
</tr>
<tr>
<td>SetScale</td>
<td>Sets the scale.</td>
</tr>
</tbody>
</table>

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the layer name.

```cpp
virtual Acad::ErrorStatus SetLayerName(
    const ACHAR* pszLayerName
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pszLayerName</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the pattern name.

```cpp
virtual Acad::ErrorStatus SetPatternName(
    const ACHAR* pszPatternName
);
```

**Parameters**
- **pszPatternName**
  - Input pattern name.

**Returns**
- Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.doc-omatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Class AcMapDMThematicTableItemLegendText

Represents a legend text cell of a thematic definition table.

class AcMapDMThematicTableItemLegendText : public AcDbObject;

File

DmThematicTable.h

Methods

~AcMapDMThematicTableItemLegendText
Destroys an instance of this class.

AcMapDMThematicTableItemLegendText
Constructs an instance of this class.

AcMapDMThematicTableItemLegendText
Constructs an instance of this class by using another instance of this class.

Color
Retrieves the color of the cell.

DwgInFields
Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide.

dwgOutFields
Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide.

dxfInFields
Lets this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide.

dxfOutFields
Lets this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide.

Height
Retrieves the height of the cell.
<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Implementation</td>
<td>Returns the implementation object.</td>
</tr>
<tr>
<td>LayerId</td>
<td>Retrieves the layer AcDbObjectId of the cell.</td>
</tr>
<tr>
<td>LayerName</td>
<td>Retrieves the layer name of the cell.</td>
</tr>
<tr>
<td>SetColor</td>
<td>Sets the color of the cell.</td>
</tr>
<tr>
<td>SetHeight</td>
<td>Sets the height of the cell.</td>
</tr>
<tr>
<td>SetLayerId</td>
<td>Sets the layer AcDbObjectId of the cell.</td>
</tr>
<tr>
<td>SetLayerName</td>
<td>Sets the layer name of the cell.</td>
</tr>
<tr>
<td>SetStyleId</td>
<td>Sets the style AcDbObjectId of the cell.</td>
</tr>
<tr>
<td>SetStyleName</td>
<td>Sets the style name of the cell.</td>
</tr>
<tr>
<td>SetText</td>
<td>Sets the text of the cell.</td>
</tr>
<tr>
<td>StyleId</td>
<td>Retrieves the style AcDbObjectId of the cell.</td>
</tr>
<tr>
<td>StyleName</td>
<td>Retrieves the style name of the cell.</td>
</tr>
<tr>
<td>Text</td>
<td>Retrieves the text of the cell.</td>
</tr>
</tbody>
</table>

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the layer name of the cell.

```cpp
virtual Acad::ErrorStatus SetLayerName(const ACHAR* pszLayerName);
```

**Parameters**
- `pszLayerName` Input layer name.

**Returns**
Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the style name of the cell.

```cpp
virtual Acad::ErrorStatus SetStyleName(
    const ACHAR* pszStyleName);
```

**Parameters**
- `pszStyleName`: Input style name.

**Returns**
- Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the text of the cell.

```cpp
virtual Acad::ErrorStatus SetText(
    const ACHAR* pszText
);
```

Parameters

- `pszText`: Input text.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Represented a linestyle cell of a thematic definition table.

class AcMapDMThematicTableItemLinestyle : public AcDbObject;

File

DmThematicTable.h

Methods

~AcMapDMThematicTableItemLinestyle

Destroys an instance of this class.

AcMapDMThematicTableItemLinestyle

Constructs an instance of this class.

AcMapDMThematicTableItemLinestyle

Constructs an instance of this class by using another instance of this class.

dwgInFields

Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide.

dwgOutFields

Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide.

dxfInFields

Lets this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide.

dxfOutFields

Lets this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide.

Implementation

Returns the implementation object.

LayerId

Retrieves the layer AcDbObjectId.

LayerName

Retrieves the layer name.

LinetypeId

Retrieves the linetype AcDbObjectId.
<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LinetypeName</td>
<td>Retrieves the linetype name.</td>
</tr>
<tr>
<td>SetLayerId</td>
<td>Sets the layer AcDbObjectId.</td>
</tr>
<tr>
<td>SetLayerName</td>
<td>Sets the layer name.</td>
</tr>
<tr>
<td>SetLinetypeId</td>
<td>Sets the linetype AcDbObjectId.</td>
</tr>
<tr>
<td>SetLinetypeName</td>
<td>Sets the linetype name.</td>
</tr>
<tr>
<td>SetWidth</td>
<td>Sets the width.</td>
</tr>
<tr>
<td>Width</td>
<td>Retrieves the width.</td>
</tr>
</tbody>
</table>

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the layer name.

```cpp
virtual Acad::ErrorStatus SetLayerName(
    const ACHAR* pszLayerName
);

Parameters
pszLayerName Input layer name.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the linetype name.

```cpp
virtual Acad::ErrorStatus SetLinetypeName(
    const ACHAR* pszLinetypeName
);
```

Parameters | Description
--- | ---
pszLinetypeName | Input linetype name.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](http://www.doc-o-matic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links

Classes
AcMapDMThematicTableItemLinetype Class

Classes

Represents a linetype cell of a thematic definition table.

class AcMapDMThematicTableItemLinetype : public AcDbObject;

File

DmThematicTable.h

Methods

~AcMapDMThematicTableItemLinetype

Destroys an instance of this class.

AcMapDMThematicTableItemLinetype

Constructs an instance of this class.

AcMapDMThematicTableItemLinetype

Constructs an instance of this class by using another instance of this class.

dwgInFields

Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide.

dwgOutFields

Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide.

dxfInFields

Lets this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide.

dxfOutFields

Lets this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide.

Implementation

Retrieves the implementation object.

Linetype

Retrieves the linetype.

LinetypeId

Retrieves the linetype AcDbObjectId.

SetLinetype

Sets the linetype name.
SetLinetypeId

Sets the linetype AcDbObjectId.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the linetype name.

```cpp
def virtual Acad::ErrorStatus SetLinetype(
    const ACHAR * pszLinetype
);
```

**Parameters**

- **pszLinetype**
  - Description: Input linetype name.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Class representing lineweight cell of thematic definition table.

```cpp
class AcMapDMThematicTableItemLineweight : public AcDbObject;
```

File `DmThematicTable.h`

- **Methods**
  - `~AcMapDMThematicTableItemLineweight` destroys an instance of this class.
  - `AcMapDMThematicTableItemLineweight` constructs an instance of this class.
  - `AcMapDMThematicTableItemLineweight` constructs an instance of this class by using another instance of this class.
  - `dwgInFields` lets this object read its data. See also `dwgInFields()` in the AutoCAD ObjectARX Developer's Guide.
  - `dwgOutFields` lets this object write its data. See also `dwgOutFields()` in the AutoCAD ObjectARX Developer's Guide.
  - `dxfInFields` lets this object read its data. See also `dxfInFields()` in the AutoCAD ObjectARX Developer's Guide.
  - `dxfOutFields` lets this object write its data. See also `dxfOutFields()` in the AutoCAD ObjectARX Developer's Guide.
  - `Implementation` returns the implementation object.
**Lineweight**
Retrieves the lineweight.

**SetLineweight**
Sets the lineweight.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links

Classes
AcMapDMThematicTableItemPlotstyle Class

Classes

Represents a plotstyle cell of a thematic definition table.

class AcMapDMThematicTableItemPlotstyle : public AcDbObject;

File

DmThematicTable.h

Methods

~AcMapDMThematicTableItemPlotstyle Destroys an instance of this class.

AcMapDMThematicTableItemPlotstyle Constructs an instance of this class.

AcMapDMThematicTableItemPlotstyle Constructs an instance of this class by using another instance of this class.

 lets this object read its data. See also
dwgInFields() in the AutoCAD
ObjectARX Developer's Guide.

dwgOutFields Lets this object write its data. See also
dwgOutFields() in the AutoCAD
ObjectARX Developer's Guide.

 lets this object read its data. See also
dxfInFields() in the AutoCAD
ObjectARX Developer's Guide.

dxfOutFields Lets this object write its data. See also
dxfOutFields() in the AutoCAD
ObjectARX Developer's Guide.

 Implementation Returns the implementation object.
 PlotstyleName Retrieves the plotstyle name.
 SetPlotstyleName Sets the plotstyle name.

Created with a commercial version of Doc-O-Matic. In order to make this
message disappear you need to register this software. If you have problems
registering this software please contact us at support@toolsfactory.com.
Sets the plotstyle name.

```cpp
virtual Acad::ErrorStatus SetPlotstyleName(
    const ACHAR* pszPlotstyleName
);
```

**Parameters**

- `pszPlotstyleName`  
  Input plotstyle name.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Represents a text cell of a thematic definition table.

class AcMapDMThematicTableItemText : public AcDbObject;

File

DmThematicTable.h

Methods

~AcMapDMThematicTableItemText

Destroys an instance of this class.

AcMapDMThematicTableItemText

Constructs an instance of this class.

AcMapDMThematicTableItemText

Constructs an instance of this class by using another instance of this class.

Angle

Retrieves the angle.

Color

Retrieves the color.

dwgInFields

Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide.

dwgOutFields

Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide.

dxfInFields

Lets this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide.

dxfOutFields

Lets this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide.

Height

Retrieves the height.

Implementation

Returns the implementation object.

InsertPoint

Retrieves the insertion point.

Justification

Retrieves the justification.
<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LayerId</td>
<td>Retrieves the layer AcDbObjectId.</td>
</tr>
<tr>
<td>LayerName</td>
<td>Retrieves the layer name.</td>
</tr>
<tr>
<td>SetAngle</td>
<td>Sets the angle.</td>
</tr>
<tr>
<td>SetColor</td>
<td>Sets the color.</td>
</tr>
<tr>
<td>SetHeight</td>
<td>Sets the height.</td>
</tr>
<tr>
<td>SetInsertPoint</td>
<td>Sets the insertion point.</td>
</tr>
<tr>
<td>SetJustification</td>
<td>Sets the justification type.</td>
</tr>
<tr>
<td>SetLayerId</td>
<td>Sets the layer AcDbObjectId.</td>
</tr>
<tr>
<td>SetLayerName</td>
<td>Sets the layer name.</td>
</tr>
<tr>
<td>SetStyleId</td>
<td>Sets the style AcDbObjectId.</td>
</tr>
<tr>
<td>SetStyleName</td>
<td>Sets the style name.</td>
</tr>
<tr>
<td>SetText</td>
<td>Sets the text.</td>
</tr>
<tr>
<td>StyleId</td>
<td>Retrieves the style AcDbObjectId.</td>
</tr>
<tr>
<td>StyleName</td>
<td>Retrieves the style name.</td>
</tr>
<tr>
<td>Text</td>
<td>Retrieves the text.</td>
</tr>
</tbody>
</table>

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the layer name.

```cpp
virtual Acad::ErrorStatus SetLayerName(
    const ACHAR* pszLayerName
);
```

Parameters Description
pszLayerName Input layer name.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the style name.

```cpp
virtual Acad::ErrorStatus SetStyleName(
    const ACHAR* pszStyleName);
```

**Parameters**
- `pszStyleName`: Input style name.

**Returns**
- Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the text.

```cpp
virtual Acad::ErrorStatus SetText(const ACHAR* pszText);
```

**Parameters**
- **pszText**

**Description**
- Input the text.

**Returns**
- Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Utility class that defines a table row and allows easier access to its column items.

```
class AcMapDMThematicTableRow;
```

Data Members

- `mpAnnotation`  Annotation value type.
- `mpBlock`  Block value type.
- `mpColor`  Color value type.
- `mpDataValue`  Text/value-pair value type.
- `mpHatch`  Hatch value type.
- `mpLegendText`  Legend-text value type.
- `mpLinestyle`  Linestyle value type.
- `mpLinetype`  Linetype value type.
- `mpLineweight`  Linewidth value type.
- `mpPlotstyle`  Plotstyle value type.
- `mpText`  Text value type.

Methods

- `~AcMapDMThematicTableRow`  Destroys an instance of this class.
- `AcMapDMThematicTableRow`  Constructs an instance of this class.
- `closeRow`  Closes all members.
- `Init`  Initializes the member variables.

Created with a commercial version of [Doc-O-Matic](https://www.toolsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Defines a query for all topological elements in the current drawing.

**class** `AcMapDMTopoElement` : `public AcMapDMTopoQueryElement`;

### Methods

- **~AcMapDMTopoElement**
  Destroys an instance of this class.

- **AcMapDMTopoElement**
  Constructs an instance of this class.

- **AcquireEntities**
  Runs the topological query against the current drawing and acquires both selected entities and topology objects.

- **ClonesObjectsFromExternalSource**
  Clones objects from the source drawings.

- **DismissEntities**
  Erases queried objects.

- **dwgInFields**
  Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

- **dwgOutFields**
  Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

- **dxfInFields**
  Lets this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it
dxfoutFields

dxfoutFields

Lets this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

EvaluateExpressionValues

EvaluateExpressionValues

Evaluates two expressions against the objects that meet the acquisition criteria.

EvaluateExpressionValues

Evaluates expression(s) against the element selection and returns values for the selected entities.

GetAcquisitionCriteria

GetAcquisitionCriteria

Retrieves a query definition's data-source descriptor.

GetAcquisitionCriteria

Retrieves the query definition's data-source descriptor.

OnMapProjectInitialized

OnMapProjectInitialized

Invoked when an AutoCAD Map project is initialized.

SetAcquisitionCriteria

SetAcquisitionCriteria

Sets the query definition's data-source descriptor.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Evaluates two expressions against the objects that meet the acquisition criteria.

```cpp
virtual Acad::ErrorStatus EvaluateExpressionValues(
    AcArray<AcMapValue*>& arValues1,
    AcArray<AcMapValue*>& arValues2,
    const ACHAR* kpszExpression1,
    AcMap::EDataType kDataType1,
    const ACHAR* kpszExpression2,
    AcMap::EDataType kDataType2)
```

**Parameters**

- `arValues1`: Output array of evaluated values.
- `arValues2`: Output array of evaluated values.
- `kpszExpression1`: Input expression to evaluate.
- `kDataType1`: Input return data type expected.
- `kpszExpression2`: Input expression to evaluate.
- `kDataType2`: Input return data type expected.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

**Remarks**

In the case of query elements, objects are not cloned and the expressions are evaluated against the original objects. If the expressions are not applicable to an entity (the .AREA for a line, for example), the result for this entity has type AcMap::kUnknownType.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Evaluates expression(s) against the element selection and returns values for the selected entities.

```cpp
virtual Acad::ErrorStatus EvaluateExpressionValues(
    AcArray<AcMapValue*>& arValues,
    const ACHAR* kpszExpression,
    Acad::EDataType kDataType
);
```

**Parameters**

- *arValues*: Output array of evaluated values.
- *kpszExpression*: Input expression to evaluate.
- *kDataType*: Input return data type expected.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

**Remarks**

If entities have not yet been acquired, this function acquires them.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves a query definition's data-source descriptor.

\texttt{virtual Acad::ErrorStatus GetAcquisitionCriteria(}
\texttt{  ACHAR*\& pszString}
\texttt{) const;}

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pszString</td>
<td>Output pointer to the data-source descriptor, as a string representation.</td>
</tr>
</tbody>
</table>

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of \texttt{Doc-O-Matic}. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at \texttt{support@toolsfactory.com}.
Links

Classes
AcMapDMTopoElementDataSourceDescriptor Class

Classes

Defines a topology data-source.

class AcMapDMTopoElementDataSourceDescriptor : public AcMapDMDataSource

File

DmTopoElement.h

Methods

~AcMapDMTopoElementDataSourceDescriptor
Destroys an instance of this class.

AcMapDMTopoElementDataSourceDescriptor
Constructs an instance of this class.

GetAcquisitionStatement
Retrieves the query's string representation.

GetTopologyName
Retrieves the topology name.

SetAcquisitionStatement
Sets the query string.

SetTopologyName
Sets the topology name.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the query's string representation.

```cpp
virtual Acad::ErrorStatus GetAcquisitionStatement(
    ACHAR*& pszStatement
) const;
```

**Parameters**

- `pszStatement`: Output query string.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the topology name.

```
virtual Acad::ErrorStatus GetTopologyName(
    const ACHAR*& pszTopologyName
) const;
```

Parameters

- `pszTopologyName` Output topology name.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the query string.

```cpp
virtual Acad::ErrorStatus SetAcquisitionStatement(
    const ACHAR* pszStatement);
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pszStatement</td>
<td>Input query string.</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the topology name.

```cpp
virtual Acad::ErrorStatus SetTopologyName(
    const ACHAR* pszTopologyName
);
```

Parameters

- `pszTopologyName`: Input name of the topology.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Defines a topology data-source.

```class AcMapDMTopoQueryDataSourceDescriptor : public AcMapDMAcquiredDwgsQueryDataSourceDescriptor```

Methods

- ```~AcMapDMTopoQueryDataSourceDescriptor``` Destroys an instance of this class.
- ```AcMapDMTopoQueryDataSourceDescriptor``` Constructs an instance of this class.
- ```GetAcquisitionStatement``` Retrieves the query's string representation.
- ```GetQuery``` Retrieves the query definition.
- ```GetTopologyName``` Retrieves the topology name.
- ```SetAcquisitionStatement``` Sets the query string.
- ```SetQuery``` Sets the query definition.
- ```SetTopologyName``` Sets the topology name.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the query's string representation.

```cpp
virtual Acad::ErrorStatus GetAcquisitionStatement(
    ACHAR*& pszStatement
) const;
```

**Parameters**

- `pszStatement` Output query string.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the topology name.

```cpp
virtual Acad::ErrorStatus GetTopologyName(const ACHAR*& pszTopologyName) const;
```

Parameters

- `pszTopologyName`: Output topology name.

Returns

- Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the query string.

```cpp
virtual Acad::ErrorStatus SetAcquisitionStatement(
    const ACHAR* pszStatement);
```

Parameters | Description
---|---
pszStatement | Input query string.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the topology name.

```cpp
virtual Acad::ErrorStatus SetTopologyName(
    const ACHAR* pszTopologyName
);
```

**Parameters**

- `pszTopologyName`  
  - Description: Input name of the topology.

**Returns**

- Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Defines a query element that acquires topology objects in the source drawing.

```cpp
class AcMapDMTopoQueryElement : public AcMapDMAAttachedDwgsQueryElement
```

File

DmTopoQueryElement.h

Methods

- `~AcMapDMTopoQueryElement` destroys an instance of this class.
- `AcMapDMTopoQueryElement` constructs an instance of this class.
- `AcquireEntities` runs the topological query and acquires both selected entities and topology objects.
- `AddStyle` adds a new style to this element.
- `AddStyle` adds a new style to this element.
- `ClonesObjectsFromExternalSource` clones objects from the source drawings.
- `DismissEntities` erases entities that are part of this query element.
- `DismissStylization` dismisses the current stylization.
- `dwgInFields` lets this object read its data. See also `dxfInFields()` in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.
- `dwgOutFields` lets this object write its data. See also `dxfOutFields()` in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.
dxflnFields Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

dxfoutFields Lets this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

EmptyCookieJar Called during database destruction to trigger the deletion of topology-managed cookies.

EnableStyle Enables or disables a style reference.

EvaluateExpressionValues Evaluates two expressions against the objects that meet the acquisition criteria.

EvaluateExpressionValues Evaluates expression(s) against the element selection and returns values for the selected entities.

GetAcquisitionCriteria Retrieves a query definition's data-source descriptor.

GetAcquisitionCriteria Retrieves the query definition's data-source descriptor.

GetTopologyName Retrieves the topology name.

OnMapProjectInitialized Invoked when an AutoCAD Map project is initialized.

RemoveStyle Removes a style reference from this element.

SetAcquisitionCriteria Sets the query definition's data-source descriptor.

SetTopologyName Sets the name of the topology.

SetVisible Makes the entities that are part of this element visible or invisible.

UpdateStylization Stylizes the current or updated selection at the current scale.

Created with a commercial version of Doc-O-Matic. In order to make this
message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMTopoQueryElement Class, AcMapDMTopoQueryElement Class
AcMapDMTopoQueryElement:: AddStyle Method
AcMapDMTopoQueryElement Class | AcMapDMTopoQueryElement Class

Adds a new style to this element.

virtual Acad::ErrorStatus AddStyle(
    AcDbObjectId& styleRefId,
    AcMapDMStyle* pStyle,
    AcMapDMAllStyleReferencesIterator& Position
);

Parameters

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>styleRefId</td>
</tr>
<tr>
<td>pStyle</td>
</tr>
<tr>
<td>Position</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Remarks

The style object and the owning AcMapDMMap object must be closed for this function to succeed. TopoElement will not accept AcMapDMSEAnnotationStyle.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Add a new style to this element.

**virtual Acad::ErrorStatus AddStyle(**

<table>
<thead>
<tr>
<th>virtual Acad::ErrorStatus AddStyle(</th>
<th>outputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>AcDbObjectId&amp; styleRefId,</td>
<td></td>
</tr>
<tr>
<td>const AcDbObjectId&amp; styleId,</td>
<td></td>
</tr>
<tr>
<td>AcMapDMAllStyleReferencesIterator&amp; Position</td>
<td></td>
</tr>
<tr>
<td>)</td>
<td>returns</td>
</tr>
</tbody>
</table>

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>styleRefId</td>
<td>Output ID of the added style reference.</td>
</tr>
<tr>
<td>styleId</td>
<td>Input ID of the style.</td>
</tr>
<tr>
<td>Position</td>
<td>Input position at which to add the style. See also <strong>AcMapDMAllStyleReferencesIterator</strong></td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

**Remarks**

The style object and the owning AcMapDMMap object must be closed for this function to succeed. TopoElement will not accept **AcMapDMSEAnnotationStyle**.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at **support@toolsfactory.com**.
Evaluates two expressions against the objects that meet the acquisition criteria.

```cpp
virtual Acad::ErrorStatus EvaluateExpressionValues(
    AcArray<AcMapValue*>& arValues1,
    AcArray<AcMapValue*>& arValues2,
    const ACHAR* kpszExpression1,
    AcMap::EDataType kDataType1,
    const ACHAR* kpszExpression2,
    AcMap::EDataType kDataType2
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Description</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Output array of evaluated values.</td>
<td>arValues1</td>
</tr>
<tr>
<td>Output array of evaluated values.</td>
<td>arValues2</td>
</tr>
<tr>
<td>Input expression to evaluate.</td>
<td>kpszExpression1</td>
</tr>
<tr>
<td>Input return data type expected.</td>
<td>kDataType1</td>
</tr>
<tr>
<td>Input expression to evaluate.</td>
<td>kpszExpression2</td>
</tr>
<tr>
<td>Input return data type expected.</td>
<td>kDataType2</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

**Remarks**

In the case of query elements, objects are not cloned and the expressions are evaluated against the original objects. If the expressions are not applicable to an entity (the .AREA for a line, for example), the result for this entity has type AcMap::kUnknownType.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Evaluates expression(s) against the element selection and returns values for the selected entities.

```cpp
virtual Acad::ErrorStatus EvaluateExpressionValues(
    AcArray<AcMapValue*>& arValues,
    const ACHAR* kpszExpression,
    AcMap::EDataType kDataType
);
```

**Parameters**

- *arValues*: Output array of evaluated values.
- *kpszExpression*: Input expression to evaluate.
- *kDataType*: Input return data type expected.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

**Remarks**

If entities have not yet been acquired, this function acquires them.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves a query definition's data-source descriptor.

```cpp
virtual Acad::ErrorStatus GetAcquisitionCriteria(
    ACHAR*& pszString
) const;
```

**Parameters**

<table>
<thead>
<tr>
<th>pszString</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Description</strong></td>
</tr>
<tr>
<td>Output pointer to the data-source descriptor, as a string representation.</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Retrieves the topology name.

```cpp
Acad::ErrorStatus GetTopologyName(
    const ACHAR*& kpszName
) const;
```

**Parameters**

- `kpszName`<br>
  Output topology name.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the name of the topology.

```cpp
Acad::ErrorStatus SetTopologyName(
    const ACHAR* kpszName
);
```

**Parameters**

- `kpszName`: Description
  - Input topology name.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

**Remarks**

Current-drawing topologies are ignored.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
This namespace provides structs and enumerations that the AcMapIE classes use in various import-export contexts.

**Functions**

- **AcMapExporter**
  Retrieves the AutoCAD Map exporter object, as a singleton.

- **AcMapExportFormatIterator**
  Retrieves the AutoCAD Map export format iterator.

- **AcMapImporter**
  Retrieves the AutoCAD Map importer object, as a singleton.

- **AcMapImportFormatIterator**
  Retrieves the AutoCAD Map import format iterator.

**Classes**

- **AcMapIEColumn**
  Provides functions that handle imported columns (also called fields).

- **AcMapIEColumnIterator**
  An iterator over a collection of

- **AcMapIEExporter**
  Provides functions that handle the export process.

  Provides callback functions for export reactors. To implement an export reactor, derive classes from these virtual base classes, overwriting the desired functions. Pass instances of the classes to the exporter (an instance of AcMapIEExporter) and the exporter will call back into your classes at various points in the export process. This behavior allows you to modify the export process at runtime or track the export process externally. Do not explicitly delete any strings returned by any of the functions or their output parameters in this class (but delete all other types of returned objects).

- **AcMapIEExportReactor**
  An iterator over a collection of expression-target pairs.

  Provides functions that handle import and
<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AcMapIEFormat</td>
<td>export file formats.</td>
</tr>
<tr>
<td>AcMapIEFormatIterator</td>
<td>An iterator over a collection of</td>
</tr>
<tr>
<td>AcMapIEImporter</td>
<td>Provides functions that handle the import</td>
</tr>
<tr>
<td>AcMapIEImportReactor</td>
<td>Provides callback functions for import</td>
</tr>
<tr>
<td>AcMapIEInputLayer</td>
<td>Provides functions that handle input layers</td>
</tr>
<tr>
<td>AcMapIEInputLayerIterator</td>
<td>(also called themes).</td>
</tr>
<tr>
<td>AcMapIENameValueIterator</td>
<td>An iterator over a collection of</td>
</tr>
<tr>
<td></td>
<td>An iterator over a collection of name-value</td>
</tr>
<tr>
<td></td>
<td>pairs.</td>
</tr>
</tbody>
</table>
This namespace provides structs and enumerations that the AcMapIE classes use in various import-export contexts. For more information, search for *importing* and *exporting* in AutoCAD Map Help.

- **Enumerations**
  - **ErrCode**
    - Enumerates the import-export error codes.
  - **DataType**
    - This is record AcMapIE::DataType.
  - **GeometryType**
    - Enumerates the geometric types to export.
  - **ImportDataMapping**
    - Enumerates the types of incoming-column data mappings.
  - **ExportClassMapping**
    - Enumerates the type of source data for mapping from source to target Feature classes.
  - **LayerNameType**
    - Enumerates the ways to use layer names in the importer.
  - **LocationOption**
    - Enumerates the types of location window.
  - **PointMappingType**
    - Enumerates the ways to import points.
  - **StorageType**
    - Enumerates the storage types of import and export formats. These enumerated values are returned by functions in the AcMapIEFormatclass and describe the data set type. A value can apply to only import, only export, or to both.

- **Structures**
  - **CExportResults**
    - Returns the results of an export.
  - **CImportResults**
    - Returns the results of an import.
  - **CNewExportResults**
    - Returns the results of an export.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns the results of an export.

```c
struct CExportResults {
    unsigned long m_ulEntitiesExported;
    unsigned long m_ulEntitiesSkippedCouldntTransform;
};
```

### Parameters

<table>
<thead>
<tr>
<th>Member Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>m_ulEntitiesExported</td>
<td>Total number of entities that were exported. This will correspond to the number we thought we were exporting.</td>
</tr>
<tr>
<td>m_ulEntitiesSkippedCouldntTransform</td>
<td>Number of entities that were not exported because they could not be transformed to the target coordinate system.</td>
</tr>
</tbody>
</table>

### Remarks

Use this struct to monitor an export or to summarize an export for an end-user, warning the user about export anomalies if necessary.

#### Data Members

- **m_ulEntitiesExported**: This is `m_ulEntitiesExported`, a member of class `CExportResults`.
- **m_ulEntitiesSkippedCouldntTransform**: This is `m_ulEntitiesSkippedCouldntTransform`, a member of class `CExportResults`.

#### Methods

- **CExportResults**: This is `CExportResults`, a member of class `CExportResults`.
Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns the results of an import.

```c
struct CImportResults {
    unsigned long m_ulEntitiesImported;
    unsigned long m_ulEntitiesSkippedCouldntTransform;
    bool m_bEvaluateHatchFailed;
    unsigned long m_ulEntitiesWithColorCloseToBackground;
    unsigned long m_ulEntitiesClassified;
    unsigned long m_ulEntitiesNotClassifiable;
    unsigned long m_ulOutOfRangeNotFixed;
};
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>m_ulEntitiesImported</td>
<td>Total number of entities that were imported.</td>
</tr>
<tr>
<td>m_ulEntitiesSkippedCouldntTransform</td>
<td>Number of entities that were not imported because they could not be transformed to the target coordinate system.</td>
</tr>
<tr>
<td>m_bEvaluateHatchFailed</td>
<td>Number of imported entities whose hatch patterns appear incorrectly on the drawing. If the scale of the global hatch pattern is incorrect for the drawing, then AutoCAD Map will not draw the hatch pattern and will display empty polygons. This situation will not occur if the global hatch pattern is solid (the AutoCAD Map default).</td>
</tr>
<tr>
<td>m_ulOutOfRangeNotFixed</td>
<td>Number of imported entities whose color is almost indistinguishable</td>
</tr>
</tbody>
</table>
m_ulEntitiesWithColorCloseToBackground

from the background color and therefore appear invisible on the drawing. This situation will not occur if the file MapImport.ini contains the setting "Color=Closest ACI only", which forces AutoCAD Map to adjust colors so that entities are visible.

m_ulEntitiesClassified

Number of imported entities that were tagged with a feature class.

m_ulEntitiesNotClassifiable

Number of imported entities that could not be tagged with a feature class.

m_ulOutOfRangeNotFixed

Number of imported entities with at least one out-of-range property that could not be reset to its default value. Often, a property cannot be reset because a line type is not loaded or a layer does not exist. Note that this value counts the number of entities, not the number of properties; it is possible for an entity to have multiple out-of-range properties that could not be reset.

Remarks

Use this struct to monitor an import or to summarize an import for an end-user, warning the user about import anomalies if necessary.

□ Data Members

m_bEvaluateHatchFailed

This is m_bEvaluateHatchFailed, a member of class CImportResults.

m_ulEntitiesClassified

This is m_ulEntitiesClassified, a member of class CImportResults.

m_ulEntitiesImported

This is m_ulEntitiesImported, a member of class CImportResults.

m_ulEntitiesNotClassifiable

This is m_ulEntitiesNotClassifiable,
m_ulEntitiesNotClassifiable member of class CImportResults.

m_ulEntitiesSkippedCouldn'tTransform This is m_ulEntitiesSkippedCouldn'tTransform member of class CImportResults.

m_ulEntitiesWithColorCloseToBackground This is m_ulEntitiesWithColorCloseToBackground a member of class CImportResults.

m_ulOutOfRangeNotFixed This is m_ulOutOfRangeNotFixed, member of class CImportResults.

Methods

- CImportResults This is CImportResults, a member of class CImportResults.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns the results of an export.

```cpp
struct CNewExportResults : public CExportResults {
    unsigned long m_ulTotalEntitiesExported;
    unsigned long m_ulEntitiesExportedMoreThanOnce;
};
```

File

AcMapIEEnum.h

### Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>m_ulEntitiesExported</td>
<td>Total number of entities that were exported. This will correspond to the number we thought we were exporting.</td>
</tr>
<tr>
<td>m_ulEntitiesSkippedCouldntTransform</td>
<td>Number of entities that were not exported because they could not be transformed to the target coordinate system.</td>
</tr>
<tr>
<td>m_ulTotalEntitiesExported</td>
<td>With multi class export some entities will be exported more than once. This contains the total count.</td>
</tr>
<tr>
<td>m_ulEntitiesExportedMoreThanOnce</td>
<td>With multi class export, some entities will be exported more than once. This contains the count of entities which were exported multiple times.</td>
</tr>
</tbody>
</table>

### Remarks

Use this struct to monitor an export or to summarize an export for an end-user, warning the user about export anomalies if necessary. Using CNewExportResults with Map 2007 will return additional results of total entities and entities exported multiple times. Using CNewExportResults with Map 2006 and below will always return 0 for these parameters.
Data Members

`m_ulEntitiesExportedMoreThanOnce` This is `m_ulEntitiesExportedMoreThanOnce`, a member of class `CNewExportResults`.

`m_ulTotalEntitiesExported` This is `m_ulTotalEntitiesExported`, a member of class `CNewExportResults`.

Methods

`CNewExportResults` This is `CNewExportResults`, a member of class `CNewExportResults`.

Created with a commercial version of `Doc-O-Matic`. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
This is m_ulEntitiesExportedMoreThanOnce, a member of class CNewExportResults.

```c
unsigned long m_ulEntitiesExportedMoreThanOnce;
```

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
This is m_ulTotalEntitiesExported, a member of class CNewExportResults.

```c
unsigned long m_ulTotalEntitiesExported;
```

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
This is CNewExportResults, a member of class CNewExportResults.

CNewExportResults();
Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Enumerates the import-export error codes.

```cpp
enum ErrCode {
    kErr_Invalid = 0,
    kErr_OK,
    kErr_Fail,
    kErr_OutOfMemory,
    kErr_FileError,
    kErr_DialogNotAvailable,
    kErr_TableNotSet,
    kErr_ClassNotSet,
    kErr_BadFieldName,
    kErr_BadTableName,
    kErr_ConflictWithTarget,
    kErr_Conflict,
    kErr_ConflictWithKey,
    kErr_NotInitialized,
    kErr_BadParams,
    kErr_InvalidWindow,
    kErr_InvalidMapping,
    kErr_NoBlocksInDrawing,
    kErr_CannotModifySchema,
    kErr_ClassExists,
    kErr_InvalidDataStore
};
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>kErr_Invalid</td>
<td>An invalid reference or input value.</td>
</tr>
<tr>
<td>kErr_OK</td>
<td>The action completed successfully.</td>
</tr>
<tr>
<td>kErr_Fail</td>
<td>The action failed.</td>
</tr>
<tr>
<td>kErr_OutOfMemory</td>
<td>Insufficient system resources.</td>
</tr>
<tr>
<td>kErr_FileError</td>
<td>A file-related error such as an invalid file or directory name, a missing file, or insufficient file permissions.</td>
</tr>
</tbody>
</table>
kErr_DialogNotAvailable  The specified dialog box is unavailable.

kErr_TableNotSet     No table was set.

kErr_ClassNotSet     No feature class was set.

kErr_BadFieldName    Column does not exist.

kErr_BadTableName    Table does not exist or already exists.

kErr_ConflictWithTarget  Target column already exists.

kErr_Conflict        Two incoming columns are mapped to the same target column.

kErr_ConflictWithKey  Target column has the same name as the unique key column.

kErr_NotInitialized   The importer or exporter is not initialized.

kErr_BadParams       Invalid arguments.

kErr_InvalidWindow    Invalid location window.

kErr_InvalidMapping   Reserved.

kErr_NoBlocksInDrawing  Unable to map points to blocks.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
This is record AcMapIE::DataType.

```cpp
enum DataType {
    kData_String,
    kData.Decimal,
    kData_Float,
    kData_Double,
    kData.Short,
    kData_Long,
    kData_Boolean,
    kData_Date,
    kData_Unknown,
    kData_LongLong,
    kData_Raster,
    kData_Invalid
};
```

File

AcMapIEEnum.h

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Enumerates the geometric types to export.

```cpp
enum GeometryType {
  kGeom_Invalid,
  kGeom_All,
  kGeom_Point,
  kGeom_Line,
  kGeom_Polygon,
  kGeom_Text,
  kGeom_Point_Line,
  kGeom_Point_Polygon,
  kGeom_Line_Polygon
};
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>kGeom_Invalid</td>
<td>Invalid geometric type.</td>
</tr>
<tr>
<td>kGeom_All</td>
<td>Export all geometric types.</td>
</tr>
<tr>
<td>kGeom_Point</td>
<td>Export points (AcDbPoint and AcDbBlockReference).</td>
</tr>
<tr>
<td>kGeom_Line</td>
<td>Export lines (AcDbPolyline, AcDb2dPolyline, AcDbSpline, AcDb3dPolyline, and AcDbArc).</td>
</tr>
<tr>
<td>kGeom_Polygon</td>
<td>Export polygons (AcDbPolygon, AcDbEllipse, AcDbCircle, AcDbPolyline if closed, AcDb2dPolyline if closed, AcDbSpline if closed, and AcDb3dPolyline if closed).</td>
</tr>
<tr>
<td>kGeom_Text</td>
<td>Export text (AcDbText and AcDbMText).</td>
</tr>
</tbody>
</table>

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Enumerates the types of incoming-column data mappings.

```cpp
enum ImportDataMapping {
    kNoImportMapping,
    kNewObjectDataOnly,
    kExistingObjectDataOnly,
    kLinkTemplate,
    kLinkOnly
};
```

**Parameters**

- **kNoImportMapping**: Do not import column.
- **kNewObjectDataOnly**: Import column to a new object-data table.
- **kExistingObjectDataOnly**: Import column to an existing object-data table.
- **kLinkTemplate**: Import column to an SQL table.
- **kLinkOnly**: Import column as a link only. For a column, indicates that the column is a key column. For

For **AcMapIEInputLayer::SetDataMapping()**, indicates that only key columns are mapped.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Enumerates the type of source data for mapping from source to target Feature classes.

```cpp
enum ExportClassMapping {
    kExport_InvalidMapping,
    kExport_SingleClass,
    kExport_ByLayer,
    kExport_ByObjectClass,
    kExport_ByObjectData,
    kExport_ByLinkTemplate
};
```

File

AcMapIEEnum.h

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>kExport_InvalidMapping</td>
<td>Value not set</td>
</tr>
<tr>
<td>kExport_SingleClass</td>
<td>All source data is Exported to a single target Class</td>
</tr>
<tr>
<td>kExport_ByLayer</td>
<td>Source of each Feature Class is an AutoCAD Layer</td>
</tr>
<tr>
<td>kExport_ByObjectClass</td>
<td>Source of each Feature Class is an AutoCAD Map Object Class</td>
</tr>
<tr>
<td>kExport_ByObjectData</td>
<td>Source of each Feature Class is an AutoCAD Map Object Data Table</td>
</tr>
<tr>
<td>kExport_ByLinkTemplate</td>
<td>Source of each Feature Class is an AutoCAD Map Link Template</td>
</tr>
</tbody>
</table>

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Enumerates the ways to use layer names in the importer.

```cpp
enum LayerNameType {
    kLayerName_Direct,
    kLayerName_Indirect
};
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>kLayerName_Direct</td>
<td>Use the specified layer name.</td>
</tr>
<tr>
<td>kLayerName_Indirect</td>
<td>Place the entities from this input layer on the layer specified attribute data from this column.</td>
</tr>
</tbody>
</table>

Created with a commercial version of [Doc-O-Matic](#). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Enumerates the types of location window.

```cpp
define LocationOption {
    kDontUse,
    kUseScreenBoundary,
    kUseLocationWindow
};
```

Parameters

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>kDontUse</td>
</tr>
<tr>
<td>kUseScreenBoundary</td>
</tr>
<tr>
<td>kUseLocationWindow</td>
</tr>
</tbody>
</table>

Created with a commercial version of [Doc-O-Matic](https://www.doctoratic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Enumerates the ways to import points.

```cpp
enum PointMappingType {
    kMapPoint_Invalid,
    kMapPoint_ToPoint,
    kMapPoint_ToNamedBlock,
    kMapPoint_ToBlockFromData,
    kMapPoint_ToTextFromData
};
```

File

AcMapIEEnum.h

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>kMapPoint_Invalid</td>
<td>Invalid procedure.</td>
</tr>
<tr>
<td>kMapPoint_ToPoint</td>
<td>Import the point as a point.</td>
</tr>
<tr>
<td>kMapPoint_ToNamedBlock</td>
<td>Import the point as a named block.</td>
</tr>
<tr>
<td>kMapPoint_ToBlockFromData</td>
<td>Import the point as a block named by the specified column.</td>
</tr>
<tr>
<td>kMapPoint_ToTextFromData</td>
<td>Import the point as text.</td>
</tr>
</tbody>
</table>

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Enumerates the storage types of import and export formats. These enumerated values are returned by functions in the AcMapIEFormat class and describe the data set type. A value can apply to only import, only export, or to both.

```cpp
enum StorageType {
    kStorage_Invalid,
    kStorage_FileAllEntityTypes,
    kStorage_FileMultiSelect,
    kStorage_FileOneEntityType,
    kStorage_FolderWithPrefix,
    kStorage_FolderNoPrefix,
    kStorage_FolderOneEntityType,
    kStorage_FileMultiClassNone,
    kStorage_FileMultiClassLayer,
    kStorage_FileMultiClassObjectClass,
    kStorage_FileMultiClassObjectData,
    kStorage_FileMultiClassLinkTemplate,
    kStorage_FolderMultiClassNone,
    kStorage_FolderMultiClassLayer,
    kStorage_FolderMultiClassObjectClass,
    kStorage_FolderMultiClassObjectData,
    kStorage_FolderMultiClassLinkTemplate,
    kStorage_Database,
    kStorage_DatabaseMultiClassNone,
    kStorage_DatabaseMultiClassLayer,
    kStorage_DatabaseMultiClassObjectClass,
    kStorage_DatabaseMultiClassObjectData,
    kStorage_DatabaseMultiClassLinkTemplate
};
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>kStorage_Invalid</td>
<td>Invalid storage type. For import, the importer Init() expects a single file name. For export, the exporter Init() expects a single file</td>
</tr>
<tr>
<td>kStorage_FileAllEntityTypes</td>
<td></td>
</tr>
</tbody>
</table>
name and the exporter can export all entity types.
For import, the importer Init() expects one or more file names. Does not apply to export.

Does not apply to import. For export, the exporter Init() expects a single file name and the exporter can export only one entity type.

Does not apply to import. For export, the exporter Init() expects a folder name and the exporter can export all entity types.

For import, the importer Init() expects a folder name. For export, the exporter Init() expects a folder name and the exporter can export all entity types.

Does not apply to import. For export, the exporter Init() expects a folder name and the exporter can export only one entity type.

Does not apply to import. For export, the exporter Init() expects a file name and the exporter will export a single class.

Does not apply to import. For export, the exporter Init() expects a file name and the exporter will export classes by layer.

Does not apply to import. For export, the exporter Init() expects a file name and the exporter will export classes by object class.

Does not apply to import. For export, the exporter Init() expects a file name and the exporter will export classes
<table>
<thead>
<tr>
<th>kStorage_FileMultiClassLinkTemplate</th>
<th>by object data. Does not apply to import. For export, the exporter Init() expects a file name and the exporter will export classes by link template.</th>
</tr>
</thead>
<tbody>
<tr>
<td>kStorage_FolderMultiClassNone</td>
<td>Does not apply to import. For export, the exporter Init() expects a folder name and the exporter will export a single class.</td>
</tr>
<tr>
<td>kStorage_FolderMultiClassLayer</td>
<td>Does not apply to import. For export, the exporter Init() expects a folder name and the exporter will export classes by layer.</td>
</tr>
<tr>
<td>kStorage_FolderMultiClassObjectClass</td>
<td>Does not apply to import. For export, the exporter Init() expects a folder name and the exporter will export classes by object class.</td>
</tr>
<tr>
<td>kStorage_FolderMultiClassObjectData</td>
<td>Does not apply to import. For export, the exporter Init() expects a folder name and the exporter will export classes by object data.</td>
</tr>
<tr>
<td>kStorage_FolderMultiClassLinkTemplate</td>
<td>Does not apply to import. For export, the exporter Init() expects a folder name and the exporter will export classes by link template.</td>
</tr>
<tr>
<td>kStorage_Database</td>
<td>Not supported.</td>
</tr>
</tbody>
</table>

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Provides functions that handle imported columns (also called fields). For each imported column, you define a mapping to a target column in AutoCAD Map. Two mappings are possible for each column: a feature class mapping and a data mapping. (The feature class mapping is not set by default for any column.) Prior to using the mapping functions in this class, set the column feature class and data table with `AcMapIEInputLayer::SetFeatureClassName()` and `AcMapIEInputLayer::SetDataMapping()`.

For more information, search for importing in AutoCAD Map Help. Do not explicitly delete any strings returned by any of the functions or their output parameters in this class.

An iterator over a collection of `AcMapIEColumn` instances. Do not explicitly delete any strings returned by any of the functions or their output parameters in this class (but delete all other types of returned objects).

Provides functions that handle the export process. For more information, search for exporting in AutoCAD Map Help. Do not explicitly delete any strings returned by any of the functions or their output parameters in this class (but delete all other types of returned objects).

Provides callback functions for export reactors. To implement an export reactor, derive classes from these virtual base classes, overwriting the desired functions. Pass instances of the classes to the exporter (an instance of `AcMapIEExporter`) and the exporter will call back into your class...
**AcMapIEExportReactor**: at various points in the export process. This behavior allows you to modify the export process at runtime or track the export process externally. Do not explicitly delete any strings returned by any of the functions or their output parameters in this class (but delete all other types of returned objects).

**AcMapIEExpressionTargetIterator**: An iterator over a collection of expression-target pairs. The case-insensitive target component of an expression-target pair is a key value and must be unique within the collection. This class (which is similar to the STL map class) differs from the **AcMapIENameValueIterator** class in that this class has stricter naming rules for the target component of an expression-target pair (see the `Add()` or `Set()` function for naming rules). Do not explicitly delete any strings returned by any of the functions or their output parameters in this class (but delete all other types of returned objects).

**AcMapIEFormat**: Provides functions that handle import and export file formats. For more information, search for `importing`, `import file formats`, `exporting`, and `export file formats` in AutoCAD Map Help. Do not explicitly delete any strings returned by any of the functions or their output parameters in this class (but delete all other types of returned objects).

**AcMapIEFormatIterator**: An iterator over a collection of **AcMapIEFormat** instances. Do not explicitly delete any strings returned by any of the functions or their output parameters in this class (but delete all other types of returned objects).

**AcMapIEImporter**: Provides functions that handle the import process. A default import operation brings in attribute data unless you change this behavior using functions in the **AcMapIEInputLayer** class. For more information, search for `importing` in...
AutoCAD Map Help. Do not explicitly delete any strings returned by any of the functions or their output parameters in this class (but delete all other types of returned objects).

Provides callback functions for import reactors. To implement an import reactor, derive classes from these virtual base classes, overwriting the desired functions. Pass instances of the classes to the importer (an instance of AcMapIEImporter) and the importer will call back into your classes at various points in the import process. This behavior allows you to modify the import process at runtime or track the import process externally. Do not explicitly delete any strings returned by any of the functions or their output parameters in this class (but delete all other types of returned objects).

**AcMapIEImportReactor**

Provides functions that handle input layers (also called themes).

After you call the AcMapIEImporter class's `LoadIPF()`, `InvokeDriverOptionsDialog()`, or `SetDriverOptions()` function on the importer, a new `AcMapIEInputLayerIterator` object is created if the function's `bSchemaHasChanged` output parameter is true and any previously obtained input-layer iterator becomes invalid. Calling `AcMapIEImporter::Init()` will also invalidate the `AcMapIEInputLayerIterator` because AutoCAD Map assumes you are opening a new dataset or changing format.

If you set a table name and table type by using `SetDataMapping()`, default mappings are created depending on the existence of an object data table or a link template with that table name. Valid table types, more

**AcMapIEInputLayer**

An iterator over a collection of `AcMapIEInputLayer` instances. Do not explicitly delete any strings returned by any of the...
functions or their output parameters in this class (but delete all other types of returned objects).

An iterator over a collection of name-value pairs. The case-insensitive name component of a name-value pair is a key value and must be unique within the collection. This class (which is similar to the STL map class) differs from the

**AcMapIEExpressionTargetIterator** class in that this class has more lenient naming rules for the name component of a name-value pair. Do not explicitly delete any strings returned by any of the functions or their output parameters in this class (but delete all other types of returned objects).

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Provides functions that handle imported columns (also called fields). For each imported column, you define a mapping to a target column in AutoCAD Map. Two mappings are possible for each column: a feature class mapping and a data mapping. (The feature class mapping is not set by default for any column.) Prior to using the mapping functions in this class, set the column's feature class and data table with `AcMapIEInputLayer::SetFeatureClassName()` and `AcMapIEInputLayer::SetDataMapping()`.

For more information, search for importing in AutoCAD Map Help. Do not explicitly delete any strings returned by any of the functions or their output parameters in this class (but delete all other types of returned objects).

class AcMapIEColumn;
File

AcMapIEColumn.h

Methods

Destroys an instance of this class.

Returns the name of the feature class property that is mapped to this column.

Returns the name of the column in the AutoCAD Map table that this incoming column is mapped to.

Returns the (fixed) name of this incoming column.

Sets the feature class mapping of this incoming column.

Sets the name of the column in the AutoCAD Map table that this incoming column is mapped to. The mapped column name is set by default when you call `AcMapIEInputLayer::SetDataMapping()` or reset every time you call `SetColumnDataMapping()` with different data. The default behavior maps the incoming column name to an existing column in an
existing object data table, or an existing link or link template, where the case-insensitive names match. If no such target column exists then the incoming column is not mapped by default. When you create a new object data table as the default mapping, curly brace and... more

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the feature class mapping of this incoming column.

```cpp
virtual AcMapIE::ErrCode SetColumnClassMapping(
    const ACHAR* pszMappedPropertyName
) = 0;
```

### Parameters

<table>
<thead>
<tr>
<th>Description</th>
<th>pszMappedPropertyName</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input name of the feature class property to map to this column. To reset the column to no mapping, use NULL or an empty string.</td>
<td></td>
</tr>
</tbody>
</table>

### Returns

Returns AcMapIE::ErrCode kErr_OK if successful. Returns AcMapIE::ErrCode kErr_ClassNotSet if the class name was not first set with `AcMapIEInputLayer::SetFeatureClassName`(). Returns AcMapIE::ErrCode kErr_BadFieldName if the column name does not exist in the specified feature class. Returns AcMapIE::ErrCode kErr_Conflict if a conflict with a previously set column exists.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the name of the column in the AutoCAD Map table that this incoming column is mapped to. The mapped column name is set by default when you call `AcMapIEInputLayer::SetDataMapping()` or reset every time you call SetColumnDataMapping() with different data. The default behavior maps the incoming column name to an existing column in an existing object data table, or an existing link or link template, where the case-insensitive names match. If no such target column exists then the incoming column is not mapped by default. When you create a new object data table as the default mapping, curly brace and dot characters are changed to underscores. Object data table column names must be 31 or fewer characters long. Object data table names must be 25 or fewer characters long.

```cpp
default virtual AcMapIE::ErrCode SetColumnDataMapping(
    const ACHAR* pszMappedFieldName
) = 0;
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pszMappedFieldName</td>
<td>Input column name. To reset the column to no mapping, use NULL or an empty string.</td>
</tr>
</tbody>
</table>

**Returns**

Returns `AcMapIE::ErrCode kErr_OK` if successful. Returns `AcMapIE::ErrCode kErr_TableNotSet` if the table name and table type was not first set with `AcMapIEInputLayer::SetDataMapping()`. Returns `AcMapIE::ErrCode kErr_BadFieldName` if the column name does not exist in the specified table. Returns `AcMapIE::ErrCode kErr_Conflict` if a conflict with a previously set feature class column exists.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
An iterator over a collection of AcMapIEColumn instances. Do not explicitly delete any strings returned by any of the functions or their output parameters in this class (but delete all other types of returned objects).

```cpp
class AcMapIEColumnIterator;
```

File

AcMapIEColumnIterator.h

Methods

- `~AcMapIEColumnIterator` Destroys an instance of this class.
- `Done` Determines whether the iterator has reached the end of the collection.
- `Find` Searches the collection for an element.
- `Get` Retrieves the current element in the iteration.
- `Rewind` Moves to the first element in the iteration.
- `Step` Advances to the next element in the iteration.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Searches the collection for an element.

```
virtual AcMapIEColumn* Find(
    const ACHAR * colName
) = 0;
```

**Parameters**
- **colName**
  - Description: Input name of the column to find.

**Returns**

Returns the named AcMapIEColumn if found; otherwise, returns NULL.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Provides functions that handle the export process. For more information, search for exporting in AutoCAD Map Help. Do not explicitly delete any strings returned by any of the functions or their output parameters in this class (but delete all other types of returned objects).

```cpp
class AcMapIEExporter;
```

File

AcMapIEExporter.h

Methods

- **AddClassMapping**: Creates a per-class mapping when exporting multi-classes.
- **AddReactor**: Adds a reactor to this exporter. You must derive a reactor from the virtual base class `AcMapIEExportReactor`. Reactors are triggered through only the API; an end-user cannot trigger a reactor by using the AutoCAD Map user interface. Reactors are live until they are removed explicitly with `RemoveReactor()` or until AutoCAD Map exits. An exporter is a singleton; all callers add reactors to the same exporter instance. If an exporter has multiple reactors, they are called in the order added.
- **ClosedPolylinesAsPolygons**: Determines whether closed polylines are exported as polygons.
- **CountObjects**: Counts the number of objects to export in the current drawing.
- **DiscretizationAngle**: Retrieves the discretization angle. The discretization angle setting is stored in the file `MapExport.ini`, and labeled `SegmentationDegrees`.  

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>DriverOptions</strong></td>
<td>Retrieves the current driver options. Driver options can be set in the file <em>MapExport.ini</em>, by calling <code>InvokeDriverOptionsDialog()</code> or by loading a profile. Use this function to first retrieve the current driver options before adding driver options with <code>SetDriverOptions()</code>.</td>
</tr>
<tr>
<td><strong>Export</strong></td>
<td>Performs the export.</td>
</tr>
<tr>
<td><strong>ExportAll</strong></td>
<td>Determines whether a selection set is ignored during export.</td>
</tr>
<tr>
<td><strong>ExportDataMappings</strong></td>
<td>Determines whether a selection set is ignored during export.</td>
</tr>
<tr>
<td><strong>ExportFromPolygonTopology</strong></td>
<td>Retrieves the settings for the polygon topology to export.</td>
</tr>
<tr>
<td><strong>FeatureClassFilter</strong></td>
<td>Retrieves the feature classes as a list of comma-separated patterns.</td>
</tr>
<tr>
<td><strong>FormatName</strong></td>
<td>Returns the name of the format set during initialization. The format name is set with <code>Init()</code>.</td>
</tr>
<tr>
<td><strong>Init</strong></td>
<td>Initializes an instance of this class.</td>
</tr>
<tr>
<td><strong>InvokeDriverOptionsDialog</strong></td>
<td>Invokes the driver-specific Driver Options dialog box. Not all drivers have dialog boxes, so this function may finish immediately, with the returned error code indicating the status of the dialog box. Any setting changes that an AutoCAD Map user makes in this dialog box are reflected in subsequent calls to <code>DriverOptions()</code>. You can set driver options programmatically by using <code>SetDriverOptions()</code>. Any driver options in the file <em>MapExport.ini</em> are read and available in the set of driver options returned by <code>DriverOptions()</code>.</td>
</tr>
<tr>
<td><strong>LayerFilter</strong></td>
<td>Retrieves the layers as a list of comma-separated patterns.</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>LayerLevelMapping</strong></td>
<td>Retrieves the settings for mapping layers in the drawing to DGN levels.</td>
</tr>
<tr>
<td><strong>LoadEPF</strong></td>
<td>Loads an export profile (.).</td>
</tr>
<tr>
<td><strong>RemoveReactor</strong></td>
<td>Removes a reactor from this exporter.</td>
</tr>
<tr>
<td><strong>ResetDataTypeForProperty</strong></td>
<td>Sets the Data Type for new or existing properties.</td>
</tr>
<tr>
<td><strong>SaveEPF</strong></td>
<td>Saves an export profile (.).</td>
</tr>
<tr>
<td><strong>SelectionSet</strong></td>
<td>Retrieves the selection set.</td>
</tr>
<tr>
<td><strong>SetClassMappingType</strong></td>
<td>Sets the Create-By type when using Per-Class Mapping for FDO targets.</td>
</tr>
<tr>
<td><strong>SetClosedPolylinesAsPolygons</strong></td>
<td>Changes the setting for exporting closed polylines as polygons.</td>
</tr>
<tr>
<td><strong>SetDiscretizationAngle</strong></td>
<td>Sets the discretization angle. The discretization angle setting is stored in the file MapExport.ini, and labeled SegmentationDegrees.</td>
</tr>
<tr>
<td><strong>SetDriverOptions</strong></td>
<td>Sets the driver options. Typically, you should retrieve the current driver options from DriverOptions() and make changes as needed by using this function.</td>
</tr>
<tr>
<td><strong>SetExportAll</strong></td>
<td>Changes the setting for using a selection set during export.</td>
</tr>
<tr>
<td><strong>SetExportDataMappings</strong></td>
<td>Sets the source-column and output-column data mappings for export.</td>
</tr>
<tr>
<td><strong>SetExportFromPolygonTopology</strong></td>
<td>Changes the settings for the polygon topology to export.</td>
</tr>
<tr>
<td><strong>SetFeatureClassFilter</strong></td>
<td>Sets the feature classes.</td>
</tr>
<tr>
<td><strong>SetGeometryTypeForClass</strong></td>
<td>Sets the Geometry Type for new target class.</td>
</tr>
<tr>
<td><strong>SetIsUniqueKeyProperty</strong></td>
<td>Sets a property to be a Unique/Primary Key.</td>
</tr>
<tr>
<td><strong>SetLayerFilter</strong></td>
<td>Sets the layer patterns.</td>
</tr>
<tr>
<td><strong>SetLayerLevelMapping</strong></td>
<td>Changes the settings for mapping layers in the drawing to DGN levels.</td>
</tr>
<tr>
<td><strong>SetSelectionSet</strong></td>
<td>Sets the ADS (AutoCAD Development System) selection set.</td>
</tr>
<tr>
<td></td>
<td>Sets the storage options. Format-specific</td>
</tr>
</tbody>
</table>
storage options are stored in the file MapExport.ini. This function attempts to make sense of its input values even if those values are inconsistent with the storage type. Format support changes dynamically according to the installed drivers so, because it is impossible to know in advance which drivers are installed, this function cannot always determine which combinations of input values make sense for a particular format; this function's behavior is undefined when using invalid input values. Only a few cases exists where it makes sense to change the storage type value from that returned...

more

Sets the target coordinate system for the exported data.

Changes the settings for creating a unique value for each exported object.

Retrieves the storage options. Format-specific storage options are stored in the file MapExport.ini.

Retrieves the target coordinate system for the exported data.

Determines whether a unique value is created for each exported object.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Creates a per-class mapping when exporting multi-classes.

```cpp
virtual AcMapIE::ErrCode AddClassMapping(
    const AcMapIEExportClassMapping*& mapping
) = 0;
```

**Parameters**

- `mapping` - The Class Mapping

**Description**

Returns

- `kErr_OK` if successful
- `kErr_CannotModifySchema` if the datastore does not have that capability
- `kErr_Conflict` if global mappings have also been set
- `kErr_InvalidMapping` if the source has already been mapped
- `kErr_InvalidDataStore` if the target datastore is not supported (FDO)

**Remarks**

The source type for each mapping is based on the StorageOpts setting: Layers, Object Data, Object Classes, or Link Templates. This mapping can only be used with FDO data stores as the target. If this mapping is set, then any global mappings created by `SetLayerLevelMapping()` and `SetExportDataMappings()` are not used. Any new Schema, Class, or Property names which appear in the mapping, but not in an existing datastore will be appended - if the datastore has an append capability.

Created with a commercial version of [Doc-O-Matic](https://www.toolsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the settings for the polygon topology to export.

```cpp
virtual AcMapIE::ErrCode ExportFromPolygonTopology(
    bool& bGroupComplex,
    const ACHAR*& pszTopologyName
) const = 0;
```

**Parameters**

- **bGroupComplex**
  - Output true if complex polygons are grouped when exporting this topology; otherwise, false.

- **pszTopologyName**
  - Output name of the topology.

**Returns**

Returns AcMapIE::ErrCode kErr_OK if successful. Returns AcMapIE::ErrCode kErr_Fail if the process failed.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Initializes an instance of this class.

```cpp
virtual AcMapIE::ErrCode Init(const ACHAR* pszFormatName,
                               const ACHAR* pszFileName) = 0;
```

**Parameters**

- **pszFormatName**: Input name of an AcMapIEFormat object.
- **pszFileName**: Input fully qualified file or directory name, depending on the format.

**Returns**

- Returns AcMapIE::ErrCode kErr_OK if successful.
- Returns AcMapIE::ErrCode kErr_Invalid if the format is invalid.
- Returns AcMapIE::ErrCode kErr_FileError if the file or directory name is invalid.
- Returns AcMapIE::ErrCode kErr_Fail if the process failed for some other reason.

**Remarks**

Call this function before any other export function (or that function will return AcMapIE::ErrCode kErr_NotInitialized).

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Loads an export profile (.

```cpp
virtual AcMapIE::ErrCode LoadEPF(
    const ACHAR* pszFileName
) = 0;
```

**Parameters**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pszFileName</td>
</tr>
</tbody>
</table>

**Returns**

Returns AcMapIE::ErrCode kErr_OK if successful. Returns AcMapIE::ErrCode kErr_FileError if the file could not be loaded. Returns AcMapIE::ErrCode kErr_Fail if the process failed for some other reason.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the Data Type for new or existing properties.

```cpp
virtual AcMapIE::ErrCode ResetDataTypeForProperty(
    const ACHAR* schemaName,
    const ACHAR* className,
    const ACHAR* propertyName,
    AcMapIE::DataType dataType
) = 0;
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>schemaName</td>
<td>Name of the target schema</td>
</tr>
<tr>
<td>className</td>
<td>Name of the target class</td>
</tr>
<tr>
<td>propertyName</td>
<td>Name of the target property</td>
</tr>
<tr>
<td>dataType</td>
<td>Data Type of the property</td>
</tr>
</tbody>
</table>

Returns

- kErr_OK if successful
- kErr_ClassExists if the property is not a newly appended one
- kErr_InvalidMapping if the property is not mapped to
- kErr_InvalidDataStore if the target datastore is not supported (FDO)

Remarks

Normally the data type is automatically determined based on the source data type. This setting allows the type to be reset to another type so that expressions can be used to convert the source type. For example, a source type of .Color can be mapped to Reals as pipe diameters. This cannot be used on already existing properties.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Saves an export profile (.).

```cpp
virtual AcMapIE::ErrCode SaveEPF(
    const ACHAR* pszFileName
) = 0;
```

**Parameters**

- **pszFileName**
  
  Input fully qualified file name. The filename extension .epf is added automatically if it is omitted.

**Returns**

Returns AcMapIE::ErrCode kErr_OK if successful. Returns AcMapIE::ErrCode kErr_FileError if the file could not be saved. Returns AcMapIE::ErrCode kErr_Fail if the process failed for some other reason.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the Create-By type when using Per-Class Mapping for FDO targets.

```cpp
virtual AcMapIE::ErrCode SetClassMappingType(
    AcMapIE::ExportClassMapping mappingType
) = 0;
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mappingType</td>
<td>Type of source objects the class mapping is using, such as Layer, Object Data Tables.</td>
</tr>
</tbody>
</table>

Remarks

Do not reset this unless using Per-Class Mapping. This must be called before calling AddClassMapping().

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Changes the settings for the polygon topology to export.

```cpp
virtual AcMapIE::ErrCode SetExportFromPolygonTopology(
    bool bGroupComplex,
    const ACHAR* pszTopologyName
) = 0;
```

**Parameters**

- `bGroupComplex` Input true to group complex polygons when exporting; otherwise, false.
- `pszTopologyName` Input name of the topology.

**Returns**

Returns `AcMapIE::ErrCode kErr_OK` if successful. Returns `AcMapIE::ErrCode kErr_Fail` if the process failed.

**Remarks**

This function does not check if the specified topology exists.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the feature classes.

```c
virtual AcMapIE::ErrCode SetFeatureClassFilter(
    const ACHAR* pszClassList
) = 0;
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pszClassList</td>
<td>Input feature class list, as a list of comma-separated patterns.</td>
</tr>
</tbody>
</table>

Returns AcMapIE::ErrCode kErr_OK if successful. Returns AcMapIE::ErrCode kErr_Fail if the process failed.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the Geometry Type for new target class.

```cpp
virtual AcMapIE::ErrCode SetGeometryTypeForClass(
    const ACHAR* schemaName,
    const ACHAR* className,
    AcMapIE::GeometryType geomType
) = 0;
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>schemaName</td>
<td>Name of the target schema</td>
</tr>
<tr>
<td>className</td>
<td>Name of the target class</td>
</tr>
<tr>
<td>geomType</td>
<td>Type of Geometry to set. Currently kGeom_Text is not supported</td>
</tr>
</tbody>
</table>

**Returns**

- kErr_OK if successful
- kErr_ClassExists if the class is not a newly appended one
- kErr_InvalidMapping if the class is not mapped to
- kErr_BadParams if the unsupported kGeom_Text is passed in
- kErr_InvalidDataStore if the target datastore is not supported (FDO)

**Remarks**

This cannot be called on existing classes.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets a property to be a Unique/Primary Key.

```
virtual AcMapIE::ErrCode setIsUniqueKeyProperty(
    const ACHAR* schemaName,
    const ACHAR* className,
    const ACHAR* propertyName
) = 0;
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>schemaName</td>
<td>Name of the target schema</td>
</tr>
<tr>
<td>className</td>
<td>Name of the target class</td>
</tr>
<tr>
<td>propertyName</td>
<td>Name of the target property</td>
</tr>
</tbody>
</table>

Returns

- kErr_OK if successful
- kErr_ClassExists if the property is not a newly appended one
- kErr_InvalidDataStore if the target datastore is not supported (FDO)

Remarks

This can only be called for new properties. The property does not need to have a mapping - and should not, if it is an autogenerated key field. In that case, the new property will be added as an autogenerated Unique Key field.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the layer patterns.

```cpp
virtual AcMapIE::ErrCode SetLayerFilter(
    const ACHAR* pszLayerList
) = 0;
```

**Parameters**
- `pszLayerList`: Input layer list, as a list of comma-separated patterns.

**Returns**
- Returns `AcMapIE::ErrCode kErr_OK` if successful. Returns `AcMapIE::ErrCode kErr_Fail` if the process failed.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the storage options. Format-specific storage options are stored in the file
\textit{MapExport.ini}. This function attempts to make sense of its input values even if
those values are inconsistent with the storage type. Format support changes
dynamically according to the installed drivers so, because it is impossible to
know in advance which drivers are installed, this function cannot always
determine which combinations of input values make sense for a particular
format; this function's behavior is undefined when using invalid input values.
Only a few cases exists where it makes sense to change the storage type value
from that returned by \texttt{StorageOpts()}. The following \texttt{StorageType} storage types
are valid for export: \texttt{AcMapIE::StorageType kStorage_FileAllEntityTypes} \texttt{AcMapIE::StorageType kStorage_FileOneEntityType} \texttt{AcMapIE::StorageType kStorage_FolderWithPrefix} \texttt{AcMapIE::StorageType kStorage_FolderNoPrefix} \texttt{AcMapIE::StorageType kStorage_FolderOneEntityType} If the storage type is
\texttt{kStorage_FolderWithPrefix}, then set a prefix. Use \texttt{kStorage_FolderWithPrefix} to
set up a Shape format as a folder type. The output will consist of up to four
shape files, named \texttt{<prefix>_point.shp}, \texttt{<prefix>_line.shp},
\texttt{<prefix>_polygon.shp}, and \texttt{<prefix>_text.shp}. The other possible storage type
for the Shape format is \texttt{kStorage_FileOneEntityType}. It is possible to change the
storage type between these two values for the Shape format. Among the formats
that AutoCAD Map supports, Shape is the only one that requires a geometry
type and prefix. For the ARCINFO format, corresponding to Coverage, the
storage type can be either \texttt{kStorage_FolderNoPrefix} or
\texttt{kStorage_FolderOneEntityType}. Other formats have only one valid storage type,
which is the type returned by \texttt{StorageOpts()}. If the storage type is
\texttt{kStorage_FileOneEntityType} or \texttt{kStorage_FolderOneEntityType} then the default
geometry type is \texttt{kGeom_Point}. Other storage types will ignore the geometry
type setting. To export different types of entities, set \texttt{geomType} to \texttt{kGeom_Point},
\texttt{kGeom_Line}, \texttt{kGeom_Polygon}, or \texttt{kGeom_Text}. AutoCAD Map will filter out
entities of incorrect types before exporting.

\begin{verbatim}
virtual AcMapIE::ErrCode SetStorageOpts(
   AcMapIE::StorageType storageType,
   AcMapIE::GeometryType geomType,
   const ACHAR* pszPrefix

\end{verbatim}
Parameters | Description  
---|---  
storageType | Input StorageType storage type.  
geomType | Input GeometryType geometry type.  
pszPrefix | Input prefix.  

Returns

Returns AcMapIE::ErrCode kErr_OK if successful. Returns AcMapIE::ErrCode kErr_Fail if the process failed.

Created with a commercial version of [Doc-O-Matic](#). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the target coordinate system for the exported data.

```cpp
virtual AcMapIE::ErrCode SetTargetCoordSys(
    const ACHAR* pszCoordName
) = \0;
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pszCoordName</td>
<td>Input name of the target coordinate system to set.</td>
</tr>
</tbody>
</table>

Returns AcMapIE::ErrCode kErr_OK if successful. Returns AcMapIE::ErrCode kErr_Invalid if the coordinate system name is invalid. Returns AcMapIE::ErrCode kErr_Fail if the process failed for some other reason.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Changes the settings for creating a unique value for each exported object.

```cpp
virtual AcMapIE::ErrCode SetUseUniqueKeyField(
    const ACHAR* pszKeyFieldName
) = 0;
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pszKeyFieldName</td>
<td>Input unique-key column name to set. The column name must be 255 or fewer characters long and can include the following symbols: a-z A-Z 0-9 . { } - _ . (Including accented and multibyte characters.)</td>
</tr>
</tbody>
</table>

Returns

Returns AcMapIE::ErrCode kErr_OK if successful. Returns AcMapIE::ErrCode kErr_BadFieldName if the column name is too long or contains invalid characters. Returns AcMapIE::ErrCode kErr_ConflictWithTarget if the column name already exists as a mapped target column name. Returns AcMapIE::ErrCode kErr_Fail if the process failed for some other reason.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the storage options. Format-specific storage options are stored in the file `MapExport.ini`.

```cpp
virtual AcMapIE::ErrCode StorageOpts(
    AcMapIE::StorageType& storageType,
    AcMapIE::GeometryType& geomType,
    const ACHAR*& pszPrefix
) const = 0;
```

**Parameters**

- `storageType` : Output StorageType storage type.
- `geomType` : Output GeometryType geometry type.
- `pszPrefix` : Output prefix.

**Returns**

Returns AcMapIE::ErrCode kErr_OK if successful. Returns AcMapIE::ErrCode kErr_Fail if the process failed.

Created with a commercial version of [Doc-O-Matic](#). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Determines whether a unique value is created for each exported object.

```cpp
virtual AcMapIE::ErrCode UseUniqueKeyField(
    const ACHAR*& pszKeyFieldName
) const = 0;
```

**Parameters**
- `pszKeyFieldName`: Output name of the unique key column.

**Returns**
- Returns `AcMapIE::ErrCode kErr_OK` if successful. Returns `AcMapIE::ErrCode kErr_FAIL` if the process failed.

**Remarks**

The default key column in the AutoCAD Map user interface is `AcMapKey`, but the key column has no default value in this API.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Provides callback functions for export reactors. To implement an export reactor, derive classes from these virtual base classes, overwriting the desired functions. Pass instances of the classes to the exporter (an instance of AcMapIEExporter) and the exporter will call back into your classes at various points in the export process. This behavior allows you to modify the export process at runtime or track the export process externally. Do not explicitly delete any strings returned by any of the functions or their output parameters in this class (but delete all other types of returned objects).

```cpp
class AcMapIEExportReactor;
```

File

AcMapIEReactor.h

- **Methods**
  - `~AcMapIEExportReactor` Destroys an instance of this class.
  - `AcMapIEExportReactor` Constructs an instance of this class.
  - `RecordError` Invoked if an error occurs during export.
  - `RecordExported` Invoked after an entity is exported.
  - `RecordReadyForExport` Invoked before an entity is exported.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Invoked if an error occurs during export.

```cpp
virtual void RecordError(
    const ACHAR* error,
    AcDbEntity* pEnt
);
```

**Parameters**

- `error` Input error description.
- `pEnt` Input entity.

**Returns**

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.toolsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
An iterator over a collection of expression-target pairs. The case-insensitive target component of an expression-target pair is a key value and must be unique within the collection. This class (which is similar to the STL map class) differs from the \texttt{AcMapIENameValueIterator} class in that this class has stricter naming rules for the target component of an expression-target pair (see the \texttt{Add()} or \texttt{Set()} function for naming rules). Do not explicitly delete any strings returned by any of the functions or their output parameters in this class (but delete all other types of returned objects).

\texttt{class AcMapIEExpressionTargetIterator;}

\texttt{AcMapIEExpressionTargetIterator.h}

- **Methods**
  - \texttt{\~AcMapIEExpressionTargetIterator} Destroys an instance of this class.
  - \texttt{Add} Inserts a new element into the collection after the current element and advances the iterator to the new element.
  - \texttt{Clear} Removes all elements from the collection.
  - \texttt{Done} Determines whether the iterator has reached the end of the collection.
  - \texttt{Find} Searches the collection for an element.
  - \texttt{Get} Retrieves the current element in the iteration.
  - \texttt{Remove} Removes an element from the collection.
  - \texttt{Rewind} Moves to the first element in the iteration.
  - \texttt{Set} Sets the expression of an expression-target pair in the collection.
  - \texttt{Step} Advances to the next element in the
iteration.
Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Inserts a new element into the collection after the current element and advances the iterator to the new element.

```cpp
virtual AcMapIE::ErrCode Add(
    const ACHAR* pszExpression,
    const ACHAR* pszTarget
) = 0;
```

Parameters

<table>
<thead>
<tr>
<th>Description</th>
<th>pszExpression</th>
<th>pszTarget</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input expression of the expression-target pair to add.</td>
<td>Input target of the expression-target pair to add.</td>
<td></td>
</tr>
</tbody>
</table>

Returns

Returns AcMapIE::ErrCode kErr_OK if successful. Returns AcMapIE::ErrCode kErr_BadParams if either input string is NULL or empty. Returns AcMapIE::ErrCode kErr_BadFieldName if the input target is invalid. Returns AcMapIE::ErrCode kErr_Fail if the case-insensitive input target already exists in the collection.

Remarks

The name of the target must be 255 or fewer characters long and can include the following symbols: a-z A-Z 0-9 . { } - _ . The target to which you are exporting may have stricter naming rules, however, so it is safest to restrict target names to alphanumeric, multi-byte, and underscore characters; in particular, avoid these characters: . { }.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Searches the collection for an element.

```cpp
virtual bool Find(
    const ACHAR*& pszExpression,
    const ACHAR * pszTarget
) = 0;
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pszExpression</td>
<td>Output expression corresponding to the input target if the expression-target pair was found. An empty string indicates that the element was not found.</td>
</tr>
<tr>
<td>pszTarget</td>
<td>Input case-insensitive target of the expression-target pair to find.</td>
</tr>
</tbody>
</table>

Returns

Returns true if the element was found; otherwise, returns false. False indicates an invalid output expression.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Retrieves the current element in the iteration.

```cpp
virtual bool Get(
    const ACHAR*& pszExpression,
    const ACHAR*& pszTarget
) = 0;
```

**Parameters**

- `pszExpression`: Output expression of the current expression-target pair.
- `pszTarget`: Output target of the current expression-target pair.

**Description**

Returns true if the current element is valid, or false if the iteration has no more elements to retrieve.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapIEExpressionTargetIterator Class, AcMapIEExpressionTargetIterator Class
AcMapIEExpressionTargetIterator:: Remove Method
AcMapIEExpressionTargetIterator Class | AcMapIEExpressionTargetIterator Class

Removes an element from the collection.

```cpp
virtual bool Remove(
    const ACHAR* pszTarget
) = 0;
```

Parameters

<table>
<thead>
<tr>
<th>pszTarget</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Input case-insensitive target of the expression-target pair to remove.</td>
</tr>
</tbody>
</table>

Returns

Returns true if the element was removed successfully; otherwise, returns false.

Remarks

If the iterator is pointing to the removed element, it advances to the next element if possible.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the expression of an expression-target pair in the collection.

```cpp
declared virtual AcMapIE::ErrCode Set(
    const ACHAR* pszExpression,
    const ACHAR* pszTarget
) = 0;
```

**Parameters**

- `pszExpression`  
  Input new expression for the expression-target pair to set.
- `pszTarget`  
  Input case-insensitive target of the expression-target pair to set.

**Returns**

Returns `AcMapIE::ErrCode kErr_OK` if successful. Returns `AcMapIE::ErrCode kErr_BadParams` if either input string is NULL or empty. Returns `AcMapIE::ErrCode kErr_BadFieldName` if the input target name is invalid. Returns `AcMapIE::ErrCode kErr_Fail` if the case-insensitive input target does not already exist in the collection.

**Remarks**

The name of the target must be 255 or fewer characters long and can include the following symbols: a-z A-Z 0-9 . { } - _ . The target to which you are exporting may have stricter naming rules, however, so it is safest to restrict target names to alphanumeric and underscore characters; in particular, avoid these characters: . { }.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Provides functions that handle import and export file formats. For more information, search for *importing, import file formats, exporting, and export file formats* in AutoCAD Map Help. Do not explicitly delete any strings returned by any of the functions or their output parameters in this class (but delete all other types of returned objects).

```cpp
class AcMapIEFormat;
```

### File

`AcMapIEFormat.h`

#### Methods

- **~AcMapIEFormat**
  - Destroys an instance of this class.

- **Description**
  - Retrieves a brief description of this format.

- **Extension**
  - Retrieves the filename extension that AutoCAD Map expects to be used for this format.

- **FormatName**
  - Returns the unique name which AutoCAD Map uses to refer to this format. This format name is not translated and is used in the AutoCAD Map configuration files `MapImport.ini`, `MapExport.ini`, and `MapForeignFileProperties.ini`.

- **HasDriverOptionsDialog**
  - Determines whether a Driver Options dialog box exists for this format.

- **StorageType**
  - Returns the storage type of this format.

Created with a commercial version of [Doc-O-Matic](https://toolsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
An iterator over a collection of AcMapIEFormat instances. Do not explicitly delete any strings returned by any of the functions or their output parameters in this class (but delete all other types of returned objects).

```cpp
class AcMapIEFormatIterator;
```

File

AcMapIEFormatIterator.h

Methods

- **Done**
  - Determines whether the iterator has reached the end of the collection.
- **Find**
  - Searches the collection for an element.
- **Get**
  - Retrieves the current element in the iteration.
- **Rewind**
  - Moves to the first element in the iteration.
- **Step**
  - Advances to the next element in the iteration.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapIEFormatIterator Class, AcMapIEFormatIterator Class
AcMapIEFormatIterator:: Find Method
AcMapIEFormatIterator Class | AcMapIEFormatIterator Class

Searches the collection for an element.

```cpp
virtual AcMapIEFormat* Find(
    const ACHAR * pszFormatName
) = 0;
```

**Parameters**

<table>
<thead>
<tr>
<th>pszFormatName</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input name of the format to find.</td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

Returns the named AcMapIEFormat if found; otherwise, returns null. The caller is responsible for deleting this object.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Provides functions that handle the import process. A default import operation brings in no attribute data unless you change this behavior by using functions in the AcMapIEInputLayer class. For more information, search for "importing" in AutoCAD Map Help. Do not explicitly delete any strings returned by any of the functions or their output parameters in this class (but delete all other types of returned objects).

```
class AcMapIEImporter;
```

File

AcMapIEImporter.h

**Methods**

**AddReactor**

Adds a reactor to this importer. You must derive a reactor from the virtual base class `AcMapIEImportReactor`. Reactors are triggered through only the API; an end-user cannot trigger a reactor by using the AutoCAD Map user interface. Reactors are live until they are removed explicitly with `RemoveReactor()` or until AutoCAD Map exits. An importer is a singleton; all callers add reactors to the same importer instance. If an importer has multiple reactors, they are called in the order added.

**AuditClassifiedAfterImport**

Determines whether classified properties are audited after import.

Retrieves the current driver options. Driver options can be set in the file `MapImport.ini`, by calling `InvokeDriverOptionsDialog()`, or by
<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>DriverOptions</strong></td>
<td>loading a profile. Use this function to first retrieve the current driver options before adding driver options with SetDriverOptions().</td>
</tr>
<tr>
<td><strong>Import</strong></td>
<td>Performs the import.</td>
</tr>
<tr>
<td><strong>ImportPolygonsAsClosedPolylines</strong></td>
<td>Determines whether polygons are imported as closed polylines or as polygons.</td>
</tr>
<tr>
<td><strong>Init</strong></td>
<td>Initializes an instance of this class. Call this function before any other import function (or that function will return AcMapIE::ErrCode kErr_NotInitialized). You must call Init() again after every call to Import() (with one exception: you can call RemoveReactor() after Import() without first calling Init()). Retrieves an iterator over the input layers. An input layer refers to a single MIF/MID or Shape file, a level or geometry type in a DGN file, or a geometry type in a Coverage or E00 file. A layer consists of a layer name and a schema for attribute data. You are responsible for determining how this data is imported; for more information, see the AcMapIEInputLayer and AcMapIEColumn classes.</td>
</tr>
<tr>
<td><strong>InvokeDriverOptionsDialog</strong></td>
<td>Invokes the driver-specific Driver Options dialog box. Not all drivers have dialog boxes, so this function may finish immediately, with the returned error code indicating the status of the dialog box. Any setting changes that an AutoCAD Map user makes in this dialog box are reflected in subsequent calls to DriverOptions(). You can set driver</td>
</tr>
</tbody>
</table>
options programmatically by using SetDriverOptions(). Any driver options in the file MapImport.ini are read and available in the set of driver options returned by DriverOptions().

LoadIPF

Loads an import profile (. LocationWindowAndOptions

Determines which location window is used to filter incoming entities.

RemoveReactor

Removes a reactor from this importer.

SaveIPF

Saves an import profile (. SetAuditClassifiedAfterImport

Enables or disables classified-property audits after import.

SetDriverOptions

Sets the driver options. Typically, you should retrieve the current driver options from DriverOptions() and make changes as needed by using this function.

SetImportPolygonsAsClosedPolylines

Sets the manner in which polygons are imported: as closed polylines or as polygons.

SetLocationWindowAndOptions

Sets the location window used to filter incoming entities.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Initializes an instance of this class. Call this function before any other import function (or that function will return AcMapIE::ErrCode kErr_NotInitialized). You must call Init() again after every call to Import() (with one exception: you can call RemoveReactor() after Import() without first calling Init()).

```cpp
definition
virtual AcMapIE::ErrCode Init(
    const ACHAR* pszFormatName,
    const ACHAR* pszFileName
) = 0;
```

**Parameters**
- *pszFormatName*: Input name of an AcMapIEFormat object.
- *pszFileName*: Input fully qualified file or directory name, depending on the format.

**Returns**
- Returns AcMapIE::ErrCode kErr_OK if successful. Returns AcMapIE::ErrCode kErr_Invalid if the format is invalid. Returns AcMapIE::ErrCode kErr_FileError if the file or directory name is invalid. Returns AcMapIE::ErrCode kErr_BadParams if either input name is NULL or empty. Returns AcMapIE::ErrCode kErr_Fail if the process failed for some other reason.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Loads an import profile (.).

```cpp
virtual AcMapIE::ErrCode LoadIPF(
    const ACHAR* pszFileName,
    bool& bSchemaHasChanged
) = 0;
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pszFileName</td>
<td>Input fully qualified file name. The filename extension .ipf is added automatically if it is omitted.</td>
</tr>
<tr>
<td>bSchemaHasChanged</td>
<td>Output true if the schema has changed; otherwise, false. If this value is true, then any input-layer iterator that you obtained previously becomes invalid.</td>
</tr>
</tbody>
</table>

Returns

Returns AcMapIE::ErrCode kErr_OK if successful. Returns AcMapIE::ErrCode kErr_FileError if the file could not be loaded. Returns AcMapIE::ErrCode kErr_Fail if the process failed for some other reason.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Saves an import profile.

```cpp
virtual AcMapIE::ErrCode SaveIPF(
    const ACHAR* pszFileName
) = 0;
```

Parameters | Description
--- | ---
pszFileName | Input fully qualified file name. The filename extension .ipf is added automatically if it is omitted.

Returns

Returns AcMapIE::ErrCode kErr_OK if successful. Returns AcMapIE::ErrCode kErr_FileError if the file could not be saved. Returns AcMapIE::ErrCode kErr_Fail if the process failed for some other reason.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Provides callback functions for import reactors. To implement an import reactor, derive classes from these virtual base classes, overwriting the desired functions. Pass instances of the classes to the importer (an instance of AcMapIEImporter) and the importer will call back into your classes at various points in the import process. This behavior allows you to modify the import process at runtime or track the import process externally. Do not explicitly delete any strings returned by any of the functions or their output parameters in this class (but delete all other types of returned objects).

```cpp
class AcMapIEImportReactor;
```

File

AcMapIEReactor.h

- **Methods**
  - `~AcMapIEImportReactor` Destroys an instance of this class.
  - `AcMapIEImportReactor` Constructs an instance of this class.
  - `RecordError` Invoked if an error occurs during import.
  - `RecordImported` Invoked after an entity is imported.
  - `RecordReadyForImport` Invoked before an entity is imported.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Invoked if an error occurs during import.

```cpp
virtual void RecordError(
    const ACHAR* pError
);
```

Parameters | Description
--- | ---
pError    | Input error description.

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Provides functions that handle input layers (also called themes).

After you call the AcMapIEImporter class's LoadIPF(),
InvokeDriverOptionsDialog(), or SetDriverOptions() function on the importer, a new AcMapIEInputLayerIterator object is created if the function's bSchemaHasChanged output parameter is true and any previously obtained input-layer iterator becomes invalid. Calling AcMapIEImporter::Init() will also invalidate the AcMapIEInputLayerIterator because AutoCAD Map assumes you are opening a new dataset or changing format.

If you set a table name and table type by using SetDataMapping(), default mappings are created depending on the existence of an object data table or a link template with that table name. The valid table types, set by the input AcMapIE::ImportDataMapping tableType parameter, are kNewObjectDataOnly, kExistingObjectDataOnly, kLinkTemplate, and kLinkOnly.

If the table type is kNewObjectDataOnly and the table name is an existing object data table, then SetDataMapping() generates an error and makes no changes. If no object data table with the specified name exists, then all incoming columns are mapped by default to valid object data column names. During import, when the first entity arrives in this input layer, the table is created. If no entities exist for this input layer, no object data table is created.

If the table type is kExistingObjectDataOnly and the table name is an existing object data table, then SetDataMapping() generates an error and makes no changes. If the object data table exists, then incoming columns are mapped to object data columns that have the same (case-insensitive) names.

If the table type is kLinkTemplate or kLinkOnly and the table name is not an existing link template, then SetDataMapping() generates an error and makes no changes. If the link template exists, then incoming columns are mapped to link template or link columns that have the same (case-insensitive) names. All link templates contain link columns which are the equivalent of unique key columns. If not all link columns are mapped to incoming columns, the mapping is invalid.
and no data will be imported (and a warning is generated when setting the data mapping). It is your responsibility to ensure that all link columns are mapped. The database that you are importing data into may itself have constraints. If a key-column uniqueness constraint is violated, data may disappear silently.

class AcMapIEInputLayer;
File AcMapIEInputLayer.h

- **Methods**
  - `~AcMapIEInputLayer` Destroys an instance of this class.
  - `ColumnIterator` Retrieves an iterator over a collection of AcMapIEColumninstances.
  - `DataMapping` Retrieves the table type and table name of the data mapping.
  - `FeatureClassName` Retrieves the name of the feature class of this input layer.
  - `ImportFromInputLayerOn` Determines whether this import layer is on (and will be imported).
  - `LayerName` Retrieves the name of the target layer to which entities are imported.
  - `Name` Retrieves the name of this input layer.
  - `OriginalCoordSys` Retrieves AutoCAD Map’s best guess for the coordinate system that the data to be imported uses.
  - `PointToBlockMapping` Determines how points are imported.
  - `SetDataMapping` Sets the table type and table name of the data mapping.
  - `SetFeatureClassName` Sets the name of the feature class of this input layer.
  - `SetImportFromInputLayerOn` Turns this input layer on (to be imported) or off (not to be imported).
  - `SetLayerName` Sets the name of the target layer to which entities are imported.
  - `SetPointToBlockMapping` Sets how points are imported.
  - Sets the coordinate system that incoming data is
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SetTargetCoordSys</strong></td>
<td>Retrieves the coordinate system that incoming data is transformed from.</td>
</tr>
<tr>
<td><strong>SetUseForBlockAttributes</strong></td>
<td>Sets whether block attribute values are brought in from incoming attribute data.</td>
</tr>
<tr>
<td><strong>SetUseUniqueKeyField</strong></td>
<td>Determines whether a unique key value is created for each incoming entity. This setting is ignored unless an existing or new object data table already has been set with SetDataMapping(). The key column has type integer. In the AutoCAD Map user interface, this setting corresponds to the Add Unique Key Field check box in the Attribute Data dialog box.</td>
</tr>
</tbody>
</table>

Created with a commercial version of [Doc-O-Matic](https://www/docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Retrieves the table type and table name of the data mapping.

```cpp
virtual AcMapIE::ErrCode DataMapping(
    AcMapIE::ImportDataMapping& tableType,
    const ACHAR*& pszTableName
) const = 0;
```

**Parameters**

- **tableType**: Output ImportDataMapping type of table used. For a detailed description of this parameter, see the overview of this class.
- **pszTableName**: Output table name (object data or link template, depending on tableType).

**Returns**

Returns AcMapIE::ErrCode kErr_OK if successful. Returns AcMapIE::ErrCode kErr_Fail if the process failed.

**Remarks**

In the AutoCAD Map user interface, this setting corresponds to the Data column in the Import dialog box properties table.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the name of the feature class of this input layer.

```
virtual AcMapIE::ErrCode FeatureClassName(
    const ACHAR*& pszName
) const = 0;
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pszName</td>
<td>Output name of the feature class.</td>
</tr>
</tbody>
</table>

**Returns**

Returns AcMapIE::ErrCode kErr_OK if successful. Returns AcMapIE::ErrCode kErr_Fail if the process failed.

**Remarks**

Use the column class to map individual columns. In the AutoCAD Map user interface, this setting corresponds to the Feature Class column in the Import dialog box properties table.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Retrieves the name of the target layer to which entities are imported.

```cpp
virtual AcMapIE::ErrCode LayerName(
    AcMapIE::LayerNameType& layerNameType,
    const ACHAR*& pszName
) const = 0;
```

Parameters

- `layerNameType` Output LayerNameType value indicating how the layer name is used in the importer. The value kLayerName_Direct means that layer name specified by pszName is used. The value kLayerName_Indirect means that entities from this input layer are placed on the layer specified by the column named pszName.
- `pszName` Output name of the layer or column, depending on layerNameType.

Returns

Returns AcMapIE::ErrCode kErr_OK if successful. Returns AcMapIE::ErrCode kErr_Fail if the process failed.

Remarks

In the AutoCAD Map user interface, this setting corresponds to the Drawing Layer column in the Import dialog box properties table.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
virtual AcMapIE::ErrCode PointToBlockMapping(
    AcMapIE::PointMappingType& howToMap,
    const ACHAR*& pszName
) const = 0;

Parameters

howToMap
Output enumerated value that indicates how points are imported.

pszName
Output column name for the text string or block name, depending on howToMap.

Returns

Returns AcMapIE::ErrCode kErr_OK if successful. Returns AcMapIE::ErrCode kErr_Fail if the process failed.

Remarks

In the AutoCAD Map user interface, this setting corresponds to the Points column in the Import dialog box properties table.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the table type and table name of the data mapping.

```cpp
virtual AcMapIE::ErrCode SetDataMapping(
    const AcMapIE::ImportDataMapping tableType,
    const ACHAR* pszTableName
) = 0;
```

**Parameters**

- `tableType`: Input ImportDataMapping type of table to use. For a detailed description of this parameter, see the overview of this class.
- `pszTableName`: Input table name. This table name can be an existing object data table, a new object data table, or an existing link template (depending on `tableType`).

**Returns**

Returns AcMapIE::ErrCode kErr_OK if successful. Returns AcMapIE::ErrCode kErr_Invalid if `tableType` is kNewObjectDataOnly and `pszTableName` is an existing table; if `tableType` is kExistingObjectDataOnly and `pszTableName` in not an existing table; or if `tableType` is kLinkTemplate and `pszTableName` is not an existing link template. Returns AcMapIE::ErrCode kErr_BadTableName if the new data table name is invalid. Allowed characters are alphanumeric, multibyte, and - _ and maximum length is 25 characters. Returns AcMapIE::ErrCode kErr_Fail if the process failed for some other reason.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the name of the feature class of this input layer.

```cpp
virtual AcMapIE::ErrCode SetFeatureClassName(
    const ACHAR* pszName
) = 0;
```

**Parameters**

| pszName | Input name of the feature class to set. |

**Returns**

Returns AcMapIE::ErrCode kErr_OK if successful. Returns AcMapIE::ErrCode kErr_Invalid if the feature class name is invalid. Returns AcMapIE::ErrCode kErr_Fail if the process failed for some other reason.

**Remarks**

Use the column class to map individual columns. In the AutoCAD Map user interface, this setting corresponds to the Feature Class column in the Import dialog box properties table.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the name of the target layer to which entities are imported.

```cpp
virtual AcMapIE::ErrCode SetLayerName(
    AcMapIE::LayerNameType layerNameType,
    const ACHAR* pszName
) = 0;
```

**Parameters**

- **layerNameType**
  - Description: Input LayerNameType value specifying how to use the layer name in the importer. Set this parameter to kLayerName_Direct to use the layer name specified by pszName. Set this parameter to kLayerName_Indirect to place the entities from this input layer on the layer specified by the column named by pszName.

- **pszName**
  - Description: Input name of a layer or column, depending on layerNameType.

**Returns**

Returns AcMapIE::ErrCode kErr_OK if successful. Returns AcMapIE::ErrCode kErr_BadFieldName if layerNameType is kLayerName_Indirect and the column pszName does not exist. Returns AcMapIE::ErrCode kErr_BadParams if layerNameType is kLayerName_Direct and pszName is empty. Returns AcMapIE::ErrCode kErr_Invalid if layerNameType is kLayerName_Direct and pszName is an invalid layer name. Returns AcMapIE::ErrCode kErr_Fail if the process failed for some other reason.

**Remarks**

In the AutoCAD Map user interface, this setting corresponds to the Drawing Layer column in the Import dialog box properties table.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets how points are imported.

```cpp
virtual AcMapIE::ErrCode SetPointToBlockMapping(
    AcMapIE::PointMappingType howToMap,
    const ACHAR* pszName
) = 0;
```

### Parameters

**howToMap**
- Input enumerated value that determines how points are imported.

**pszName**
- Input column name for the text string or block name, depending on howToMap.

### Returns

Returns `AcMapIE::ErrCode` `kErr_OK` if successful. Returns `AcMapIE::ErrCode` `kErr_NoBlocksInDrawing` if `howToMap` sets a block-mapping type in a drawing with no blocks. Returns `AcMapIE::ErrCode` `kErr_BadFieldName` if `howToMap` is `kMapPoint_ToBlockFromData` or `kMapPoint_ToTextFromData` and the column name is invalid. Returns `AcMapIE::ErrCode` `kErr_Fail` if the process failed for some other reason.

### Remarks

In the AutoCAD Map user interface, this setting corresponds to the Points column in the Import dialog box properties table.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the coordinate system that incoming data is transformed from.

```cpp
virtual AcMapIE::ErrCode SetTargetCoordSys(
    const ACHAR* pszCoordSysName
) = 0;
```

### Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pszCoordSysName</td>
<td>Input name of the coordinate system, as a Mentor abbreviation.</td>
</tr>
</tbody>
</table>

### Returns

Returns AcMapIE::ErrCode kErr_OK if successful. Returns AcMapIE::ErrCode kErr_Fail if the process failed.

### Remarks

In the AutoCAD Map user interface, this setting corresponds to the Coordinate System column in the Import dialog box properties table.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets whether a unique key value is created for each incoming entity. This setting is ignored unless an existing or new object data table already has been set with SetDataMapping(). The key column has type integer. In the AutoCAD Map user interface, this setting corresponds to the Add Unique Key Field check box in the Attribute Data dialog box.

```cpp
virtual AcMapIE::ErrCode SetUseUniqueKeyField( 
    const ACHAR* pszKeyFieldName ) = 0;
```

**Parameters**

<table>
<thead>
<tr>
<th>pszKeyFieldName</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input name for key column. This name must follow the pre-Release 14 AutoCAD symbol-naming convention; that is, it can contain only alphanumeric, underscore, and hyphen characters. Multibyte characters are allowed.</td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

Returns AcMapIE::ErrCode kErr_OK if successful. Returns AcMapIE::ErrCode kErr_BadFieldName if the column name is invalid. Returns AcMapIE::ErrCode kErr_Invalid if a column with the same name already exists as an input column. (This situation is not necessarily a problem, but it could render other import-related error reports and actions confusing.) Returns AcMapIE::ErrCode kErr_ConflictWithTarget if the column name already exists as a mapped target column name. Returns AcMapIE::ErrCode kErr_Fail if the process failed for some other reason.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Detersmines whether a unique value is created for each incoming entity. This setting is ignored unless an existing or new object data table already has been set with SetDataMapping(). The key column has type integer. In the AutoCAD Map user interface, this setting corresponds to the Add Unique Key Field check box in the Attribute Data dialog box.

```cpp
virtual bool UseUniqueKeyField(
    const ACHAR*& pszKeyFieldName
) const = 0;
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pszKeyFieldName</td>
<td>Output name of the key column.</td>
</tr>
</tbody>
</table>

Returns

Returns true if unique values are created; otherwise, returns false. If true, pszKeyFieldName column contains a non-empty string; if false, pszKeyFieldName contains an empty string.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
An iterator over a collection of AcMapIEInputLayer instances. Do not explicitly delete any strings returned by any of the functions or their output parameters in this class (but delete all other types of returned objects).

**class** AcMapIEInputLayerIterator;

**File**

AcMapIEInputLayerIterator.h

**Methods**

- **~AcMapIEInputLayerIterator** Destroys an instance of this class.
- **Done** Determines whether the iterator has reached the end of the collection.
- **Find** Searches the collection for an element.
- **Get** Retrieves the current element in the iteration.
- **Rewind** Moves to the first element in the iteration.
- **Step** Advances to the next element in the iteration.
AcMapIEInputLayerIterator Class, AcMapIEInputLayerIterator Class
AcMapIEInputLayerIterator:: Find Method
AcMapIEInputLayerIterator Class | AcMapIEInputLayerIterator Class

Searches the collection for an element.

```cpp
virtual AcMapIEInputLayer* Find(
    const ACHAR * pInputLayerName
) = 0;
```

Parameters | Description
--- | ---
pInputLayerName | Input name of the input layer to find.

Returns

Returns the named AcMapIEInputLayer if found; otherwise, returns NULL.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
An iterator over a collection of name-value pairs. The case-insensitive name component of a name-value pair is a key value and must be unique within the collection. This class (which is similar to the STL map class) differs from the \texttt{AcMapIEExpressionTargetIterator} class in that this class has more lenient naming rules for the name component of a name-value pair. Do not explicitly delete any strings returned by any of the functions or their output parameters in this class (but delete all other types of returned objects).

\textbf{class} \texttt{AcMapIENValueIterator};

\textbf{File}

\texttt{AcMapIENValueIterator.h}

\textbf{Methods}

\texttt{\textasciitilde AcMapIENValueIterator} \textbf{Destroys} an instance of this class.

\texttt{Add} \textbf{Inserts} a new element into the collection after the current element and advances the iterator to the new element. After adding a new element successfully, call \texttt{Rewind()} to iterate over the entire collection.

\texttt{Clear} \textbf{Removes} all elements from the collection.

\texttt{Done} \textbf{Determines} whether the iterator has reached the end of the collection.

\texttt{Find} \textbf{Searches} the collection for an element.

\texttt{Get} \textbf{Retrieves} the current element in the iteration.

\texttt{Remove} \textbf{Removes} an element from the collection.

\texttt{Rewind} \textbf{Moves} to the first element in the iteration.

\texttt{Set} \textbf{Sets} the value of a name-value pair in the collection.

\texttt{Step} \textbf{Advances} to the next element in the iteration.

Created with a commercial version of \texttt{Doc-O-Matic}. In order to make this
message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapIENameValueIterator Class, AcMapIENameValueIterator Class
AcMapIENameValueIterator:: Add Method
AcMapIENameValueIterator Class | AcMapIENameValueIterator Class

Inserts a new element into the collection after the current element and advances the iterator to the new element. After adding a new element successfully, call Rewind() to iterate over the entire collection.

```cpp
virtual AcMapIE::ErrCode Add(
    const ACHAR* pszName,
    const ACHAR* pszValue
) = 0;
```

Parameters
- `pszName`: Input name of the name-value pair to add.
- `pszValue`: Input value of the name-value pair to add.

Returns

Returns AcMapIE::ErrCode kErr_OK if successful. Returns AcMapIE::ErrCode kErr_BadParams if either input string is NULL or empty. Returns AcMapIE::ErrCode kErr_BadFieldName if the input name is invalid. Returns AcMapIE::ErrCode kErr_Fail if the case-insensitive input name already exists in the collection.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Searches the collection for an element.

```cpp
virtual bool Find(
    constACHAR*&pszValue,
    constACHAR* pszName)
= 0;
```

**Parameters**

- `pszValue`: Output value corresponding to the input name if the name-value pair was found. An empty string indicates that the element was not found.
- `pszName`: Input case-insensitive name of the name-value pair to find.

**Returns**

Returns true if the element was found; otherwise, returns false. False indicates an invalid output value.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the current element in the iteration.

```cpp
template virtual bool Get(  
    const ACHAR*& pszName,  
    const ACHAR*& pszValue  
) = 0;
```

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pszName</td>
<td>Output name of the current name-value pair.</td>
</tr>
<tr>
<td>pszValue</td>
<td>Output value of the current name-value pair.</td>
</tr>
</tbody>
</table>

Returns

Returns true if the current element is valid, or false if the iteration has no more elements to retrieve.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Removes an element from the collection.

```cpp
virtual bool Remove(
    const ACHAR* pszName
) = 0;
```

**Parameters**

<table>
<thead>
<tr>
<th>pszName</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input case-insensitive name of the name-value pair to remove.</td>
</tr>
</tbody>
</table>

**Returns**

Returns true if the element was removed successfully; otherwise, returns false.

**Remarks**

If the iterator is pointing to the removed element, it advances to the next element if possible.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the value of a name-value pair in the collection.

```c
virtual AcMapIE::ErrCode Set(
    const ACHAR* pszName,
    const ACHAR* pszValue
) = 0;
```

**Parameters**

- **pszName**: Input case-insensitive name of the name-value pair to set.
- **pszValue**: Input new value for the name-value pair to set.

**Returns**

Returns AcMapIE::ErrCode kErr_OK if successful. Returns AcMapIE::ErrCode kErr_BadParams if either input string is NULL or empty. Returns AcMapIE::ErrCode kErr_BadFieldName if the input name is invalid. Returns AcMapIE::ErrCode kErr_Fail if the case-insensitive input name does not already exist in the collection.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
Import-Export

Functions
Import-Export

Functions

AcMapExporter

Retrieves the AutoCAD Map exporter object, as a singleton. Only a single instance of AcMapIEExporter exists (do not delete it). Always complete an active export before switching the current drawing. This function will succeed whenever the command-line command MAPEXPORT can succeed in the AutoCAD Map user interface. The return value NULL usually indicates an installation problem.

AcMapExportFormatIterator

Retrieves the AutoCAD Map export format iterator.

AcMapImporter

Retrieves the AutoCAD Map importer object, as a singleton. Only a single instance of AcMapIEImporter exists (do not delete it). Always complete an active import before switching the current drawing. This function will succeed whenever the command-line command MAPIMPORT can succeed in the AutoCAD Map user interface. The return value NULL usually indicates an installation problem.

AcMapImportFormatIterator

Retrieves the AutoCAD Map import format iterator.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the AutoCAD Map exporter object, as a singleton. Only a single instance of AcMapIEExporter exists (do not delete it). Always complete an active export before switching the current drawing. This function will succeed whenever the command-line command MAPEXPORT can succeed in the AutoCAD Map user interface. The return value NULL usually indicates an installation problem.

```
AcMapIEExporter* AcMapExporter();
```

File

AcMapIEGlobal.h

Returns

Returns a pointer to the exporter, or NULL if the exporter could not be retrieved. Do not explicitly delete this returned value.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the AutoCAD Map export format iterator.

```
AcMapIEFormatIterator* AcMapExportFormatIterator();
```

Returns

Returns a pointer to the `AcMapIEFormatIterator` export format iterator object, or `NULL` if `AcMapExporter()` returned `NULL`. Do not explicitly delete this returned value.

Remarks

This iterator is the same class as the import format iterator but is a different instance that provides information about the collection of export formats only.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the AutoCAD Map importer object, as a singleton. Only a single instance of AcMapIEImporter exists (do not delete it). Always complete an active import before switching the current drawing. This function will succeed whenever the command-line command MAPIMPORT can succeed in the AutoCAD Map user interface. The return value NULL usually indicates an installation problem.

    AcMapIEImporter* AcMapImporter();

Returns

Returns a pointer to the importer, or NULL if the importer could not be retrieved. Do not explicitly delete this returned value.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
Functions
AcMapImportFormatIterator Function
Functions

Retrieves the AutoCAD Map import format iterator.

AcMapIEFormatIterator* AcMapImportFormatIterator();

File
AcMapIEGlobal.h

Returns

Returns a pointer to the AcMapIEFormatIterator import format iterator object, or NULL if AcMapImporter() returned NULL. Do not explicitly delete this returned value.

Remarks

This iterator is the same class as the export format iterator but is a different instance that provides information about the collection of import formats only.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Map Book: Class-Based API

- **AcMapMbError**: This is class AcMapMbError. Map-book-management Map Book
- **AcMapMbMapBook**: This is a collection of static functions which manage map books.
- **AcMapMbMapBookManager**: Base class used to notify the application of Map Book events within an AutoCAD Map Book project.
- **AcMapMbMapSheetLayoutSettings**: Class AcMapMbMapSheetLayoutSettings defines a set of the settings to form a Map Sheet layout from a template drawing.
- **AcMapMbMapSheetTemplate**: This is class AcMapMbTile.
- **AcMapMbTile**: Class AcMapMbTileGenerator is a pure virtual base class which defines the interface for the Tile Generators.
- **AcMapMbTileGenerator**: This is a list of values we handle at the level of the base class. It is mostly for the maintenance of the settings classes. Map Book Scale Scale I
- **AcMapMbTileGeneratorSettings**: Tiling scheme DisableEmptyTiles IntersectEntities IntersectEntity Set OverlapPercent Class AcMapMbTileGeneratorSettings contains the added properties needed for tile generation.
- **AcMapMbTileGeneratorSettingsArea**: This is a list of values we handle for the Grid Tiler.
- **AcMapMbTileGeneratorSettingsGrid**: This is a list of values we handle for the Manual Tiler Selection set of plines.
- **AcMapMbTileGeneratorSettingsManual**: This is a list of values we handle for the Manual Tiler Selection set of plines.
AcMapMbTileNameGenerator virtual base class which defines the Name Generators.
Class AcMapMbTileNameGenerator inherits from the pure virtual base class AcMapMbTileNameGenerator the interface for the Name Generators.
Class AcMapMbTileNameGeneratorData contains the added property needed for data name generation.
Class AcMapMbTileNameGeneratorGrid contains the added properties needed for grid name generation.
Class AcMapMbTileNameGeneratorIndexer This class encapsulates a naming nugget.
Class AcMapMbTileNameGeneratorSequence inherits from the pure virtual base class AcMapMbTileNameGenerator the interface for the Name Generators.
Class AcMapMbTileNameGeneratorSettings the base settings class for name generation. It contains the properties needed by all derived classes. The user will not instantiate a member of this class as the derived classes have some properties needed by actual naming schemes.
The Name Generators and the Name Generator Settings classes have a direct relationship. For any future expansion, users will typically create both a new Generator and Settings class. The appropriate Generator will be obtained from the Settings class. The DECLARE_MEMBERS macro is used in the declaration of classes that are to be a part of the ObjectARX run-time tree.
Class AcMapMbTileNameGeneratorSettingsData contains the added property needed for name generation.
Class AcMapMbTileNameGeneratorSettingsGrid contains the added properties needed for grid name generation.
Class `AcMapMbTileNameGeneratorSettingsSequence` contains the added property needed for sequential name generation.

Provides functions to create, name the tiles used by a MapBook.
This is class AcMapMbError. Map-book-management Map B
This is a collection of static fun Manage map books.
Base class used to notify the ap Book events within an AutoC Derive custom reactors from th AcMapMBRecator class.
To add a reactor: Derive your c AcMapMBMyReactor from AcMapMbMapBookRecator.Over virtual functions of this reactor
Class AcMapMbMapSheetLayout defines a set of the settings to fo layout from a template drawing
Map-book-management Map S This is class AcMapMbTile.
Class AcMapMbTileGenerator base class which defines the int Tile Generators.
This is a list of values we handle the base class It is mostly for th maintanence of the settings class source Map Book Scale Scale I Tiling scheme DisableEmptyTi IntersectEntites InteresectEntity OverlapPercent Class AcMapMbTileGeneratorSetting settings class for tile generation Class AcMapMbTileGeneratorSettingsArea contains the added properties for tile generation.
**AcMapMbTileGeneratorSettingsGrid**  
This is a list of values we handle for Grid Tiler.

**AcMapMbTileGeneratorSettingsManual**  
This is a list of values we handle for Manual Tiler Selection set of pl ...

**AcMapMbTileNameGenerator**  
Class AcMapMbTileNameGenerator is a pure virtual base class which defines the interface for the Name Generators.

**AcMapMbTileNameGeneratorData**  
Class AcMapMbTileNameGeneratorData inherits from the pure virtual base class AcMapMbTileNameGenerator the interface for the Name Generators.

**AcMapMbTileNameGeneratorGrid**  
Class AcMapMbTileNameGeneratorGrid inherits from the pure virtual base class AcMapMbTileNameGenerator the interface for the Name Generators.

**AcMapMbTileNameGeneratorIndexer**  
This class encapsulates a naming nugget.

**AcMapMbTileNameGeneratorSequence**  
Class AcMapMbTileNameGeneratorSequence inherits from the pure virtual base class AcMapMbTileNameGenerator the interface for the Name Generators.

**AcMapMbTileNameGeneratorSettings**  
Class AcMapMbTileNameGeneratorSettings is the base settings class for name generation.

**AcMapMbTileNameGeneratorSettingsData**  
Class AcMapMbTileNameGeneratorSettingsData contains the added property needed for name generation.

**AcMapMbTileNameGeneratorSettingsGrid**  
Class AcMapMbTileNameGeneratorSettingsGrid contains the added properties needed for name generation.

**AcMapMbTileNameGeneratorSettingsSequence**  
Class AcMapMbTileNameGeneratorSettingsSequence contains the added property needed for sequential name generation.
AcMapMbTileSet  Provides functions to create, na
the tiles used by a MapBook.

Created with a commercial version of Doc-O-Matic. In order to make this
message disappear you need to register this software. If you have problems
registering this software please contact us at support@toolsfactory.com.
This is class AcMapMbError.

```cpp
class AcMapMbError;
```

File

AcMapMbError.h

Enumerations

- **EMbStatus**
  This enum represents the list of defined Map Book errors.

- **ErrorClass**
  KMapMbClassErrors represents a type of errors associated with Map Book application.

Methods

- **~AcMapMbError**
  Destroys an instance of this class.

- **AcMapMbError**
  Constructs an instance of this class.

- **IsError**
  Checks if the status code indicates an error.

- **IsSuccess**
  Checks if the status code indicates success.

- **IsWarning**
  Checks if the status code indicates a warning.

- **Message**
  Loads an error message associated with the error code from resource file.

- **PushErrorMessage**
  Pushes an error entry to the error stack.

- **PushErrorMessage**
  Pushes an error entry to the error stack.

Created with a commercial version of [Doc-O-Matic](https://www.docOMATIC.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapMbError Class, AcMapMbError Class
AcMapMbError:: PushErrorMessage Method
AcMapMbError Class | AcMapMbError Class

Pushes an error entry to the error stack.

```c
static void PushErrorMessage(
    EMbStatus es,
    const ACHAR* pszParam
);
```

Parameters | Description
--- | ---
es | Input error status
pszParam | Input string error parameter

Returns
Nothing.

Remarks

The entry includes kMapMbClassErrors as error type, es as an error code and the error message made from pszParam and format string associated with es if any.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Map-book-management Map Book class.

class AcMapMbMapBook : public AcDbObject;

Remarks

These objects are stored in the drawing in an AcDbDictionary and managed by the MapBookManager.

Enumerations

- **EModificationType**
  Bit flags indicating to a reactor what kind of modification is taking place.

Methods

- **~AcMapMbMapBook**
  Destroys an instance of this class.
- **AcMapMbMapBook**
  Constructs an instance of this class.
- **dwgInFields**
  Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.
- **dwgOutFields**
  Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.
- **dxfInFields**
  Lets this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.
- **dxfOutFields**
  Lets this object write its data. See also dxfOutFields()
in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

Retrieves the coordinate system name string.

Retrives the north direction.

Retrives the scale string.

Gets the Map Sheet Layout Settings used to generate the Map Book.

Retrives the number of tiles.

Returns the implementation object.

Checks if the Map Book in the current configuration has been saved so that it can be published.

Checks if a refered Sheet Set file exists and has been created not later than the drawing file.

Retrives the name of this Map Book.

Publishes the specified tiles to the DWF file, if there are no tiles specified, publishes the entire Map Book.

Publishes the specified tiles to the plotter specified by the configuration file.

Recreates a Sheet Set.

Resets a reference to a Sheet Set and refreshes sheets with the proper drawing location.

Sets the coordinate system name.

Sets the name of the map Book.

Sets the north direction.

Sets the scale string.

Retrives the file name of the Map Sheet Set associated with this MapBook.

Retrives the Subset name within the Map Sheet Set if any.

Shows/hides grid objects (polylines) associated with
ShowGridObjects

Invoked from within close() before the close actually occurs. The default implementation of this function returns Acad::eOk. See also subClose() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

subClose

Invoked from within erase() before the erase actually occurs. See also subErase() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

subErase

Tiles

Gets the Tile Set of the Map Book.

TileSet

Grants control of deep clone operations to the object. In the default implementation, the object is cloned and appended to the owner object pOwnerObject. See also wblockClone() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

wblockClone

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the coordinate system name string.

\[
\text{Acad::ErrorStatus GetCoordSysName(}
\begin{array}{l}
\text{ACHAR*}&
\text{pszCoordSysName}
\end{array}
\text{ ) const; }
\]

\begin{center}
\begin{tabular}{|c|c|}
\hline
\textbf{Parameters} & \textbf{Description} \\
\hline
dCoordSysName & Output the string representing coordinate system description string. \\
\hline
\end{tabular}
\end{center}

\textbf{Returns}

Returns Acad::eOk if successful; otherwise, returns a different error code.

\textbf{Remarks}

This is a Map Book property stored with the Sheet or Sheet Subset as Coordinate System custom property for the purpose of displaying the description as a field on the layouts. This value can be set by the user thru the sheet set manager UI.

Created with a commercial version of \textbf{Doc-O-Matic}. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the scale string.

```cpp
Acad::ErrorStatus GetScaleString(
    ACHAR*& pszScaleStr
) const;
```

**Parameters**

- `pszScaleStr`: Output a scale string.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

**Remarks**

This is a Map Book property stored with the Sheet or Sheet Subset as Scale custom property in "1:XXX" format; the initial value is generated from the actual layout format. Changing this property does not alter the Map Book, it just changes a Sheet Set property.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbMapBook Class, AcMapMbMapBook Class
AcMapMbMapBook:: IsSaveRequired Method
AcMapMbMapBook Class | AcMapMbMapBook Class

Checks if the Map Book in the current configuration has been saved so that it can be published.

```cpp
bool IsSaveRequired() const;
```

Returns

Returns true if the drawing has not been saved since the Map Book has been created or rebuilt.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Checks if a referred Sheet Set file exists and has been created not later than the drawing file.

```cpp
bool IsSheetSetValid() const;
```

Returns true if the .DST file is valid, false otherwise.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Publishes the specified tiles to the DWF file, if there are no tiles specified, publishes the entire Map Book.

```cpp
Acad::ErrorStatus PublishToDwf(
    const AcArray<AcMapMbMapBookManager::TILEID>& tileIds,
    const ACHAR* pszFileName
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tileIds</td>
</tr>
</tbody>
</table>

Input Ids of the tiles to be published; if the array is empty, the entire Map Book will be published.

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pszFileName</td>
</tr>
</tbody>
</table>

Input the name of the DWF file to be created; if the file with this name already exists it will be overwritten.

**Returns**

Returns Acad::eOk if successful; returns Acad::eDwgNeedsAFullSave if drawing must be saved; Acad::eNotCurrentDatabase in the case the Sheet Set is out of synch with the Map Book otherwise, returns a different error code.

**Remarks**

If the database has changed, the drawing must be saved before publishing.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Publishes the specified tiles to the plotter specified by the configuration file.

```cpp
Acad::ErrorStatus PublishToPlotter(
    const AcArray<AcMapMbMapBookManager::TILEID>& tileIds,
    const ACHAR* pszConfigFileName = NULL
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>tileIds</strong></td>
</tr>
<tr>
<td>Input Ids of the tiles to be published; if the array is empty, the entire Map Book will be published.</td>
</tr>
<tr>
<td><strong>pszConfigFileName</strong></td>
</tr>
<tr>
<td>Input the name of the plot configuration file; if not specified, the current plotter settings associated with the Sheet will be used.</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; returns Acad::eDwgNeedsAFullSave if drawing must be saved; Acad::eNotCurrentDatabase in the case the Sheet Set is out of synch with the Map Book otherwise, returns a different error code.

**Remarks**

If there are no tiles specified, publishes the entire Map Book. If the configuration file is not specified, the current plotter settings associated with the Sheet will be used. If the database has changed, the drawing must be saved before publishing.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Recreates a Sheet Set.

```cpp
Acad::ErrorStatus RebuildSheetSet(
    const TCHAR* pPathName = NULL
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pPathName</td>
<td>Passed-in a .DST path name. If NULL use the existing path.</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Resets a reference to a Sheet Set and refreshes sheets with the proper drawing location.

```
Acad::ErrorStatus ResolveSheetSet(
    const ACHAR* pszFileName
);
```

Parameters

- `pszFileName` Input Sheet Set file name.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Remarks

The specified file should be the sheet set generated with the Map Book. The sheet set manager has the ability to find a given sheet set when that sheet set has been moved to a different location in the file system. See sheet set manager documentation on this feature.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the coordinate system name.

```cpp
Acad::ErrorStatus SetCoordSysName(
    const ACHAR* pszCoordSysName
);
```

**Parameters**

- **dCoordSysName**
  
  Input the string representing coordinate system description string.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

**Remarks**

This is a Map Book property stored with the Sheet or Sheet Subset as CoordinateSystem custom property. Changing this Map Book property does not change the actual coordinate system associated with the drawing file.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the name of the map Book.

`Acad::ErrorStatus SetName(
    const ACHAR* pszName
);`

Parameters

<table>
<thead>
<tr>
<th>pszName</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Input Map Book name.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the scale string.

```
Acad::ErrorStatus SetScaleString(  
    const ACHAR* pszScaleStr
);
```

**Parameters** 

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input a scale string; it is up to an application what string to specify, but in general it should be a string in &quot;1:XXX&quot; format.</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

**Remarks**

This is a Map Book property stored with the Sheet or Sheet Subset as Scale custom property.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
This is a collection of static functions which Manage map books.

```cpp
class AcMapMbMapBookManager;
```

**File**

AcMapMbMapBookManager.h

**Remarks**

There may be several Map Books in a drawing. They are resident in the drawing.

**Enumerations**

- **EAdjacentTile**
  Defines a direction for selecting adjacent tiles.

**Methods**

- **AddMapBookCreationReactor**
  Adds a map book creation reactor to monitor map book creation process such as end of tiling and sheet creation.

- **AddMapBookReactor**
  Adds a map-book-management reactor to monitor map book management related activities.

- **GenerateMapBook**
  Generates new Map Book based on the settings supplied.

- **GetAt**
  Retrieves the ID of the named map book.

- **GetAt**
  Retrieves the named map book.

- **GetCurrent**
  Retrieves the current Map Book.

- **GetMapBookForLayout**
  Retrieves an Id of a Map Book associated with the given layout if any.

- **Has**
  Checks if a Map Book with the specified name has already been defined within a drawing database.

Determines whether a project has map book
HasMapBookData data.

LoadSettings Fills appropriate settings objects and populates them from the specified XML file.

mbMapBookManagerImp Returns the implementation object.

NewIterator Returns a new map book dictionary iterator.

Remove Removes the map book with the specified ID.

RemoveMapBookCreationReactor Removes a map book creation reactor.

RemoveMapBookReactor Removes a map book reactor from a database.

ResetName Renames the map book with the specified ID.

ResetTileView Resets the tile view for the specified tile.

SaveSettings Save the specified map book settings in the specified XML file.

SetCurrent Sets the current Map Book.

SetMapBookDMEnvironment Sets the Display Management environment such as DM Map and Scale associated with the Map Book.

SetMapBookEnvironment Sets the environment associated with the Map Book.

UpdateMapBook Modifies existing Map Book based on the settings supplied.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Generates new Map Book based on the settings supplied.

```cpp
static Acad::ErrorStatus GenerateMapBook(
    AcDbObjectId& Id,
    AcDbDatabase* pDatabase,
    const ACHAR* pszName,
    const AcMapMbTileGeneratorSettings* pTileGeneratorSettings,
    const AcMapMbTileNameGeneratorSettings* pTileNameGeneratorSettings,
    const AcMapMbMapSheetLayoutSettings* pLayoutTemplateSettings,
    const ACHAR* pszSheetSetFileName,
    const ACHAR* pszSubsetName = NULL);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Id</td>
<td>Output ID of the created Map Book object.</td>
</tr>
<tr>
<td>pDatabase</td>
<td>Input drawing database.</td>
</tr>
<tr>
<td>pszName</td>
<td>Input name of the map.</td>
</tr>
<tr>
<td>pTileGeneratorSettings</td>
<td>Input tile generator settings object. The user will create the correct settings object and set the appropriate values. It is never correct to use a base class settings object.</td>
</tr>
<tr>
<td>pTileNameGeneratorSettings</td>
<td>Input tile name generator settings object. The user will create the correct settings object and set the appropriate values. It is never correct to use a base class settings object.</td>
</tr>
<tr>
<td>pLayoutTemplateSettings</td>
<td>Input layout template settings object. The user will create the correct settings object and set the appropriate values.</td>
</tr>
<tr>
<td>pszSheetSetFileName</td>
<td>Input Sheet Set file name, if the file with this name does not exist, it will be created. If the file already exists it will be overwritten.</td>
</tr>
<tr>
<td>pszSubsetName</td>
<td>Input category name within the Map Sheet Set. The category name includes the full path where a slash &quot;/&quot; is used as a delimiter, for example: &quot;Root/Nested_Subset/Map_Book_category&quot;. If</td>
</tr>
</tbody>
</table>
this Sheet Set category already exists, it will be overwritten.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the ID of the named map book.

```cpp
static Acad::ErrorStatus GetAt(
    AcDbObjectId& Id,
    AcDbDatabase* pDatabase,
    const ACHAR* pszName
);
```

Parameters

- **Id**
  - Description: Output ID of the map book.
- **pDatabase**
  - Description: Input pointer to a drawing database.
- **pszName**
  - Description: Input name of the map book.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the named map book.

```cpp
static Acad::ErrorStatus GetAt(
    AcMapMbMapBook*& pMapBook,
    AcDbDatabase* pDatabase,
    const ACHAR* pszName,
    AcDb::OpenMode mode
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pMapBook</td>
<td>Output AcMapMbMapBook.</td>
</tr>
<tr>
<td>pDatabase</td>
<td>Input pointer to a drawing database.</td>
</tr>
<tr>
<td>pszName</td>
<td>Input name of the map book.</td>
</tr>
<tr>
<td>mode</td>
<td>Input mode for opening the map book.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Checks if a Map Book with the specified name has already been defined within a drawing database.

```cpp
static bool Has(
    const ACHAR* pszName,
    AcDbDatabase* pDatabase
);
```

Parameters

- `pszName`: Input name of the map book.
- `pDatabase`: Input pointer to a drawing database.

Returns

Returns true if a Map Book with the specified name is already defined; false otherwise.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Fills appropriate settings objects and populates them from the specified XML file.

Returns Acad::eOk if successful; otherwise, returns a different error code.

```cpp
static Acad::ErrorStatus LoadSettings(
    AcArray<AcDbObject*>& SettingsArray,
    const ACHAR* pFileName = NULL
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pszFileName</td>
<td>Input name of the XML file. This should be a fully qualified file name.</td>
</tr>
<tr>
<td>pSettingsArray</td>
<td>Output settings array.</td>
</tr>
</tbody>
</table>

Remarks

This function may push errors onto the AutoCAD Map Error Stack. Typically the user will add no more than one of each derived type of settings object to the array. Valid objects can be expected only if an object of that type was saved previously.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapMbMapBookManager Class, AcMapMbMapBookManager Class
AcMapMbMapBookManager::ResetName Method
AcMapMbMapBookManager Class | AcMapMbMapBookManager Class

Renames the map book with the specified ID.

```cpp
static Acad::ErrorStatus ResetName(
    const AcDbObjectId& mapId,
    const ACHAR* pszNewName
);
```

**Parameters**

- `mapId`  
  Input ID of the map book to rename.

- `pszNewName`  
  Input new map book name.

**Returns**

Returns Acad::eOk if successful; returns Acad::eDuplicateKey if there is another map book with this name returns Acad::eInvalidInput if the name is not a valid Extended Symbol name or if the name is the same; otherwise, returns a different error code.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at **support@toolsfactory.com**.
Links
AcMapMbMapBookManager Class, AcMapMbMapBookManager Class
AcMapMbMapBookManager:: ResetTileView Method
AcMapMbMapBookManager Class | AcMapMbMapBookManager Class

Resets the tile view for the specified tile.

```cpp
static bool ResetTileView(
    AcMapMbMapBookManager::TILEID tileId,
    const AcDbObjectId& MapBookId = NULL
);
```

Parameters Description
---
tileId Input tile Id.
MapBookId Input Map Book Id, curren Map Book by default.

Returns

Returns true if the project has map book data.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Save the specified map book settings in the specified XML file.

```c++
static Acad::ErrorStatus SaveSettings(
    AcArray<AcDbObject*>& SettingsArray,
    const ACHAR* pFileName
);
```

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pszFileName</td>
<td>Input name of the XML file. This should be a fully qualified file name.</td>
</tr>
<tr>
<td>pSettingsArray</td>
<td>Input array of valid settings objects.</td>
</tr>
</tbody>
</table>

### Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

### Remarks

This function may push errors onto the AutoCAD Map Error Stack. Typically the user will save settings objects of at most 1 of each type of derived settings objects.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapMbMapBookManager Class, AcMapMbMapBookManager Class
AcMapMbMapBookManager:: UpdateMapBook Method
AcMapMbMapBookManager Class | AcMapMbMapBookManager Class

Modifies existing Map Book based on the settings supplied.

```cpp
static Acad::ErrorStatus UpdateMapBook(
    const AcDbObjectId& Id,
    const AcMapMbTileGeneratorSettings* pTileGeneratorSettings,
    const AcMapMbTileNameGeneratorSettings* pTileNameGeneratorSettings,
    const AcMapMbMapSheetLayoutSettings* pLayoutTemplateSettings,
    const ACHAR* pszSheetSetFileName = NULL,
    const ACHAR* pszSubsetName = NULL);
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Id</td>
<td>Input ID of the Map Book object to modify.</td>
</tr>
<tr>
<td>pTileGeneratorSettings</td>
<td>Input tile generator settings object. The user will create the correct settings object and set the appropriate values. It is never correct to use a base class settings object. It is not necessary to use the same type settings object as when the MapBook was created.</td>
</tr>
<tr>
<td>pTileNameGeneratorSettings</td>
<td>Input tile generator settings object. The user will create the correct settings object and set the appropriate values. It is never correct to use a base class settings object. It is not necessary to use the same type settings object as when the MapBook was created.</td>
</tr>
<tr>
<td>pLayoutTemplateSettings</td>
<td>Input layout template settings object. The user may edit values from previous settings.</td>
</tr>
<tr>
<td>pszSheetSetFileName</td>
<td>Input Sheet Set file name, if the file with this name does not exist, it will be created. NULL value indicates that previous sheet set data will be overwritten.</td>
</tr>
<tr>
<td>pszSubsetName</td>
<td>Input category name within the Map Sheet Set. The category name includes the full path where a slash &quot;/&quot; is used as a delimiter, for example: &quot;Root/Nested_Subset/Map_Book_category&quot;. If</td>
</tr>
</tbody>
</table>
this Sheet Set already exists, it will be overwritten

pszName Input name of the map.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Base class used to notify the application of Map Book events within an AutoCAD Map project. Derive custom reactors from the AcMapMBRecator class.

To add a reactor: Derive your class AcMapMBMyReactor from AcMapMbMapBookRecator. Override the virtual functions of this reactor base class.

class AcMapMbMapBookReactor;

File
AcMapMbMapBookReactor.h

Methods

MapBookAppended Invoked when a new map book is appended or after a reactor is attached to a project that has map book(s) defined.

MapBookErased Invoked when a map book is detached or erase operation is undone.

MapBookModified Invoked when a map book is detached or erase operation is undone.

MapBookSetCurrent Invoked when a current Map Book gets set or at the very begining of the Map session.

MapBookTileModified Invoked when a map book tile is modified.

MapBookTileWillBeErased Invoked when a map book tile is about to be erased.

MapBookTreeNodeModified Invoked when a map book tree node is modified.

MapBookWillBeErased Invoked when a map book is about to be erased or erase operation is about to be undone.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links

Classes
AcMapMbMapSheetLayoutSettings Class

Class AcMapMbMapSheetLayoutSettings defines a set of the settings to form Map Sheet layout from a template drawing.

class AcMapMbMapSheetLayoutSettings : public AcDbObject;

File

AcMapMbMapSheetLayoutSettings.h

Enumerations

- EMapKeyType Enumerates supported types of a Map Key.
- EMapLegendType Enumerates supported types of legend

Methods

- ~AcMapMbMapSheetLayoutSettings Destroys an instance of this class.
- AcMapMbMapSheetLayoutSettings Constructs an instance of this class.
- AdjacentArrowBlockName Retrieves an adjacent arrow block name.
- AllocateMapSheetTemplate Allocates an instance of Map Sheet Template class.
- clone Clones the object. See also clone() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.
- copyFrom Copies the contents of an object into the messaged object, if feasible. See also copyFrom() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.
- Lets this object read its data. See also dwgInFields() in the AutoCAD
**dwgInFields**

ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

Lets this object write its data. See also `dwgOutFields()` in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

**dwgOutFields**

ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

Lets this object read its data. See also `dxfInFields()` in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

**dxfInFields**

ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

Lets this object write its data. See also `dxfOutFields()` in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

**dxfOutFields**

ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

**GetLegendBox**

Retrieves a Legend box.

**GetMapKeyLinkDrawing**

Retrieves a Display Manager Map and threshold scale value to associate a link drawing with the Linked Drawing Map Key.

**Implementation**

Returns the implementation object.

**IncludeAdjacentArrows**

Retrieves the flag to include a adjacent arrows to a Map Sheet layout.

**IncludeTitleBlock**

Retrieves the flag to include a title block to a Map Sheet layout.

**LegendType**

Retrieves a Legend type.

**MapKeyExternalFileName**

Retrieves a drawing file name for the External Drawing Map Key.

**MapKeyLayerPattern**

Retrieves a pattern to select layers visible within a current drawing Map Key.
MapKeyType

Sets adjacent arrow block name.

SetIncludeAdjacentArrows

Sets the flag to include adjacent arrows to a Map Sheet layout.

SetIncludeTitleBlock

Sets the flag to include a title block to a Map Sheet layout.

SetLegendBox

Sets a Legend box.

SetLegendType

Sets a Legend type.

SetMapKeyExternalFileName

Sets a drawing file name for the External Drawing Map Key.

SetMapKeyLayerPattern

Sets a pattern to select layers visible within a current drawing Map Key.

SetMapKeyLinkDrawing

Sets a Display Manager Map and threshold scale value to associate a link drawing with the Linked Drawing Map Key.

SetMapKeyType

Sets a Map Key type.

SetTemplateFileName

Sets a template file name.

SetTemplateLayoutName

Sets a template layout name.

SetTitleBlockName

Sets a title block name.

TemplateFileName

Retrieves a template file name.

TemplateLayoutName

Retrieves a template layout name.

TitleBlockName

Retrieves a title block name.

xmlInFields

Let this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

xmlOutFields

Let this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.
Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieve a Display Manager Map and threshold scale value to associate a link drawing with the Linked Drawing Map Key.

```cpp
Acad::ErrorStatus GetMapKeyLinkDrawing(
    const ACHAR*& pszMapName,
    double& dThresholdScale
) const;
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pszMapName</td>
<td>Output Display Manager Map name.</td>
</tr>
<tr>
<td>dThresholdScale</td>
<td>Output Display Manager Map Threshold Scale value.</td>
</tr>
</tbody>
</table>

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets an adjacent arrow block name.

```cpp
Acad::ErrorStatus SetAdjacentArrowBlockName(
    const ACHAR* pszBlockName);
```

**Parameters**

- **pszBlockName**: Input adjacent arrow block name. This block has to be defined within a template file specified via SetTemplateFileName()

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets a drawing file name for the External Drawing Map Key.

```
Acad::ErrorStatus SetMapKeyExternalFileName(
    const ACHAR* pszFileName
);
```

**Parameters**

- **pszFileName**
  - Input file name. The name should specify a complete path name with the .DWG extension. The function will validate to insure the file exists.

**Returns**

- Returns Acad::eOk if successful; Acad::eFilerError if the file cannot be found; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets a pattern to select layers visible within a current drawing Map Key.

\[
\text{Acad::ErrorStatus SetMapKeyLayerPattern( const ACHAR* pszLayerPattern );}
\]

Parameters

<table>
<thead>
<tr>
<th>pszLayerPattern</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input a pattern to select layers visible within a current drawing Map Key. The pattern relies on acdbSNValid() to select layers.</td>
<td></td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Remarks

The patterns will be comma delimited, and may contain the regular expressions defined for acutWcMatch.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets a Display Manager Map and threshold scale value to associate a link drawing with the Linked Drawing Map Key.

```cpp
Acad::ErrorStatus SetMapKeyLinkDrawing(
    const ACHAR* pszMapName,
    double dThresholdScale
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pszMapName</td>
<td>Input Display Manager Map name. The Display Manager Map with this name should have been defined in the current drawing.</td>
</tr>
<tr>
<td>dThresholdScale</td>
<td>The specified Display Manager map should have this Threshold Scale value defined.</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docOMATIC.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets a template file name.

```cpp
Acad::ErrorStatus SetTemplateFileName(
    const ACHAR* pszFileName
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pszFileName</td>
<td>Input file name. The name should specify a complete path name with the .DWT or .DWG file extension. This method validates that a file with the specified name exists.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Remarks

A layout from this template will be used as a prototype for the Map Sheet layout.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets a template layout name.

\begin{verbatim}
Acad::ErrorStatus SetTemplateLayoutName(
    const ACHAR* pszLayoutName
);
\end{verbatim}

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pszLayoutName</td>
<td>Input template layout name. A template with this name should exist within a template file.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets a title block name.

```cpp
Acad::ErrorStatusSetTitleBlockName(
    const ACHAR* pszBlockName
);
```

**Parameters**

- **pszBlockName**
  - Description: Input title block name. This block has to be defined within a template file specified via SetTemplateFileName(). Setting the title block name switches it on, i.e. IncludeTitleBlock() will return true.
  
**Returns**

- Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](http://www.docs-factory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbMapSheetLayoutSettings Class, AcMapMbMapSheetLayoutSettings Class
AcMapMbMapSheetLayoutSettings::xmlInFields Method
AcMapMbMapSheetLayoutSettings Class | AcMapMbMapSheetLayoutSettings Class

Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus xmlInFields(
    AcMapMbXmlFiler* pFiler
);
```

**Parameters**

- **pFiler**
  
  Description: Input filer to use to read the object's data.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links

AcMapMbMapSheetLayoutSettings Class, AcMapMbMapSheetLayoutSettings Class
AcMapMbMapSheetLayoutSettings::xmlOutFields Method
AcMapMbMapSheetLayoutSettings Class | AcMapMbMapSheetLayoutSettings Class

Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus xmlOutFields(
    AcMapMbXmlFiler* pFiler
) const;
```

Parameters Description
pFiler Input filer to use to write the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Map-book-management Map Sheet Template.

class AcMapMbMapSheetTemplate;
File
AcMapMbMapSheetTemplate.h

Remarks
This class provides a wrapper for the Template drawing file and provides helper functions as needed to access this file.

Enumerations

ETemplateElementType Defines template elements.

Methods

~AcMapMbMapSheetTemplate Destroys an instance of this class.
AcMapMbMapSheetTemplate Constructs an instance of this class.
AdjacentEnumToString Gets the key string for the specified direction.
AdjacentStringToEnum Gets the adjacent direction for the specified key string value.
CloneLayout Clones the layout specified by the settings object from the template drawing to the target drawing.
Close Closes the template drawing file.
DwgDatabase Retrieves the drawing database object associated with the Map Sheet template.
GetAdjacentArrow Gets the adjacent arrow Id for the specified direction from the layout specified by the settings object.
GetAdjacentArrowIds Gets the adjacent arrow Ids for layout specified
GetAdjacentArrows

by the settings object.

GetArrowDirection

Gets adjacency direction from the adjacent arrow placeholder.

GetBlockNames

Retrieves the list of block names defined within the specified Map Sheet template drawing.

GetElement

Retrieves the Id of the specified template element.

Returns the object id of the object in the specified layout that is marked as an element of the specified type.

Returns Acad::eOk if successful; Returns Acad::eAmbiguousOutput if there is more than one element of the specified type. Returns Acad::eKeyNotFound if there are no elements of the specified type; otherwise, returns a different error code.

Returns an array of object ids of the objects in the specified layout that are marked as an element of the specified type.

Returns Acad::eOk if successful; Returns Acad::eKeyNotFound if there are no entities of the specified type; otherwise, returns a different error code.

Retrieves the list of layout names defined within the specified Map Sheet template drawing.

GetTileSize

Gets height and width of tile from the Map viewport.

Implementation

Returns the implementation object.

IsElement

Checks if the object specified by the Id is an element of the specified type.

LayoutId

Retrieves current template layout Id.

Marks the selected template object with XData so that it is considered as a template element of the specified type.

MarkElement
Returns Acad::eOk if successful; otherwise, returns a different error code.

**Open**

Opens the template drawing file for reading. Remove marker XData from the selected template object so that it is considered as a template element any more. Returns Acad::eOk if successful; otherwise, returns a different error code.

**UnMarkElement**

Validates if the specified layout can be used as a Map Sheet layout.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbMapSheetTemplate Class, AcMapMbMapSheetTemplate Class
AcMapMbMapSheetTemplate::AdjacentEnumToString Method
AcMapMbMapSheetTemplate Class | AcMapMbMapSheetTemplate Class

Gets the key string for the specified direction.

```cpp
static Acad::ErrorStatus AdjacentEnumToString(
    const ACHAR*& pszString,
    AcMapMbMapBookManager::EAdjacentTile adjDirection
);
```

Parameters
- `pszString`  Description: Output key string
- `adjDirection`  Description: Input adjacent direction

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Gets the adjacent direction for the specified key string value.

```cpp
static Acad::ErrorStatus AdjacentStringToEnum(
    AcMapMbMapBookManager::EAdjacentTile & adjDirection,
    const ACHAR* pszSting
);
```

Parameters

- `adjDirection` - Output adjacent direction
- `pszSting` - Input key string

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Clones the layout specified by the settings object from the template drawing to the target drawing.

```cpp
virtual Acad::ErrorStatus CloneLayout(
    AcDbDatabase* pTargetDb,
    const ACHAR* pszDestinationName,
    AcDbIdMapping& idMapping
) const;
```

**Parameters**

- `pTargetDb`: Input target drawing database.
- `pszDestinationName`: Input name to be assigned to the cloned layout in the destination database.
- `idMapping`: Input after cloning contains mapping between template database and destination database IDs.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

**Remarks**

The specified name gets assigned to the layout in the destination drawing.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at **support@toolsfactory.com**.
Retrieves the list of block names defined within the specified Map Sheet template drawing.

```cpp
static Acad::ErrorStatus GetBlockNames(
    AcMapStringArray& aBlockNames,
    ACHAR* pszFileName
);
```

**Parameters**

- **aBlockNames**
  - Description: Output array of block names defined within a drawing file. Anonymous blocks are filtered out from this list.

- **pszFileName**
  - Description: Input file name. The name should specify a complete path name with the .DWT or .DWG extension. This method validates if a file with the specified name exists.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the list of layout names defined within the specified Map Sheet template drawing.

```cpp
static Acad::ErrorStatus GetLayoutNames(
    AcMapStringArray& aLayoutNames,
    const ACHAR* pszFileName
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>aLayoutNames</td>
<td>Output array of layout names defined within a drawing file.</td>
</tr>
<tr>
<td>pszFileName</td>
<td>Input file name. The name should specify a complete path name with the .DWT or .DWG extension. This method validates if a file with the specified name exists.</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Validates if the specified layout can be used as a Map Sheet layout.

```cpp
static bool Validate(
    const ACHAR * pszFileName,
    const ACHAR* pszLayoutName
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pszFileName</td>
<td>Input file name. The name should specify a complete path name with the .DWT or .DWG extension. This method validates if a file with the specified name exists and that the specified layout exists within that file.</td>
</tr>
<tr>
<td>pszLayoutName</td>
<td>Input layout name</td>
</tr>
</tbody>
</table>

**Returns**

Returns true if this is a valid template layout; otherwise, returns false.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
This is class AcMapMbTile.

class AcMapMbTile : public AcRxObject;

File

AcMapMbTile.h

Methods

~AcMapMbTile  Destroys an instance of this class.
AcMapMbTile  Constructs an instance of this class.
AcMapMbTile  Constructs an instance of this class.
AcMapMbTile  Copies an instance of this class.

AddHyperlink  Assignes a hyperlink to this tile to the specified adjacent arrow object.
Center  Returns the center of a tile.
GetAdjacentTile  Gets the adjacent tile of the specified direction for the tile.
GetAdjacentTile  Gets the adjacent tile of the specified direction for the tile.
GetLayoutIdAt  Returns the layout object id at the index.
GetTileSize  Returns the size of the tile.
GetViewPortSize  Gets the size of the tile as viewed.
Implementation  Returns the implementation object.
IsEnabled  Gets whether the tile is enabled.
Name  Returns the name of the tile.
OnTileErased  This is a notification that every tile except the erased one receives.
RefObject  Returns the reference object associated with the tile.
SetAdjacentTile  Sets the adjacent tile of the specified direction for the tile.
SetName                  Sets a tile name.
SheetHandle             Returns the unique handle of the Sheet in the Sheet Set.
StreamIn                Reads data from a dwg file.
StreamIn                Reads data from a dxf file.
StreamOut               Loads data into a dwg file.
StreamOut               Loads data into a dxf file.
TileId                  Returns the unique tile Id for this tile.
TileSet                 Returns the pointer to the Tile Set it belongs to.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets a tile name.

**virtual** Acad::ErrorStatus SetName(
    **const** ACHAR* pszName
);

**Parameters**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input tile name. The name should specify a valid tile name. Tile names follow Extended Symbol Name rules. Many AutoCAD symbols such as Layer Names follow these rules.</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code. Returns Acad::eNullPtr if pszName is NULL. Returns Acad::eInvalidInput if the input name does not conform to Extended Symbol Name rules.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Class AcMapMbTileGenerator is a pure virtual base class which defines the interface for the Tile Generators.

```csharp
class AcMapMbTileGenerator;
```

File

AcMapMbTileGenerator.h

Remarks

All Tile Generators have the same interface and are closely linked with their respective Settings classes. The TileGenerator should not be instantiated directly, but must be obtained from the appropriate settings class. A settings class of the appropriate type must be passed back into the GenerateTiles function.

Methods

- `~AcMapMbTileGenerator` Destroys an instance of this class.
- `AcMapMbTileGenerator` Constructs an instance of this class.
- `GenerateTiles` Creates the tile set from the settings.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
This is a list of values we handle at the level of the base class It is mostly for the creation and maintainence of the settings classes Map Book source Map Book Scale Scale DM Map name Tiling scheme DisableEmptyTiles IntersectEntites InteresectionEntitySet OverlapPercent Class AcMapMbTileGeneratorSettings is the base settings class for tile generation.

```cpp
class AcMapMbTileGeneratorSettings : public AcDbObject;
```

AcMapMbTileGeneratorSettings.h

Remarks

It contains the properties needed by all derived classes. The user will not instantiate a member of this class as the derived classes have some properties needed by actual tiling schemes. The Tile Generators and the Tile Generator Settings classes have a direct relationship. For any future expansion, users will typically create both a new Generator and Settings class. The appropriate Generator will be obtained from the Settings class. The DECLARE_MEMBERS macro is used in the declaration of classes that are to be a part of the ObjectARX run-time tree.

- **Enumerations**
  - **EMapBookSource**
    - This enum defines the different sources for a MapBook.
  - **ETilingScheme**
    - This enum defines the different tiling schemes we have available.

- **Methods**
  - ~GetAcMapMbTileGeneratorSettings
    - Destroys an instance of this class.
  - GetAcMapMbTileGeneratorSettings
    - Constructs an instance of this class.
  - GetActiveTilingScheme
    - Returns an enum which specifies the active tiling scheme.
**AllocateTileGenerator**

This method is used to allocate a tile generator that can be used by the settings object.

**copyFrom**

Copies the contents of an object into the messaged object, if feasible. See also copyFrom() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

**DisableEmptyTiles**

Returns the boolean value whether to disable empty tiles.

**DMMapName**

The pointer points to memory holding the dm map name. It is recommended that if you are going to keep this pointer for any length of time you make a copy of it as the contents can be changed at any time. The user should not attempt to delete this memory. This string can be empty to indicate model space but will never be NULL.

**DmScaleFactor**

Returns the scale factor of the DM Map for the Map Book.

**dwgInFields**

Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

**dwgOutFields**

Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

**dxfInFields**

Lets this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.
dxfoutFields

Let this object write its data. See also dxfoutFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

GetIntersectEntitySet

Returns the selection set of entities intersected.

Implementation

Returns the implementation object.

IntersectEntities

Returns the boolean value whether to use a particular set of entities as an intersection set when determining which tiles are empty.

Methods used to store and retrieve the data.

MapBookSource

Returns an enum which specifies the Source type.

OverlapPercent

Returns the Overlap percent.

ScaleFactor

Returns the scale factor for the Map Book.

SetActiveTilingScheme

Sets an enum which specifies the active tiling scheme.

SetDisableEmptyTiles

Sets the boolean value whether to disable empty tiles.

SetDMMMapName

Sets the DM Map name to be used in the tiler.

SetDmScaleFactor

Sets the scale of the DM Map for the Map Book.

SetIntersectEntities

Sets the boolean value whether to use the intersection set.

SetIntersectEntitySet

Sets the selection set of entities for determining empty tiles.

SetMapBookSource

Sets an enum which specifies the Source type.

SetOverlapPercent

Sets the Overlap data member.

SetScaleFactor

Sets the scale data member.

SetTileLayerName

Sets the tile layer name to be used in the tiler.

TileLayerName

Returns a pointer to memory holding the tile
layer name.

xmlInFields

Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

xmlOutFields

Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the DM Map name to be used in the tiler.

```
Acad::ErrorStatus SetDMMapName(
    const ACHAR* pszDMMapName
);
```

Parameters

- `pszDMMapName`: Input Display Manager Map name.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the tile layer name to be used in the tiler.

**virtual** Acad::ErrorStatus SetTileLayerName(
    **const** ACHAR* pszLayerName)

Parameters

pszLayerName

Description

Input layer name where tile reference polygons will be created.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbTileGeneratorSettings Class, AcMapMbTileGeneratorSettings Class
AcMapMbTileGeneratorSettings:: xmlInFields Method
AcMapMbTileGeneratorSettings Class | AcMapMbTileGeneratorSettings Class

Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

void virtual Acad::ErrorStatus xmlInFields(
   AcMapMbXmlFiler* pFiler
);

Parameters Description
pFiler Input filer to use to read the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbTileGeneratorSettings Class, AcMapMbTileGeneratorSettings Class
AcMapMbTileGeneratorSettings::xmlOutFields Method
AcMapMbTileGeneratorSettings Class | AcMapMbTileGeneratorSettings Class

Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus xmlOutFields(
    AcMapMbXmlFiler* pFiler
) const;
```

**Parameters**

- **pFiler**
  - Input filer to use to write the object's data.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Class AcMapMbTileGeneratorSettingsArea contains the added properties needed for area tile generation.

**class** AcMapMbTileGeneratorSettingsArea : **public** AcMapMbTileGenerator

**File**

AcMapMbTileGeneratorSettingsArea.h

**Remarks**

These are the start point and the end point.

**Methods**

- **~AcMapMbTileGeneratorSettingsArea**
  - Destroys an instance of this class.

- **AcMapMbTileGeneratorSettingsArea**
  - Constructs an instance of this class.

- **AllocateTileGenerator**
  - This method is used to allocate a tile generator that can be used by the settings object.

- **AreaFirstPoint**
  - Returns the first corner point of the area to tile.

- **AreaLastPoint**
  - Return the second or opposite corner point of the area to tile.

- **clone**
  - Clones the object. See also clone() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

  - Copies the contents of an object into the messaged object, if feasible. See also copyFrom() in the AutoCAD
**copyFrom**
ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

**dwgInFields**
Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

**dwgOutFields**
Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

**dxfInFields**
Lets this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

**dxfOutFields**
Lets this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

**GetAreaPoints**
Returns the extent points of the area to tile.

**GetExtents**
Gets the extents of the area to tile.

**SetAreaFirstPoint**
Sets the first corner point of the area to tile.

**SetAreaLastPoint**
Sets the second or opposite corner point of the area to tile.

**SetAreaPoints**
Sets the start point of the area to tile.

**SetExtents**
Sets the extents of the area to tile.
xmlInFields

Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

xmlOutFields

Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Let this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus xmlInFields(
    AcMapMbXmlFiler* pFiler
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pFiler</td>
<td>Input filer to use to read the object's data.</td>
</tr>
</tbody>
</table>

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
virtual Acad::ErrorStatus xmlOutFields(
    AcMapMbXmlFiler* pFiler
) const;

Parameters Description

pFiler Input filer to use to write the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
This is a list of values we handle pertinent to a Grid Tiler.

class AcMapMbTileGeneratorSettingsGrid : public AcMapMbTileGenerator

File

AcMapMbTileGeneratorSettingsGrid.h

Remarks

Start point NumberOfRows NumberOfColumns Class
AcMapMbTileGeneratorSettingsGrid contains the added properties needed for
grid tile generation. These are the start point, and the number of rows and
columns.

Methods

~AcMapMbTileGeneratorSettingsGrid Destroys an instance of this class.

AcMapMbTileGeneratorSettingsGrid Constructs an instance of this class.

AllocateTileGenerator This method is used to allocate a tile
generator that can be used by the
settings object.

AreaStartPoint Return the start point of the area to tile.

Clone the object. See also clone() in
the AutoCAD ObjectARX Developer's
Guide. This overridden function is
called by the system as needed; it is
unlikely that you will need to call it
directly.

copyFrom Copies the contents of an object into the
messaged object, if feasible. See also
copyFrom() in the AutoCAD
ObjectARX Developer's Guide. This
overridden function is called by the
Let this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

Let this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

Let this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

Let this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

Returns the integer value indicating number of columns.

Returns the integer value indicating number of rows.

Set the start point of the area to tile.

Sets the integer value indicating number of columns.

Sets the integer value indicating number of rows.

Let this object read its data. See also dwgInFields() in the AutoCAD
ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

xmlOutFields

Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus xmlInFields(
    AcMapMbXmlFiler* pFiler
);
```

**Parameters**

- **pFiler**: Input filer to use to read the object's data.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.toolsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbTileGeneratorSettingsGrid Class,
AcMapMbTileGeneratorSettingsGrid Class
AcMapMbTileGeneratorSettingsGrid::xmlOutFields Method
AcMapMbTileGeneratorSettingsGrid Class |
AcMapMbTileGeneratorSettingsGrid Class

Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus xmlOutFields(
    AcMapMbXmlFiler* pFiler
) const;
```

Parameters
- **pFiler**
  - Input filer to use to write the object's data.

Returns
- Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
This is a list of values we handle pertinent to a Manual Tiler Selection set of plines. Class `AcMapMbTileGeneratorSettingsManual` contains the added properties needed for manual tile generation.

```cpp
class AcMapMbTileGeneratorSettingsManual : public AcMapMbTileGeneratorSettings
```

File

`AcMapMbTileGeneratorSettingsManual.h`

Remarks

This is the selection set of plines.

Methods

- `~AcMapMbTileGeneratorSettingsManual` destroy an instance of this class.
- `AcMapMbTileGeneratorSettingsManual` constructs an instance of this class.
- `AllocateTileGenerator` this method is used to allocate a tile generator that can be used by the settings object.
- `clone` clones the object. See also `clone()` in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.
- `copyFrom` copies the contents of an object into the messaged object, if feasible. See also `copyFrom()` in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.
**dwgInFields**

Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

**dwgOutFields**

Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

**dxfInFields**

Lets this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

**dxfOutFields**

Lets this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

**GetSelectedTileBoundaries**

Returns the object ids of entities selected for tile boundaries.

**SetSelectedTileBoundaries**

Sets the selection set of entities selected.

**SetTileLayerName**

Overriding base class version.

**TileLayerName**

This is overridden from the base class since layer name is not appropriate for manual selection.

**xmlInFields**

Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the
system as needed; it is unlikely that you will need to call it directly.

Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

xmlOutFields

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbTileGeneratorSettingsManual Class,
AcMapMbTileGeneratorSettingsManual Class
AcMapMbTileGeneratorSettingsManual:: SetTileLayerName Method
AcMapMbTileGeneratorSettingsManual Class |
AcMapMbTileGeneratorSettingsManual Class

Overriding base class version.

```cpp
virtual Acad::ErrorStatus SetTileLayerName(
    const ACHAR* pszLayerName);
```

Parameters | Description
------------- |-------------
pszLayerName | Input Layer name.

Returns

Returns Acad::eNotApplicable.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapMbTileGeneratorSettingsManual Class,
AcMapMbTileGeneratorSettingsManual Class
AcMapMbTileGeneratorSettingsManual:: xmlInFields Method
AcMapMbTileGeneratorSettingsManual Class |
AcMapMbTileGeneratorSettingsManual Class

Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus xmlInFields(
        AcMapMbXmlFiler* pFiler
    );
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pFiler</td>
<td>Input filer to use to read the object's data.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbTileGeneratorSettingsManual Class,
AcMapMbTileGeneratorSettingsManual Class
AcMapMbTileGeneratorSettingsManual:: xmlOutFields Method
AcMapMbTileGeneratorSettingsManual Class | AcMapMbTileGeneratorSettingsManual Class

Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus xmlOutFields(
    AcMapMbXmlFiler* pFiler
) const;
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pFiler</td>
<td>Input filer to use to write the object's data.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Class `AcMapMbTileNameGenerator` is a pure virtual base class which defines the interface for the Name Generators.

```c++
class AcMapMbTileNameGenerator;
```

File

`AcMapMBTileNameGenerator.h`

Remarks

All Name Generators have the same interface and are closely linked with their respective Settings classes. The `TileNameGenerator` should not be instantiated directly, but must be obtained from the appropriate settings class. A settings class of the appropriate type must be passed back into the `GenerateTileNames` function.

Methods

- `~AcMapMbTileNameGenerator` Destroys an instance of this class.
- `AcMapMbTileNameGenerator` Constructs an instance of this class.
- `GenerateTileNames` Iterates through tile set and sets the names of individual tiles.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Class AcMapMbTileNameGeneratorData inherits from the pure virtual base class AcMapMbTileNameGenerator which defines the interface for the Name Generators.

```cpp
class AcMapMbTileNameGeneratorData : public AcMapMbTileNameGenerator
```

Remarks

All Name Generators have the same interface and are closely linked with their respective Settings classes. The TileNameGenerator should not be instantiated directly, but must be obtained from the appropriate settings class. A settings class of type AcMapMbTileNameGeneratorSettingsData must be passed back into the GenerateTileNames function.

Methods

- ~AcMapMbTileNameGeneratorData: Destroys an instance of this class.
- AcMapMbTileNameGeneratorData: Constructs an instance of this class.
- GenerateTileNames: Iterates through tile set and sets the names of individual tiles.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Class AcMapMbTileNameGeneratorGrid inherits from the pure virtual base class AcMapMbTileNameGenerator which defines the interface for the Name Generators.

class AcMapMbTileNameGeneratorGrid : public AcMapMbTileNameGenerator

File

AcMapMBTileNameGeneratorGrid.h

Remarks

All Name Generators have the same interface and are closely linked with their respective Settings classes. The TileNameGenerator should not be instantiated directly, but must be obtained from the appropriate settings class. A settings class of type AcMapMbTileNameGeneratorSettingsGrid must be passed back into the GenerateTileNames function.

Methods

~AcMapMbTileNameGeneratorGrid

Destroys an instance of this class.

AcMapMbTileNameGeneratorGrid

Constructs an instance of this class.

GenerateTileNames

Iterates through tile set and sets the names of individual tiles.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
This class encapsulates a naming nugget.

```cpp
class AcMapMbTileNameGeneratorIndexer;
```

Remarks

Basically it stores whether It's alphabetic or numeric, what the start index is and what the increment is. The NameGeneratorSettings classes will supply either one or 2 of these. A sequential naming will need one, and a Row Column naming scheme will need both a Row and a Column indexer. The class is read only in that once you have created it, you can only look at its properties. It is saved and restored from the dwg or dxf file containing the map book it is associated with using the StreamIn, StreamOut functions. These are not typically called directly.

Enumerations

- `EIndexOrientation` This is record `AcMapMbTileNameGeneratorIndexer::EIndexOrientation`.

Methods

- `~AcMapMbTileNameGeneratorIndexer` Destroys an instance of this class.
- `AcMapMbTileNameGeneratorIndexer` Constructs an instance of this class.
- `Increment` Retrieves the increment value for the index.
- `Index` Retrieves the starting Index.
- `IsInverted` Retrieves whether the indexing should start at the top or front or at the back or bottom.
- `IsNumeric` Retrieves whether the indexing is numeric or alphabetic.
**Orientation**

Retrieves the orientation or type of Index.

**StreamIn**

This is StreamIn, a member of class AcMapMbTileNameGeneratorIndexer.

**StreamIn**

This is StreamIn, a member of class AcMapMbTileNameGeneratorIndexer.

**StreamOut**

This is StreamOut, a member of class AcMapMbTileNameGeneratorIndexer.

**StreamOut**

This is StreamOut, a member of class AcMapMbTileNameGeneratorIndexer.

**StreamOut**

This is StreamOut, a member of class AcMapMbTileNameGeneratorIndexer.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links

AcMapMbTileNameGeneratorIndexer Class,
AcMapMbTileNameGeneratorIndexer Class
AcMapMbTileNameGeneratorIndexer:: StreamIn Method
AcMapMbTileNameGeneratorIndexer Class |
AcMapMbTileNameGeneratorIndexer Class

This is StreamIn, a member of class AcMapMbTileNameGeneratorIndexer.

virtual Acad::ErrorStatus StreamIn(
   AcMapMbXmlFiler* pFiler
);

Created with a commercial version of Doc-O-Matic. In order to make this
message disappear you need to register this software. If you have problems
registering this software please contact us at support@toolsfactory.com.
This is StreamOut, a member of class AcMapMbTileNameGeneratorIndexer.

```cpp
virtual Acad::ErrorStatus StreamOut(
    AcMapMbXmlFiler* pFiler
) const;
```

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Class AcMapMbTileNameGeneratorSequence inherits from the pure virtual base class AcMapMbTileNameGenerator which defines the interface for the Name Generators.

class AcMapMbTileNameGeneratorSequence : public AcMapMbTileNameGenerator

File

AcMapMBTileNameGeneratorSequence.h

Remarks

All Name Generators have the same interface and are closely linked with their respective Settings classes. The TileNameGenerator should not be instantiated directly, but must be obtained from the appropriate settings class. A settings class of type AcMapMbTileNameGeneratorSettingsSequence must be passed back into the GenerateTileNames function.

Methods

~AcMapMbTileNameGeneratorSequence Destroys an instance of this class.

AcMapMbTileNameGeneratorSequence Constructs an instance of this class.

GenerateTileNames Iterates through tile set and sets the names of individual tiles.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Class AcMapMbTileNameGeneratorSettings is the base settings class for name generation.

class AcMapMbTileNameGeneratorSettings : public AcDbObject;

Remarks

It contains the properties needed by all derived classes. The user will not instantiate a member of this class as the derived classes have some properties needed by actual naming schemes. The Name Generators and the Name Generator Settings classes have a direct relationship. For any future expansion, users will typically create both a new Generator and Settings class. The appropriate Generator will be obtained from the Settings class. The DECLARE_MEMBERS macro is used in the declaration of classes that are to be a part of the ObjectARX run-time tree.

Enumerations

- ETileNamingScheme

Methods

- ~AcMapMbTileNameGeneratorSettings
- AcMapMbTileNameGeneratorSettings
- AllocateTileNameGenerator
- copyFrom

Destroys an instance of this class.

Constructs an instance of this class.

Returns the associated Name Generator.

Copies the contents of an object into the messaged object, if feasible. See also copyFrom() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the
CountDisabledTiles

Returns the count disable tiles data member.

Let's this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

dwgInFields

Let's this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

dwgOutFields

dxfInFields

Let's this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

dxfOutFields

GetAlphaSequence

Returns the implementation object.

Implementation

SetCountDisabledTiles

Sets the count disable tiles data member.

SetTileNamingScheme

Returns an enum which specifies the tile naming scheme.

TileNamingScheme

Returns an enum which specifies the tile naming scheme.

Let's this object read its data. See also
\textbf{xmlInFields}

dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

\textbf{xmlOutFields}

Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

Created with a commercial version of \texttt{Doc-O-Matic}. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at \texttt{support@toolsfactory.com}. 
static Acad::ErrorStatus GetAlphaSequence(
  ACHAR*& pszAlphaSequence
);

Parameters | Description
--- | ---
pszAlphaSequence |

Returns

Returns the implementation, or NULL if unsuccessful.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links

AcMapMbTileNameGeneratorSettings Class, AcMapMbTileNameGeneratorSettings Class
AcMapMbTileNameGeneratorSettings:: xmlInFields Method

AcMapMbTileNameGeneratorSettings Class | AcMapMbTileNameGeneratorSettings Class

Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

```
virtual Acad::ErrorStatus xmlInFields(
   AcMapMbXmlFiler* pFiler
);
```

Parameters          Description
pFiler              Input filer to use to read the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbTileNameGeneratorSettings Class,
AcMapMbTileNameGeneratorSettings Class
AcMapMbTileNameGeneratorSettings:: xmlOutFields Method
AcMapMbTileNameGeneratorSettings Class |
AcMapMbTileNameGeneratorSettings Class

 Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

 virtual Acad::ErrorStatus xmlOutFields(  
    AcMapMbXmlFiler* pFiler
 ) const;

 Parameters | Description
---------- | ------------
pFiler | Input filer to use to write the object's data.

 Returns

 Returns Acad::eOk if successful; otherwise, returns a different error code.

 Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Class AcMapMbTileNameGeneratorSettingsData contains the added property needed for data name generation.

class AcMapMbTileNameGeneratorSettingsData : public AcMapMbTileNameG

Remarks

This is the expression needed to get the tile name from the reference object for the tile. The reference objects will typically be rectangular polygons with object data or some other property supplying the tile name.

Methods

~AcMapMbTileNameGeneratorSettingsData
Destroys an instance of this class.

AcMapMbTileNameGeneratorSettingsData
Constructs an instance of this class.

AllocateTileNameGenerator
Returns the associated Name Generator.

clone
Clones the object. See also clone() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

copyFrom
Copies the contents of an object into the messaged object, if feasible. See also copyFrom() in the AutoCAD ObjectARX Developer's Guide. This
overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

Let's this object read its data. See also `dwgInFields()` in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

Let's this object write its data. See also `dwgOutFields()` in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

Let's this object read its data. See also `dxfInFields()` in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

Let's this object write its data. See also `dxfOutFields()` in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

Retrieves the expression to be applied to the reference object.

**dwxInFields**

**dwxOutFields**

**dxFInFields**

**dxFOutFields**

**Expression**
SetExpression

Sets the expression to be applied to the reference object.

xmlInFields

Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

xmlOutFields

Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the expression to be applied to the reference object.

```cpp
Acad::ErrorStatus SetExpression(const ACHAR* pszExpression);
```

**Parameters**

<table>
<thead>
<tr>
<th>pszExpression</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Variable to set</td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus xmlInFields(
    AcMapMbXmlFiler* pFiler
);
```

**Parameters**

- **pFiler**
  - Description: Input filer to use to read the object's data.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus xmlOutFields(
    AcMapMbXmlFiler* pFiler
) const;
```

**Parameters**
- **pFiler** Input filer to use to write the object's data.

**Returns**
- Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Class AcMapMbTileNameGeneratorSettingsGrid contains the added properties needed for grid name generation.

class AcMapMbTileNameGeneratorSettingsGrid : public AcMapMbTileNameGenerator

File

AcMapMbTileNameGeneratorSettingsGrid.h

Remarks

These are the primary index, the secondary index, and the separator string. The primary index could be either row or column, and the secondary index will be the other type. The primary index comes first in the tile name, followed by the separator string, followed by the secondary index.

- Enumerations
  - EGridNamingOrder This enum indicates the naming order for the grid.

- Methods
  - ~AcMapMbTileNameGeneratorSettingsGrid Destroys an instance of this class.
  - AcMapMbTileNameGeneratorSettingsGrid Constructs an instance of this class.
  - AllocateTileNameGenerator Returns the associated Name Generator.
  - clone Clones the object. See also clone() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.
  - Copies the contents of an object into the messaged object, if
copyFrom

feasible. See also copyFrom() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

Let this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

dwgInFields

dwgOutFields

Let this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

dxfInFields

Let this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

dxfOutFields

Let this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.
**GetPrimaryIndex**
Returns the primary indexer.

**GetSecondaryIndex**
Returns the column index data mamber.

**Separator**
Returns the seperator data mamber.

**SetPrimaryIndex**
Sets the primary indexer.

**SetSecondaryIndex**
Sets the primary indexer.

**SetSeparator**
Sets the seperator data mamber.

**xmlInFields**
Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

**xmlOutFields**
Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the separator data member.

\[
\text{Acad::ErrorStatus SetSeparator(}
\text{    \textbf{const} ACHAR* pszSeperator}
\text{);}\]

Parameters | Description
---|---
sSeparator | Variable to hold value set

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
virtual Acad::ErrorStatus xmlInFields(
    AcMapMbXmlFiler* pFiler
);

Parameters Description
pFiler Input filer to use to read the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus xmlOutFields(
    AcMapMbXmlFiler* pFiler
) const;
```

Parameters

- `pFiler`: Input filer to use to write the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Class AcMapMbTileNameGeneratorSettingsSequence contains the added property needed for sequential name generation.

```cpp
class AcMapMbTileNameGeneratorSettingsSequence : public AcMapMbTileN
```

File

AcMapMbTileNameGeneratorSettingsSequence.h

Remarks

This is the sequential index.

Methods

- `~AcMapMbTileNameGeneratorSettingsSequence` - Destroys an instance of this class.
- `AcMapMbTileNameGeneratorSettingsSequence` - Constructs an instance of this class.
- `AllocateTileNameGenerator` - Returns the associated Name Generator.
- `clone` - Clones the object. See also clone() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.
- `copyFrom` - Copies the contents of an object into the messaged object, if feasible. See also copyFrom() in the AutoCAD ObjectARX Developer's Guide. This overridden
function is called by the system as needed; it is unlikely that you will need to call it directly.

Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

Lets this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

Lets this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

Retrieves the sequence
**GetIndex**

Sets the primary index.

**SetIndex**

Lets this object read its data. See also `dwgInFields()` in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

**xmlInFields**

Lets this object write its data. See also `dwgOutFields()` in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Let this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus xmlInFields(
    AcMapMbXmlFiler* pFiler
);
```

**Parameters**

- `pFiler`: Input filer to use to read the object's data.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
virtual Acad::ErrorStatus xmlOutFields(
    AcMapMbXmlFiler* pFiler
) const;

Parameters               Description
pFiler                   Input filer to use to write the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Provides functions to create, name, and return the tiles used by a MapBook.

```cpp
class AcMapMbTileSet;
```

File

AcMapMbTileSet.h

Remarks

The UI can display the tiles in several ways. Numeric sequences are displayed differently from alphabetic sequences, and there may be differences between an Alpha sequence and an alphabetized sequence. If the tiles are returned as a sequence, then the root will contain a single child node, and this node will have several siblings. None of the siblings will have any children. If the requested return is RowCol, then.

- **Enumerations**
  - **EMbNameSchema** This is record AcMapMbTileSet::EMbNameSchema.
  - **EMbTreeType** This is record AcMapMbTileSet::EMbTreeType.

- **Classes**
  - **AcMapContainer** Alternate interface for tree.
    This is class AcMapMbTileSet::AcMapContainer.
  - **AcMapContainerIterator** This is class AcMapMbTileSet::AcMapContainerIterator.
  - **AcMapTileIterator** This is class AcMapMbTileSet::AcMapTileIterator.
  - **AcMapTreeNode** Class AcMapTreeNode models the most general structure for an organization of tiles.

- **Methods**
  - **~AcMapMbTileSet** Destroys an instance of this class.
  - **AcMapMbTileSet** Constructs an instance of this class.
  - **AddTiles** Sets the new tiles into the tile set.
  - **BuildAdjacentInfo** Generate adjacency information and assigns it to every tile.
**ColumnNumber**
Returns the number of columns in a grid tiling.

**ConstContainer**
Returns the root of the tree organized according to the tree type as a container.

**ConstTree**
Returns the root of the tree organized according to the tree type.

**CreateGridLayer**
Sets the layer on which the grid polygons will be created.

**CreateNode**
Creates a new node for the tree.

**Database**
Returns the pointer to the drawing database the tile set is associated with.

**Erase**
Removes the specified tile from the tile set.

**Find**
Returns the tile corresponding to the given tile id if it exists.

**GetExtents**
Returns the total extents of the view ports of this tile set.

**Implementation**
Returns the implementation object.

**LayerId**
When a tile set gets generated (except manual), it creates polylines per every tile and puts them to the specified layer.

**MapBook**
Returns the Id of the AcMapMbMapBook object it belongs to.

**NamingScheme**
Returns the way or schema used when naming the tiles.

**RowNumber**
Returns the number of rows in a grid tiling.

**SetAdjacentProperties**
Iterates a Tile Set and sets adjacent properties for the corresponding Sheets.

**StreamIn**
Reads data from a dwg file.

**StreamIn**
Reads data from a dxf file.

**StreamOut**
Loads data into a dwg file.

**StreamOut**
Loads data into a dxf file.

**TileNumber**
Returns the total number of tiles in the set.

**TilingScheme**
Returns the schema or organization used when creating the tiles.

**Tree**
Returns the root of the tree organized according to the tree type.
Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the layer on which the grid polygons will be created.

```c
Acad::ErrorStatus CreateGridLayer(
    const ACHAR* pszName
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pszName</td>
<td>Input name of the layer to create polygons on. This must be a valid extended symbol name.</td>
</tr>
<tr>
<td>pDb</td>
<td>Input Acad::Database where the polygons will be drawn.</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

**Remarks**

The database is needed to indicate where to create them.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Map Book Global-Function API (Deprecated)

The global-function API for plotting map books is deprecated in favor of the [class-based API](#).

<table>
<thead>
<tr>
<th>Map Book Function Catalog</th>
<th>Map Book functions sorted by name.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Map Book Function Synopsis</strong></td>
<td>Map Book functions sorted by functional group.</td>
</tr>
</tbody>
</table>
Map Book functions sorted by name.

**Note** The global-function API for plotting map books is deprecated in favor of the class-based API.

dwg | plt | topo

map_dwgbreakobj map_dwgtrimobj

map_pltblkatts
map_pltblklist
map_pltblkyps
map_pltcleanup
map_pltcurrdef
map_pltcurreldel
map_pltcurret
map_pltcursave
map_pltcursset
map_pltdefdelete
map_pltdefget
map_pltdeflist
map_pltdefread
map_pltdefsave
map_pltdefvalid
map_pltdefverify
map_pltdisplay
map_pltexecute
map_pltininit
map_pltplot
map_pltrestore

map_topoaudit
map_topoclose
map_topocomplete
map_topostat
Map Book Function Synopsis

Map Book Global-Function API

Map Book functions sorted by functional group.

**Note** The global-function API for plotting map books is deprecated in favor of the [class-based API](#).

[Boundary Functions](#) [Plotting Functions](#) [Topology Functions](#)
Object Filter

Filters objects (entities) in the current drawing based on layer, feature-class, and block criteria. Filters are constructed with "*" as the default filtering value, indicating that all entities are to be filtered. Other valid values are "" (an empty string) to ignore entities during filtering, and NULL (a null filter) to exclude any entities returned by the filter set. The default AcDbBasicFilter object is constructed with * as the value for the Layers, FeatureClasses, and Blocks filtering criteria, and so will filter out classified blocks from all layers. The following table shows how to set entity-filtering criteria for various filtering scenarios. (Note that these scenarios do not include objects on frozen, locked, or off layers unless you have set the layer status mask with SetLayerStatusMask()).

Base class for creating filters that filter objects (entities) in the current drawing based on the specified criteria.

Filters objects (entities) in the current drawing based on the criteria of one or more listed filters.
Filters objects (entities) in the current drawing based on layer, feature-class, and block criteria. Filters are constructed with "*" as the default filtering value, indicating that all entities are to be filtered. Other valid values are "" (an empty string) to ignore entities during filtering, and NULL (a null filter) to exclude any entities returned by the filter set. The default AcDbBasicFilter object is constructed with * as the value for the Layers, FeatureClasses, and Blocks filtering criteria, and so will filter out classified blocks from all layers. The following table shows how to set entity-filtering criteria for various filtering...

**AcDbBasicFilter**

Base class for creating filters that filter objects (entities) in the current drawing based on the specified criteria. Use this class to derive custom filters if you want to filter objects on criteria other than layers, feature classes, and blocks; otherwise, use the AcDbBasicFilter class. Any class that you derive from this class must implement FilterObjects() to define the filtering criteria. To use multiple filters, use the AcDbObjectFilterGroup class.

**AcDbObjectFilter**

Filters objects (entities) in the current drawing based on the criteria of one or more listed filters. To create a filter, construct an instance of the AcDbBasicFilter class or an instance of a class derived from the AcDbObjectFilter class.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Filters objects (entities) in the current drawing based on layer, feature-class, and block criteria.

Filters are constructed with "*" as the default filtering value, indicating that all entities are to be filtered. Other valid values are "" (an empty string) to ignore entities during filtering, and NULL (a null filter) to exclude any entities returned by the filter set. The default AcDbBasicFilter object is constructed with * as the value for the Layers, FeatureClasses, and Blocks filtering criteria, and so will filter out classified blocks from all layers. The following table shows how to set entity-filtering criteria for various filtering scenarios. (Note that these scenarios do not include objects on frozen, locked, or off layers unless you have set the layer status mask with SetLayerStatusMask().)

<table>
<thead>
<tr>
<th></th>
<th>Layers</th>
<th>FeatureClasses</th>
<th>Blocks</th>
</tr>
</thead>
<tbody>
<tr>
<td>All objects from all layers</td>
<td>*</td>
<td>&quot;&quot;</td>
<td>&quot;&quot;</td>
</tr>
<tr>
<td>All classified objects from all layers</td>
<td>*</td>
<td>*</td>
<td>&quot;&quot;</td>
</tr>
<tr>
<td>All blocks (inserts) from all layers</td>
<td>*</td>
<td>&quot;&quot;</td>
<td>*</td>
</tr>
<tr>
<td>Classified blocks from all layers</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>Classified objects from specified layers</td>
<td>Layer1,Layer5</td>
<td>*</td>
<td>&quot;&quot;</td>
</tr>
<tr>
<td>Blocks from specified layers</td>
<td>Layer1,Layer5</td>
<td>&quot;&quot;</td>
<td>*</td>
</tr>
<tr>
<td>Specified classified objects from all layers</td>
<td>*</td>
<td>Road,Lake</td>
<td>&quot;&quot;</td>
</tr>
<tr>
<td>Specified blocks from all layers</td>
<td>*</td>
<td>&quot;&quot;</td>
<td>Block1,Block5</td>
</tr>
<tr>
<td>Description</td>
<td>Filter Type(s)</td>
<td>Example</td>
<td></td>
</tr>
<tr>
<td>----------------------------------------------------------------------------</td>
<td>----------------------</td>
<td>---------</td>
<td></td>
</tr>
<tr>
<td>and specified blocks from all layers</td>
<td>*</td>
<td>Road,Lake Block1,Block5</td>
<td></td>
</tr>
<tr>
<td>Specified classified objects from specified layers</td>
<td>Layer1,Layer5</td>
<td>Road,Lake &quot;&quot;</td>
<td></td>
</tr>
<tr>
<td>Specified blocks from specified layers</td>
<td>Layer1,Layer5</td>
<td>&quot;&quot;</td>
<td></td>
</tr>
<tr>
<td>Specified classified objects and specified blocks from specified layers</td>
<td>Layer1,Layer5</td>
<td>Road,Lake Block1,Block5</td>
<td></td>
</tr>
<tr>
<td>All objects from all layers except classified objects</td>
<td>*</td>
<td>NULL &quot;&quot;</td>
<td></td>
</tr>
<tr>
<td>All objects from all layers except blocks</td>
<td>*</td>
<td>&quot;&quot;</td>
<td></td>
</tr>
<tr>
<td>All objects from all layers except classified objects and blocks</td>
<td>*</td>
<td>NULL NULL</td>
<td></td>
</tr>
<tr>
<td>All objects from specified layers except classified objects</td>
<td>Layer1,Layer5</td>
<td>NULL &quot;&quot;</td>
<td></td>
</tr>
<tr>
<td>All objects from specified layers except blocks</td>
<td>Layer1,Layer5</td>
<td>&quot;&quot;</td>
<td></td>
</tr>
</tbody>
</table>

To use multiple filters, see the class `AcDbObjectFilterGroup`.

```cpp
class AcDbBasicFilter : public AcDbObjectFilter;
```

File

`AcDbObjectFilter.h`

- **Enumerations**
  - `ELayerStatus` Enumerates the types of layer-status filters.

- **Methods**
  - `~AcDbBasicFilter` Destroys an instance of this class.
AcDbBasicFilter

Constructs an instance of this class with the default value filter values set to "*" to filter all entities.

Constructs an instance of this class by using the specified layer, feature-class, and block filtering criteria.

AddBlocks

Adds a block to the list of blocks on which to filter objects.

AddFeatureClasses

Adds a feature class to the list of feature classes on which to filter objects.

AddLayers

Adds a layer to the list of layers on which to filter objects.

Blocks

Retrieves the block(s) on which to filter. The block filter is defined with SetBlocks() (twoforms).

FeatureClasses

Retrieves the feature class(es) on which to filter. The feature-class filter is defined with SetFeatureClasses() (twoforms).

FilterObjects

Filters objects on the current drawing based on the layer, feature-class, and block criteria.

A filtering criterion can take the following values: "*" - (Default) Filters all entities that reside on layers with normal layer status. Unless SetLayerStatusMask() has been called with specific layer mask settings, all objects on layers which are not frozen, off, or locked are considered in the filtering process. "" (empty string) - Indicates that a particular filter is to be ignored during the filtering process. NULL - Indicates that particular filtered entities are to be excluded from the filtered set.

A name - The name of...

Layers

Retrieves the layer(s) on which to filter. The layer filter is defined with SetLayers() (twoforms). NULL and empty strings (""") are invalid values.

LayerStatusMask

Determines whether the layer status mask has been set by using SetLayerStatusMask(). A nonzero return value indicates that objects on layers which are frozen, locked, or off are to be considered in the filtering process. A zero return value indicates that no layer mask is set and filtering will be applied to objects with
<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ResetBlocks</strong></td>
<td>Clears the filter list of blocks.</td>
</tr>
<tr>
<td><strong>ResetFeatureClasses</strong></td>
<td>Clears the filter list of feature classes.</td>
</tr>
<tr>
<td><strong>ResetLayers</strong></td>
<td>Clears the list of filter layers.</td>
</tr>
<tr>
<td><strong>SetBlocks</strong></td>
<td>Sets the block filters. See also Blocks().</td>
</tr>
<tr>
<td><strong>SetLayers</strong></td>
<td>Sets the layers on which to filter objects. See also Layers().</td>
</tr>
<tr>
<td><strong>SetFeatureClasses</strong></td>
<td>Sets the feature classes on which to filter objects. See also FeatureClasses().</td>
</tr>
<tr>
<td><strong>SetLayerStatusMask</strong></td>
<td>Sets the layer-status filter criterion that specifies whether objects on frozen, locked, or off layers are filtered. Use LayerStatusMask() to determine whether the layer-status filter is set.</td>
</tr>
</tbody>
</table>

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Constructs an instance of this class by using the specified layer, feature-class, and block filtering criteria.

```cpp
AcDbBasicFilter(
    const ACHAR* layerFilter,
    const ACHAR* featureClassFilter,
    const ACHAR* blockfilter
);
```

### Parameters

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>layerFilter</strong></td>
</tr>
<tr>
<td>Input layer name(s). Separate multiple names by commas. Layer names are case-insensitive.</td>
</tr>
<tr>
<td><strong>featureClassFilter</strong></td>
</tr>
<tr>
<td>Input feature-class name(s). Separate multiple names by commas.</td>
</tr>
<tr>
<td><strong>blockfilter</strong></td>
</tr>
<tr>
<td>Input block names. Separate multiple names by commas.</td>
</tr>
</tbody>
</table>

### Returns

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Add a block to the list of blocks on which to filter objects.

```
Acad::ErrorStatus AddBlocks(
    const ACHAR* blockFilter
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>blockFilter</td>
<td>Input name of the block to add to the list.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::ErrorStatus eOk if successful. Returns Acad::ErrorStatus eInvalidInput if parameter validation failed.

Created with a commercial version of [Doc-O-Matic](https://toolsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Add a feature class to the list of feature classes on which to filter objects.

```cpp
Acad::ErrorStatus AddFeatureClasses(
    const ACHAR* featureClassFilter
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>featureClassFilter</td>
<td>Input name of the feature class to add to the list.</td>
</tr>
</tbody>
</table>

Returns Acad::ErrorStatus eOk if successful. Returns Acad::ErrorStatus eInvalidInput if parameter validation failed.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcDbBasicFilter Class, AcDbBasicFilter Class
AcDbBasicFilter:: AddLayers Method
AcDbBasicFilter Class | AcDbBasicFilter Class

Adds a layer to the list of layers on which to filter objects.

```c++
Acad::ErrorStatus AddLayers(
    const ACHAR* layerFilter
);
```

Parameters

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input name of the layer to add to the list. Layer names are case-insensitive for filtering operations but are displayed in their original case.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::ErrorStatus eOk if successful. Returns Acad::ErrorStatus eInvalidInput if parameter validation failed.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the block filters. See also Blocks().

`Acad::ErrorStatus SetBlocks(`

`AcArray<ACHAR*>& blockFilters`

`);`

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>blockFilters</td>
<td>Input array of block names.</td>
</tr>
</tbody>
</table>

Returns `Acad::ErrorStatus eOk` if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the blocks on which to filter objects. See also Blocks().

```cpp
Acad::ErrorStatus SetBlocks(
    const ACHAR* blockFilter
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input block name(s) in any of the following forms: A single block name (&quot;Block1&quot;, for example) A list of comma-separated block names (&quot;Block1,Block2&quot;, for example) A &quot;*&quot; to filter all blocks (this is the default setting) A &quot;&quot; (empty string) to ignore blocks during filtering A NULL to exclude blocks from the filtered setBlock names with wildcard characters (&quot;Block?&quot; or &quot;Block- *&quot;, for example)</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::ErrorStatus eOk if successful. Returns Acad::ErrorStatus eInvalidInput if parameter validation failed. Returns Acad::ErrorStatus eInvalidInput if the block list cannot be obtained (applies only if blockFilter is "*")

Created with a commercial version of [Doc-O-Matic](https://www.docsomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
AcDbBasicFilter:: SetFeatureClasses Method

Returns Acad::ErrorStatus eOk if successful. Returns Acad::ErrorStatus eInvalidInput if parameter validation failed. Returns Acad::ErrorStatus eInvalidInput if the feature-class list cannot be obtained (applies only if featureClassFilter is "*")

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the feature classes on which to filter objects. See also FeatureClasses().

```cpp
Acad::ErrorStatus SetFeatureClasses(
    const AcArray<ACHAR*>& featureClassFilters
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>featureClassFilters</td>
<td>Input array of feature-class names.</td>
</tr>
</tbody>
</table>

Returns Acad::ErrorStatus eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the layers on which to filter objects. NULL and empty strings ("") are invalid values. See also Layers().

```cpp
Acad::ErrorStatus SetLayers(const ACHAR* layerFilter);
```

**Parameters**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input layer name(s). Separate multiple names by commas. If the layer filter is &quot;,*&quot;, and both block and feature-class filters are &quot;&quot; (ignore during filtering), then objects on all layers are filtered. Layer names are case-insensitive.</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::ErrorStatus eOk if successful. Returns Acad::ErrorStatus eInvalidInput if parameter validation failed.
Sets the layers on which to filter objects. See also Layers().

\begin{verbatim}
Acad::ErrorStatus SetLayers(
    const AcArray<ACHAR*>& layerFilters
);
\end{verbatim}

Parameters

layerFilters

Input array of layer names. Layer names are case-insensitive.

Returns

Returns Acad::ErrorStatus eOk if successful; otherwise, returns a different error code.

Created with a commercial version of \textit{Doc-O-Matic}. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at \texttt{support@toolsfactory.com}.  

Links

AcDbBasicFilter Class, AcDbBasicFilter Class
AcDbBasicFilter:: SetLayers Method
AcDbBasicFilter Class | AcDbBasicFilter Class
Base class for creating filters that filter objects (entities) in the current drawing based on the specified criteria. Use this class to derive custom filters if you want to filter objects on criteria other than layers, feature classes, and blocks; otherwise, use the AcDbBasicFilter class. Any class that you derive from this class must implement FilterObjects() to define the filtering criteria. To use multiple filters, use the AcDbObjectFilterGroup class.

```cpp
class AcDbObjectFilter;
```

### File

AcDbObjectFilter.h

#### Methods

- **~AcDbObjectFilter**  
  Destroys an instance of this class.
- **AcDbObjectFilter**  
  Constructs an instance of this class.
- **FilterObjects**  
  Filters objects on the current drawing based on the filtering criteria.
- **IsActive**  
  Determines whether this filter is active.
  Activates or deactivates this filter. When this filter is active, objects are filtered based on the filtering criteria when FilterObjects() in the derived class is called.
- **SetActive**  
  When this filter is inactive, no objects are filtered. This setting applies only when the AcDbObjectFilterGroup class is used to filter objects.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Filters objects (entities) in the current drawing based on the criteria of one or more listed filters. To create a filter, construct an instance of the AcDbBasicFilter class or an instance of a class derived from the AcDbObjectFilter class.

class AcDbObjectFilterGroup : public AcDbObjectFilter;

File
AcDbObjectFilter.h

Methods

~AcDbObjectFilterGroup
Destroys an instance of this class.

AcDbObjectFilterGroup
Constructs an instance of this class.

AddObjectFilter
Adds a filter to the end of the filter list.

FilterObjects
Filters objects on the current drawing based on the filtering criteria of all the filters in the filter list.

GetObjectFilter
Retrieves a filter from the filter list.

InsertObjectFilter
Inserts a filter into the filter list.

IsEmpty
Determines whether the filter list is empty.

ObjectFilterCount
Counts the number of filters in the filter list.

RemoveAllObjectFilter
Clears the filter list.

RemoveObjectFilter
Removes a filter from the filter list.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
## Oracle Spatial Advanced

### Functions

<table>
<thead>
<tr>
<th>Class Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AcMapOSEGetConnection</td>
<td>Gets the connection object.</td>
</tr>
</tbody>
</table>

### Classes

<table>
<thead>
<tr>
<th>Class Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>AcMapOSEConnection</td>
</tr>
<tr>
<td>AcMapOSEConnectionReactor</td>
</tr>
<tr>
<td>AcMapOSEExport</td>
</tr>
<tr>
<td>AcMapOSEExportReactor</td>
</tr>
<tr>
<td>AcMapOSEImport</td>
</tr>
<tr>
<td>AcMapOSEImportReactor</td>
</tr>
<tr>
<td>AcMapOSEObject</td>
</tr>
<tr>
<td>AcMapOSEProject</td>
</tr>
<tr>
<td>AcMapOSEQuery</td>
</tr>
</tbody>
</table>
## Oracle Spatial (Deprecated)

### Functions

<table>
<thead>
<tr>
<th>Class Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AcMapOracleGetConnection</td>
<td>Gets the connection object.</td>
</tr>
</tbody>
</table>

### Classes

<table>
<thead>
<tr>
<th>Class Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>AcMapOracleConnection</td>
</tr>
<tr>
<td>AcMapOracleConnectionReactor</td>
</tr>
<tr>
<td>AcMapOracleExport</td>
</tr>
<tr>
<td>AcMapOracleExportReactor</td>
</tr>
<tr>
<td>AcMapOracleIdentification</td>
</tr>
<tr>
<td>AcMapOracleImport</td>
</tr>
<tr>
<td>AcMapOracleImportReactor</td>
</tr>
<tr>
<td>AcMapOracleQuery</td>
</tr>
</tbody>
</table>
Links

**Structs, Records, Enums**

Topology: Class-Based API
**Structs, Records, Enums**

- **Namespaces**
  - **AcMapTopologyManager** The topology manager object.

- **Types**
  - **AcMapFullEdgePtrArray** Dynamic array of
  - **AcMapHalfEdgePtrArray** Dynamic array of
  - **AcMapNodePtrArray** Dynamic array of
  - **AcMapObjectPtrArray** Dynamic array of
  - **AcMapOverlayDataArray** This is type AcMapOverlayDataArray.
  - **AcMapPolygonPtrArray** Dynamic array of
  - **AcMapRingPtrArray** Dynamic array of
  - **AcMapTopologySourceArray** This is type AcMapTopologySourceArray.

- **Classes**
  - **AcMapEntityCreationSettings** Settings for creating entities.
  - **AcMapFloodParameters** Settings for flood parameters.
  - **AcMapMarkerStyles** Settings for error markers.
  - **AcMapNetAnalysisParameters** Settings for analysis parameters.
  - **AcMapPointCreationSettings** Settings for creating points.
  - **AcMapTopoElement** Manages topology objects.
  - **AcMapTopoElementPtrArray** Template for dynamic arrays of topology element pointers.
  - **AcMapTopoFullEdge** Manages topology full edges.
  - **AcMapTopoHalfEdge** Manages topology half edges.
  - **AcMapTopoIterator** An iterator over a collection of
  - **AcMapTopology** The main topology object.
  - **AcMapTopologySource** Stores the topology source information. For more information, search for *topology* in AutoCAD Map Help.
  - **AcMapTopoNode** Manages topology nodes.
  - **AcMapTopoOverlayData** For topology overlay analysis, use this class to set the data attributes that contain object data from linked database to include in the resulting
topology, and the name and description of the
new object data table.

AcMapTopoPolygon
Manages topology polygons.

AcMapTopoRing
Manages topology rings.

AcMapTraceParameters
Settings for trace parameters.

Structs, Records, Enums

AcMapObjectDataField
Information about a field in a topology object data
table.

AcMapObjectDataTable
Information about a topology object data table.

ETopologyMarkType
Topology error-marker shapes.

ETopologyType
Topology types.

Links
Structs, Records, Enums
The topology manager object. For more information, search for topology in AutoCAD Map Help.

- **Enumerations**
  - **ETopologyScope**: Enumerates the scope of a topology.

- **Functions**
  - **Delete**: Deletes the specified topology from the current drawing and all source drawings.
  - **GetTopologyScope**: Retrieves the scope of the specified topology. The returned value is cast to an ETopologyScope value to indicate whether the specified topology exists on the current drawing, source drawing, or both.
  - **GetTopologySource**: Retrieves the source information of the specified topology.
  - **Rename**: Renames the existing topology.
  - **TopologyExists**: Determines whether a specified topology exists in current or source (attached) drawing.
  - **TopologyExists**: Determines whether a specified topology exists in specified source.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Enumerates the scope of a topology.

```cpp
enum ETopologyScope {
    eCurrentDwg = 0x0001,
    eSourceDwg = 0x0002
};
```

File

AcMapTopologyManager.h

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>eCurrentDwg</td>
<td>The topology is in the current drawing.</td>
</tr>
<tr>
<td>eSourceDwg</td>
<td>The topology is in the source (attached) drawing.</td>
</tr>
</tbody>
</table>

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Deletes the specified topology from the current drawing and all source drawings.

```cpp
AcMap::EErrCode Delete(
    const ACHAR* pszName,
    bool bDelEntities
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pszName</td>
<td>Input name of the topology to delete.</td>
</tr>
<tr>
<td>bDelEntities</td>
<td>Input true to delete all the entities; otherwise, false.</td>
</tr>
</tbody>
</table>

**Returns**

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

**Remarks**

You must close a topology before you delete it.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the scope of the specified topology. The returned value is cast to an ETopologyScope value to indicate whether the specified topology exists on the current drawing, source drawing, or both.

```cpp
int GetTopologyScope(
    const ACHAR* pszName
);
```

**File**

AcMapTopologyManager.h

**Parameters**

<table>
<thead>
<tr>
<th>pszName</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input name of a the topology to check.</td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

Returns eCurrentDwg if the topology exists on current drawing. Returns eSourceDwg if the topology exists on source drawing. Returns eCurrentDwg | eSourceDwg if the topology exists on both current and source drawings. Returns 0 if the specified topology does not exist.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the source information of the specified topology.

```c
AcMap::EErrCode GetTopologySource(
    AcMapTopologySourceArray& apSource,
    const ACHAR* pszName
);
```

File

`AcMapTopologyManager.h`

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>apSource</td>
<td>Output array of topology source information.</td>
</tr>
<tr>
<td>pszName</td>
<td>Input name of the topology.</td>
</tr>
</tbody>
</table>

Returns

Returns `AcMap::EErrCode kOk` if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapTopologyManager Namespace
AcMapTopologyManager:: Rename Function
AcMapTopologyManager Namespace

Renames the existing topology.

AcMap::EErrCode Rename(
    const ACHAR* pszOldName,
    const ACHAR* pszNewName
);

File
AcMapTopologyManager.h

Parameters | Description
pszOldName | Input the existing name of the topology model.
pszNewName | Input the new name of the topology model.

Returns

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determines whether a specified topology exists in current or source (attached) drawing.

```cpp
bool TopologyExists(
    const ACHAR* pszName
);
```

File

AcMapTopologyManager.h

Parameters | Description
--- | ---
pszName | Input name of the topology to check.

Returns

Returns true if the topology exists; otherwise, returns false.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determine whether a specified topology exists in specified source.

```cpp
bool TopologyExists(
    const ACHAR* pszName,
    ETopologyScope enmSource
);
```

**Parameters**

- `pszName`: Input name of the topology to check.
- `enmSource`: Input ETopologyScope topology source.

**Returns**

Returns true if the topology exists in specified source; otherwise, returns false.

Created with a commercial version of [Doc-O-Matic](https://www.toolsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links

Topology: Class-Based API

Classes

Topology: Class-Based API

- **AcMapEntityCreationSettings**: Settings for creating entities. For more information, search for `topology` in AutoCAD Map Help.

- **AcMapFloodParameters**: Settings for flood parameters. For more information, search for `topology` in AutoCAD Map Help.

- **AcMapMarkerStyles**: Settings for error markers. For more information, search for `topology` in AutoCAD Map Help.

- **AcMapNetAnalysisParameters**: Settings for analysis parameters. For more information, search for `topology` in AutoCAD Map Help.

- **AcMapPointCreationSettings**: Settings for creating points. For more information, search for `topology` in AutoCAD Map Help.

- **AcMapTopoElement**: Manages topology objects. For more information, search for `topology` in AutoCAD Map Help.

- **AcMapTopoFullEdge**: Manages topology full edges. For more information, search for `topology` in AutoCAD Map Help.

- **AcMapTopoHalfEdge**: Manages topology half edges. For more information, search for `topology` in AutoCAD Map Help.

- **AcMapTopoIterator**: An iterator over a collection of AcMapTopology instances. For more information, search for `topology` in AutoCAD Map Help.

- **AcMapTopology**: The main topology object. For more information, search for `topology` in AutoCAD Map Help.

Stores the topology source information. For
<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AcMapTopologySource</td>
<td>more information, search for <em>topology</em> in AutoCAD Map Help.</td>
</tr>
<tr>
<td></td>
<td>Manages topology nodes. For more information, search for <em>topology</em> in AutoCAD Map Help.</td>
</tr>
<tr>
<td>AcMapTopoNode</td>
<td>For topology overlay analysis, use this class to set the data attributes that contain object data from linked database to include in the resulting topology, and the name and description of the new object data table. For more information, search for <em>topology</em> in AutoCAD Map Help.</td>
</tr>
<tr>
<td>AcMapTopoOverlayData</td>
<td>Manages topology polygons. For more information, search for <em>topology</em> in AutoCAD Map Help.</td>
</tr>
<tr>
<td>AcMapTopoPolygon</td>
<td>Manages topology rings. For more information, search for <em>topology</em> in AutoCAD Map Help.</td>
</tr>
<tr>
<td>AcMapTopoRing</td>
<td>Settings for trace parameters. For more information, search for <em>topology</em> in AutoCAD Map Help.</td>
</tr>
</tbody>
</table>

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Settings for creating entities. For more information, search for *topology* in AutoCAD Map Help.

class AcMapEntityCreationSettings;

File

AcMapTopology.h

Methods

```cpp
~AcMapEntityCreationSettings  // Destroys an instance of this class.
AcMapEntityCreationSettings   // Constructs an instance of this class.
```

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
AcMapEntityCreationSettings:: AcMapEntityCreationSettings Constructor

Constructs an instance of this class.

```cpp
AcMapEntityCreationSettings(
    const ACHAR* pszLayerName,
    int nColor
);
```

Parameters

<table>
<thead>
<tr>
<th>pszLayerName</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Input name of the layer for the created entity.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>nColor</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Input color index.</td>
</tr>
</tbody>
</table>

Returns

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.docomaticsoftware.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Settings for flood parameters. For more information, search for *topology* in AutoCAD Map Help.

```
class AcMapFloodParameters : public AcMapNetAnalysisParameters;
```

File

AcMapTopology.h

**Methods**

- `~AcMapFloodParameters` Destroys an instance of this class.
- `AcMapFloodParameters` Constructs an instance of this class.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Constructs an instance of this class.

```c++
AcMapFloodParameters(
    double dMaxResistance,
    const ACHAR* aNodeResistanceExpression,
    const ACHAR* aLinkDirectionExpression,
    const ACHAR* aLinkResistanceExpression,
    const ACHAR* aLinkReverseResistanceExpression,
    bool bUseReverseDirection,
    const ACHAR* aName = NULL,
    const ACHAR* aDescription = NULL
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dMaxResistance</td>
<td>Input maximum resistance.</td>
</tr>
<tr>
<td>aNodeResistanceExpression</td>
<td>Input node resistance expression.</td>
</tr>
<tr>
<td>aLinkDirectionExpression</td>
<td>Input link direction expression.</td>
</tr>
<tr>
<td>aLinkResistanceExpression</td>
<td>Input link resistance expression.</td>
</tr>
<tr>
<td>aLinkReverseResistanceExpression</td>
<td>Input link reverse resistance expression.</td>
</tr>
<tr>
<td>bUseReverseDirection</td>
<td>Input true to use reverse direction; otherwise, false.</td>
</tr>
<tr>
<td>aName</td>
<td>Input name of the new topology. If this value is NULL, no new topology is</td>
</tr>
<tr>
<td></td>
<td>created. The default value is NULL.</td>
</tr>
<tr>
<td>aDescription</td>
<td>Input description of the new topology. The default value is NULL.</td>
</tr>
</tbody>
</table>

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Settings for error markers. For more information, search for `topology` in AutoCAD Map Help.

```c
class AcMapMarkerStyles;
```

File

AcMapTopology.h

Methods

- `~AcMapMarkerStyles` Destroys an instance of this class.
- `SetMarkerSize` Sets the error-marker size.
- `SetTopoDuplicateCentroidMarker` Sets the marker shape and color for a duplicate-centroid error.
- `SetTopoIncompleteAreaMarker` Sets the marker shape and color for an incomplete-area error.
- `SetTopoIntersectionMarker` Sets the marker shape and color for an intersection error.
- `SetTopoMissingCentroidMarker` Sets the marker shape and color for a missing-centroid error.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Settings for analysis parameters. For more information, search for *topology* in AutoCAD Map Help.

```cpp
class AcMapNetAnalysisParameters;
```

File

`AcMapTopology.h`

**Methods**

- `~AcMapNetAnalysisParameters` Destroys an instance of this class.
- `AcMapNetAnalysisParameters` Constructs an instance of this class.

Created with a commercial version of *Doc-O-Matic*. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Constructs an instance of this class.

```cpp
AcMapNetAnalysisParameters(
    double dMinResistance,
    double dMaxResistance,
    const ACHAR* aNodeResistanceExpression,
    const ACHAR* aLinkDirectionExpression,
    const ACHAR* aLinkResistanceExpression,
    const ACHAR* aLinkReverseResistanceExpression,
    bool bUseReverseDirection,
    const ACHAR* aName = NULL,
    const ACHAR* aDescription = NULL
);
```

Parameters

- **dMinResistance**: Input minimum resistance.
- **dMaxResistance**: Input maximum resistance.
- **aNodeResistanceExpression**: Input node resistance expression.
- **aLinkDirectionExpression**: Input link direction expression.
- **aLinkResistanceExpression**: Input link resistance expression.
- **aLinkReverseResistanceExpression**: Input link reverse resistance expression.
- **bUseReverseDirection**: Input true to use reverse direction; otherwise, false.
- **aName**: Input name of the new topology. The default value is NULL.
- **aDescription**: Input description of the new topology. The default value is NULL.

Returns

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Settings for creating points. For more information, search for *topology* in AutoCAD Map Help.

```c
class AcMapPointCreationSettings : public AcMapEntityCreationSettings

File

AcMapTopology.h
```

Methods

- `~AcMapPointCreationSettings` Destroys an instance of this class.
- `AcMapPointCreationSettings` Constructs an instance of this class.

Created with a commercial version of *Doc-O-Matic*. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Constructs an instance of this class.

```
AcMapPointCreationSettings(
    const ACHAR* pszLayerName,
    int nColor,
    bool bCreate,
    const ACHAR* pszBlockname
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pszLayerName</td>
<td>Input name of layer name for the created node.</td>
</tr>
<tr>
<td>nColor</td>
<td>Input color index.</td>
</tr>
<tr>
<td>bCreate</td>
<td>Input true if creating a node; otherwise, false.</td>
</tr>
<tr>
<td>pszBlockname</td>
<td>Input name of the block for the created node.</td>
</tr>
</tbody>
</table>

**Returns**

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Manages topology objects. For more information, search for topology in AutoCAD Map Help.

```cpp
class AcMapTopoElement;
```

File

AcMapTopoObject.h

**Methods**

- `~AcMapTopoElement` Destroys an instance of this class.
- `AcMapTopoElement` Constructs an instance of this class.
- `GetEntity` Retrieves the object identifier of the AutoCAD entity associated with this object.
- `GetID` Retrieves the unique identifier of this object in its topology.
- `IsOnThisObject` Determines whether a specified point is within tolerance of this object.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Template for dynamic arrays of topology element pointers. Memory allocated for such arrays is released automatically when they go out of scope.

Most of the member functions of this class are defined in its parent class, AcArray.

```cpp
template <class T>
class AcMapTopoElementPtrArray : public AcArray<T *>{
```

File

AcMapTopologyCommonDef.h

Methods

~AcMapTopoElementPtrArray  Destroys the array.

Empty  Releases memory allocated to the array.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Manages topology full edges. For more information, search for *topology* in AutoCAD Map Help.

**class** `AcMapTopoFullEdge : public AcMapTopoElement;`

**File**

`AcMapTopoFullEdge.h`

**Enumerations**

``````
```EDirection``` Enumerates the directions of a full edge.

**Methods**

``````
```~AcMapTopoFullEdge``` Destroys an instance of this class.

`GetDirection` Retrieves the direction of this edge.

`GetEntity` Retrieves the object identifier of the AutoCAD entity associated with this edge.

`GetHalfEdge` Retrieves the half edge that traverses one way along this full edge.

`GetID` Retrieves the identifier of this edge in its topology.

`GetLength` Retrieves the length of this edge.

`GetNextEdge` Retrieves the next edge by moving forward or backward along this edge and turning left or right at the next node.

`GetNextNode` Retrieves the next node by moving forward or backward along this edge.

`GetPolygon` Retrieves the polygon associated with this edge (default level).

`GetResistance` Retrieves the resistance of this edge.

`GetRing` Retrieves the ring associated with this edge (default level).

`GetTopology` Retrieves the topology that contains this edge.
**IsOnThisObject**  Determines whether a specified point is within tolerance of this edge.

**SetDirection**  Sets the direction of this edge.

**SetResistance**  Sets the resistance of this edge.

Created with a commercial version of [Doc-O-Matic](#). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links

Classes
AcMapTopoHalfEdge Class

**Class**

Manages topology half edges. For more information, search for *topology* in AutoCAD Map Help.

```cpp
class AcMapTopoHalfEdge;
```

**File**

AcMapTopoHalfEdge.h

**Methods**

- `~AcMapTopoHalfEdge`
  - Destroys an instance of this class.
- `GetFullEdge`
  - Retrieves the full edge that contains this half edge.
- `GetNextEdge`
  - Retrieves the next edge by turning left or right at the next node.
- `GetNextNode`
  - Retrieves the next node by moving forward along this edge.
- `GetPolygon`
  - Retrieves the polygon associated with this edge (default level).
- `GetPreviousNode`
  - Retrieves the previous node by moving backward along this edge.
- `GetResistance`
  - Retrieves the resistance of this edge.
- `GetRing`
  - Retrieves the ring associated with this edge (default level).
- `GetTopology`
  - Retrieves the topology that contains this edge.
- `SetResistance`
  - Sets the resistance of this edge.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
An iterator over a collection of AcMapTopology instances. For more information, search for topology in AutoCAD Map Help.

```c
class AcMapTopoIterator;
```

File

AcMapTopoIterator.h

Methods

- `~AcMapTopoIterator` Destroys an instance of this class.
- `AcMapTopoIterator` Constructs an instance of this class.
- `Count` Counts the number of elements in the iteration.
- `First` Moves to the first element in the iteration.
- `GetDescription` Retrieves the description of this topology model.
- `GetName` Retrieves the name of this topology model.
- `GetTopology` Retrieves the topology of the current element in the iteration.
- `GetType` Retrieves the type of this topology model.
- `IsDone` Determines whether the iterator has reached the end of the collection.
- `Next` Advances to the next element in the iteration.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
The main topology object. For more information, search for *topology* in AutoCAD Map Help.

```cpp
class AcMapTopology;
```

File

AcMapTopology.h

- **Enumerations**
  - `EAuditResults` Enumerates the result types of an audit.
  - `ECreateOptions` Enumerates the options for creating a topology.
  - `EOpenMode` Enumerates the ways to open a topology model.
  - `EStatus` Enumerates the status types of a topology model.

- **Methods**
  - `~AcMapTopology` Destroys an instance of this class.
  - `AcMapTopology` Constructs an instance of this class.
  - `AddCurveObject` Adds a curve (linear object) to this topology model.
  - `AddPointObject` Adds a point to this topology model.
  - `AddPolygons` Adds objects to a polygon topology model.
  - `Buffer` Creates a new topology with the specified buffer setting on the current topology model. To set topology-creation values, call the three functions `SetNodeCreationSettings()`, `SetEdgeCreationSettings()`, and `SetCentroidCreationSettings()`.
  - `Clip` Uses the overlay polygon topology as a boundary to clip and discard the parts of the source polygons outside the overlay polygons. To set topology-creation values, call the three functions `SetNodeCreationSettings()`,
SetEdgeCreationSettings(), and
SetCentroidCreationSettings().

**Close**
Closes this topology model.

Create
Creates a topology. To set topology-creation values, call the three functions
SetNodeCreationSettings(),
SetEdgeCreationSettings(), and
SetCentroidCreationSettings(). If caller does not call these functions, then node creation defaults to false, missing-centroid creation defaults to true, and the node or centroid block name defaults to ACAD_POINT.

Create
Creates a topology - this function is not yet implemented in AutoCAD Map; use the other Create() function.

DeleteEdge
Deletes an edge from this topology model.

DeleteNode
Deletes a node from this topology model.

DeletePolygon
Deletes a polygon from the current topology model.

DeletePolygon
Deletes a polygon from the current topology model.

Dissolve
Combines entities that share the same data value in a specified column.

Erase
Uses the overlay polygon topology as a mask to erase everything in the source polygon topology that is covered by the overlay topology. To set topology-creation values, call the three functions
SetNodeCreationSettings(),
SetEdgeCreationSettings(), and
SetCentroidCreationSettings().

FindEdge
Finds the edge closest to a specified point.

FindEdge
Finds the edge with the specified ID.

FindNeighborPolygons
Finds all polygons that are neighbors of those edges that are based on a specified curve.

FindNode
Finds the node nearest a specified point.
<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>FindPolygon</strong></td>
<td>Finds the smallest polygon containing a specified point.</td>
</tr>
<tr>
<td><strong>FindTopologyObject</strong></td>
<td>Finds the object (node, full edge, or polygon) nearest a specified point within a specified tolerance.</td>
</tr>
<tr>
<td><strong>GetBackwardEdge</strong></td>
<td>Retrieves the half edge with the specified object ID from this topology model.</td>
</tr>
<tr>
<td><strong>GetDescription</strong></td>
<td>Retrieves the description of this topology model.</td>
</tr>
<tr>
<td><strong>GetEntityId</strong></td>
<td>Retrieves the object ID of the AutoCAD entity associated with the specified topology element ID in the current drawing or source drawing.</td>
</tr>
<tr>
<td><strong>GetEntityIds</strong></td>
<td>Retrieves the object IDs of all the AutoCAD entities associated with the topology elements in the current drawing.</td>
</tr>
<tr>
<td><strong>GetForwardEdge</strong></td>
<td>Retrieves the half edge with the specified object ID from this topology model.</td>
</tr>
<tr>
<td><strong>GetFullEdge</strong></td>
<td>Retrieves the full edge with the specified object ID from this topology model.</td>
</tr>
<tr>
<td><strong>GetFullEdges</strong></td>
<td>Retrieves all full edges from this topology model.</td>
</tr>
<tr>
<td><strong>GetMarkerStyles</strong></td>
<td>Retrieves the pointer to the error-marker settings of this topology model.</td>
</tr>
<tr>
<td><strong>GetName</strong></td>
<td>Retrieves the name of this topology model.</td>
</tr>
<tr>
<td><strong>GetNode</strong></td>
<td>Retrieves the node with the specified object ID from this topology model.</td>
</tr>
<tr>
<td><strong>GetNodes</strong></td>
<td>Retrieves all nodes from this topology model.</td>
</tr>
<tr>
<td><strong>GetPolygon</strong></td>
<td>Retrieves the polygon with the specified object ID from this topology model.</td>
</tr>
<tr>
<td><strong>GetPolygons</strong></td>
<td>Retrieves all polygons from this topology model.</td>
</tr>
<tr>
<td><strong>GetStatus</strong></td>
<td>Retrieves the status of this topology model.</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Retrieves the type of this topology model.</td>
</tr>
<tr>
<td><strong>Identity</strong></td>
<td>Performs a Union() on the source topology and an Intersect() on the overlay topology. To set topology-creation values, call the three functions <strong>SetNodeCreationSettings()</strong>, <strong>SetEdgeCreationSettings()</strong>, and <strong>SetCentroidCreationSettings()</strong>.</td>
</tr>
</tbody>
</table>
Intersect Combines topologies and keeps only the common geometry. Intersect acts like the Boolean AND operation. The results are the same no matter the order topologies are chosen. Object data are combined for the two operations. To set topology-creation values, call the three functions 
SetNodeCreationSettings(), SetEdgeCreationSettings(), and SetCentroidCreationSettings().

IsComplete Determines whether this topology model is complete.

IsFixedType Determines whether this topology model can contain only elements of its own type.

IsLinearType Determines whether this topology model is a network topology.

IsLogicalType Determines whether the elements in this topology model can be connected if their graphical representations are farther apart than the tolerance value.

IsPointType Determines whether this topology model is a point topology.

IsPolygonType Determines whether this topology model is a polygon topology.

MergePolygons Merges neighboring polygons, dissolving the boundaries between them.

MoveNode Moves a node to the new position.
**Open**

Opens the topology model in only the current drawing. After you are finished with a topology model, call Close() to close it.

**Open**

Opens the topology model in both current and source (attached) drawings. After you are finished with a topology model, call Close() to close it.

**Paste**

Pastes the overlay polygon topology on top of the source polygons. The source polygons not covered by the overlay remain. Paste can be used with only polygons. To set topology-creation values, call the three functions

SetNodeCreationSettings(),
SetEdgeCreationSettings(), and
SetCentroidCreationSettings().

**Refresh**

Updates a topology element.

**SetCentroidCreationSettings**

Sets the centroid-creation parameters of this topology model.

**SetDescription**

Sets the description of this topology model.

**SetEdgeCreationSettings**

Sets the edge-creation parameters of this topology model.

**SetNodeCreationSettings**

Sets the node-creation parameters of this topology model.

**ShowGeometry**

Highlights the topology model with the specified color.

**SplitPolygon**

Splits a polygon along a curve.

**SplitPolygon**

Splits a polygon along a curve.

**TraceBestPath**

Finds the least-cost path between start and end points with intermediate points to visit.

**TraceFlood**

Traces the flood from a specified point.

**TraceLeastCostPath**

Finds the least-cost path between two points.

**Union**

Combines polygons with polygons and keeps all geometry. Union acts like the Boolean OR operation and can be used with only polygons. To set topology-creation values, call the three functions SetNodeCreationSettings(), SetEdgeCreationSettings(), and
SetCentroidCreationSettings().

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Constructs an instance of this class.

```
AcMapTopology(
    const ACHAR* pszName
);
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pszName</td>
<td>Input name of the new topology.</td>
</tr>
</tbody>
</table>

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapTopology Class, AcMapTopology Class
AcMapTopology:: Buffer Method
AcMapTopology Class | AcMapTopology Class

Creates a new topology with the specified buffer setting on the current topology model. To set topology-creation values, call the three functions SetNodeCreationSettings(), SetEdgeCreationSettings(), and SetCentroidCreationSettings().

AcMap::EErrCode Buffer(
   const ACHAR* pszOffset,
   const ACHAR* pszTopoName,
   const ACHAR* pszTopoDesc = NULL,
   double defaultDistance = 1.0
);

Parameters   Description
pszOffset      Input offset distance, as a string. Set this value to NULL to use the default value.
pszTopoName    Input name of the new topology.
pszTopoDesc    Input description of the new topology. The default value is NULL.
defaultDistance Input default distance of the new topology. The default value is 1.0. The measurement units depend on the current drawing's unit settings.

Returns

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Uses the overlay polygon topology as a boundary to clip and discard the parts of the source polygons outside the overlay polygons. To set topology-creation values, call the three functions `SetNodeCreationSettings()`, `SetEdgeCreationSettings()`, and `SetCentroidCreationSettings()`.

```cpp
AcMap::EErrCode Clip(
    const AcMapTopology*& pOverlayTopology,
    const ACHAR* pszTopoName,
    const ACHAR* pszTopoDesc = NULL,
    const AcMapObjectDataTable* pResultDataTable = NULL,
    const AcMapOverlayDataArray* paSourceData = NULL,
    const AcMapOverlayDataArray* paOverlayData = NULL
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pOverlayTopology</td>
<td>Input overlay topology pointer.</td>
</tr>
<tr>
<td>pszTopoName</td>
<td>Input name of the new topology.</td>
</tr>
<tr>
<td>pszTopoDesc</td>
<td>Input description of the new topology. The default value is NULL.</td>
</tr>
<tr>
<td>pResultDataTable</td>
<td>Input <code>AcMapObjectDataTable</code> result object data table settings. The default value is NULL.</td>
</tr>
<tr>
<td>paSourceData</td>
<td>Input source data settings. The default value is NULL.</td>
</tr>
<tr>
<td>paOverlayData</td>
<td>Input overlay data settings. The default value is NULL.</td>
</tr>
</tbody>
</table>

**Returns**

Returns `AcMap::EErrCode kOk` if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Combines entities that share the same data value in a specified column.

```cpp
AcMap::EErrCode Dissolve(
    const ACHAR* pszAttributeExpr,
    const ACHAR* pszTopoName,
    const ACHAR* pszTopoDesc = NULL,
    const AcMapObjectDataField* pResultDataField = NULL
);
```

**Parameters**

- **pszAttributeExpr**: Input attribute expression string.
- **pszTopoName**: Input name of the new topology.
- **pszTopoDesc**: Input description of the new topology. The default value is NULL.
- **pResultDataField**: Input `AcMapObjectDataField` result object data table column settings. The default value is NULL.

**Returns**

Returns `AcMap::EErrCode kOk` if successful; otherwise, returns a different error code.

**Remarks**

When you dissolve a topology, AutoCAD Map checks each boundary between entities to see if the dissolve column value is the same for both. If it is, the boundary and one of the entities are removed. All object data columns except the dissolve and any new topology columns are removed.

Created with a commercial version of [Doc-O-Matic](#). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Uses the overlay polygon topology as a mask to erase everything in the source polygon topology that is covered by the overlay topology. To set topology-creation values, call the three functions `SetNodeCreationSettings()`, `SetEdgeCreationSettings()`, and `SetCentroidCreationSettings()`.

```cpp
AcMap::EErrCode Erase(
    const AcMapTopology*& pOverlayTopology,
    const ACHAR* pszTopoName,
    const ACHAR* pszTopoDesc = NULL,
    const AcMapObjectDataTable* pResultDataTable = NULL,
    const AcMapOverlayDataArray* paSourceData = NULL,
    const AcMapOverlayDataArray* paOverlayData = NULL
);
```

Parameters

- **pOverlayTopology**: Input overlay topology pointer.
- **pszTopoName**: Input name of the new topology.
- **pszTopoDesc**: Input description of the new topology. The default value is NULL.
- **pResultDataTable**: Input `AcMapObjectDataTable` result object data table settings. The default value is NULL.
- **paSourceData**: Input source data settings. The default value is NULL.
- **paOverlayData**: Input overlay data settings. The default value is NULL.

Returns

Returns `AcMap::EErrCode kOk` if successful; otherwise, returns a different error code.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at `support@toolsfactory.com`.
Links
AcMapTopology Class, AcMapTopology Class
AcMapTopology:: Identity Method
AcMapTopology Class | AcMapTopology Class

Performs a Union() on the source topology and an Intersect() on the overlay topology. To set topology-creation values, call the three functions SetNodeCreationSettings(), SetEdgeCreationSettings(), and SetCentroidCreationSettings().

AcMap::EErrCode Identity(
    const AcMapTopology* & pOverlayTopology,
    const ACHAR* pszTopoName,
    const ACHAR* pszTopoDesc = NULL,
    const AcMapObjectDataTable* pResultDataTable = NULL,
    const AcMapOverlayDataArray* paSourceData = NULL,
    const AcMapOverlayDataArray* paOverlayData = NULL
);

Parameters Description
pOverlayTopology Input overlay topology pointer.
pszTopoName Input name of the new topology.
pszTopoDesc Input description of the new topology. The default value is NULL.
pResultDataTable Input AcMapObjectDataTable result object data table settings. The default value is NULL.
paSourceData Input source data settings. The default value is NULL.
paOverlayData Input overlay data settings. The default value is NULL.

Returns
Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Combines topologies and keeps only the common geometry. Intersect acts like the Boolean AND operation. The results are the same no matter the order topologies are chosen. Object data are combined for the two operations. To set topology-creation values, call the three functions SetNodeCreationSettings(), SetEdgeCreationSettings(), and SetCentroidCreationSettings().

```cpp
AcMap::EErrCode Intersect(
    const AcMapTopology*& pOverlayTopology,
    const ACHAR* pszTopoName,
    const ACHAR* pszTopoDesc = NULL,
    const AcMapObjectDataTable* pResultDataTable = NULL,
    const AcMapOverlayDataArray* paSourceData = NULL,
    const AcMapOverlayDataArray* paOverlayData = NULL
);
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pOverlayTopology</td>
<td>Input overlay topology pointer.</td>
</tr>
<tr>
<td>pszTopoName</td>
<td>Input name of the new topology.</td>
</tr>
<tr>
<td>pszTopoDesc</td>
<td>Input description of the new topology . The default value is NULL.</td>
</tr>
<tr>
<td>pResultDataTable</td>
<td>Input AcMapObjectDataTable result object data table settings. The default value is NULL.</td>
</tr>
<tr>
<td>paSourceData</td>
<td>Input source data settings. The default value is NULL.</td>
</tr>
<tr>
<td>paOverlayData</td>
<td>Input overlay data settings. The default value is NULL.</td>
</tr>
</tbody>
</table>

Returns

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Pastes the overlay polygon topology on top of the source polygons. The source polygons not covered by the overlay remain. Paste can be used with only polygons. To set topology-creation values, call the three functions `SetNodeCreationSettings()`, `SetEdgeCreationSettings()`, and `SetCentroidCreationSettings()`.

```cpp
AcMap::EErrCode Paste(
    const AcMapTopology*& pOverlayTopology,
    const ACHAR* pszTopoName,
    const ACHAR* pszTopoDesc = NULL,
    const AcMapObjectDataTable* pResultDataTable = NULL,
    const AcMapOverlayDataArray* paSourceData = NULL,
    const AcMapOverlayDataArray* paOverlayData = NULL
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pOverlayTopology</td>
<td>Input overlay topology pointer.</td>
</tr>
<tr>
<td>pszTopoName</td>
<td>Input name of the new topology.</td>
</tr>
<tr>
<td>pszTopoDesc</td>
<td>Input description of the new topology. The default value is NULL.</td>
</tr>
<tr>
<td>pResultDataTable</td>
<td>Input <code>AcMapObjectDataTable</code> result object data table settings. The default value is NULL.</td>
</tr>
<tr>
<td>paSourceData</td>
<td>Input source data settings. The default value is NULL.</td>
</tr>
<tr>
<td>paOverlayData</td>
<td>Input overlay data settings. The default value is NULL.</td>
</tr>
</tbody>
</table>

**Returns**

Returns `AcMap::EErrCode kOk` if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapTopology Class, AcMapTopology Class
AcMapTopology:: SetDescription Method
AcMapTopology Class | AcMapTopology Class

Sets the description of this topology model.

AcMap::EErrCode SetDescription(
    const ACHAR* pszName
);  
Parameters Description
pszName Input description of the topology model.

Returns

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Combines polygons with polygons and keeps all geometry. Union acts like the Boolean OR operation and can be used with only polygons. To set topology-creation values, call the three functions SetNodeCreationSettings(), SetEdgeCreationSettings(), and SetCentroidCreationSettings().

```cpp
AcMap::EErrCode Union(
    const AcMapTopology*& pOverlayTopology,
    const ACHAR* pszTopoName,
    const ACHAR* pszTopoDesc = NULL,
    const AcMapObjectDataTable* pResultDataTable = NULL,
    const AcMapOverlayDataArray* paSourceData = NULL,
    const AcMapOverlayDataArray* paOverlayData = NULL
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pOverlayTopology</td>
<td>Input overlay topology pointer.</td>
</tr>
<tr>
<td>pszTopoName</td>
<td>Input name of the new topology.</td>
</tr>
<tr>
<td>pszTopoDesc</td>
<td>Input description the new topology. The default value is NULL.</td>
</tr>
<tr>
<td>pResultDataTable</td>
<td>Input AcMapObjectDataTable result object data table settings. The default value is NULL.</td>
</tr>
<tr>
<td>paSourceData</td>
<td>Input source data settings. The default value is NULL.</td>
</tr>
<tr>
<td>paOverlayData</td>
<td>Input overlay data settings. The default value is NULL.</td>
</tr>
</tbody>
</table>

Returns

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Stores the topology source information. For more information, search for *topology* in AutoCAD Map Help.

```
class AcMapTopologySource;
```

File

AcMapTopologyManager.h

Methods

- `~AcMapTopologySource` Destroys an instance of this class.
- `AcMapTopologySource` Constructs an instance of this class.
- `AcMapTopologySource` Constructs an instance of this class by using a copy constructor.
- `GetName` Retrieves the topology name.
- `GetPath` Retrieves the full path of the topology.
- `GetSource` Retrieves the source of the topology.
- `IsLoaded` Determines whether the topology is loaded.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links

Classes
AcMapTopoNode Class

Classes

Manages topology nodes. For more information, search for *topology* in AutoCAD Map Help.

```cpp
class AcMapTopoNode : public AcMapTopoElement;
```

File

AcMapTopoNode.h

Methods

- `~AcMapTopoNode` Destroys an instance of this class.
- `GetEdges` Retrieves this node's edges in counter-clockwise order.
- `GetEntity` Retrieves the object identifier of the AutoCAD entity associated with this node.
- `GetID` Retrieves the unique identifier of this node in its topology.
- `GetLocation` Retrieves the position of this node.
- `GetNextEdge` Retrieves the next edge given an incoming full edge and a left or right turn direction.
- `GetNextEdge` Retrieves the next edge given an incoming half edge and a left or right turn direction.
- `GetResistance` Retrieves the resistance of this node.
- `GetTopology` Retrieves the topology that contains this node.
- `IsMoveable` Determines whether this node is moveable - this function is not yet implemented in AutoCAD Map and always returns true.
- `IsOnThisObject` Determines whether a specified point is within tolerance of this node.
- `SetIsMoveable` Sets whether this node is moveable - this function is not yet implemented in AutoCAD Map.
- `SetLocation` Sets the position of this node - this function is not yet implemented in AutoCAD Map.
SetResistance  
Sets the resistance of this node.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at **support@toolsfactory.com**.
For topology overlay analysis, use this class to set the data attributes that contain object data from linked database to include in the resulting topology, and the name and description of the new object data table. For more information, search for *topology* in AutoCAD Map Help.

```c
class AcMapTopoOverlayData;
```

**File**
AcMapTopology.h

**Methods**
- `~AcMapTopoOverlayData` Destroys an instance of this class.
- `AcMapTopoOverlayData` Constructs an instance of this class.
- `AcMapTopoOverlayData` Constructs an instance of this class by using the specified expression, column name, and column data type.
- `AcMapTopoOverlayData` Constructs an instance of this class by using a copy constructor.
- `GetColumnName` Retrieves the column name of the object data table.
- `GetColumnType` Retrieves the column data type of the object data table.
- `GetExpression` Retrieves the expression.
- `SetColumnName` Sets the column name of the object data table.
- `SetColumnType` Sets the column data type of the object data table.
- `SetExpression` Sets the expression.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Constructs an instance of this class by using the specified expression, column name, and column data type.

```cpp
AcMapTopoOverlayData(
    const ACHAR* pszExpression,
    const ACHAR* pszColumnName,
    AcMap::EDataType enmColumnType
);
```

Parameters | Description
---|---
pszExpression | Input the expression.
pszColumnName | Input column name.
enmColumnType | Input column data type.

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the column name of the object data table.

```cpp
AcMap::EErrCode SetColumnName(
    const ACHAR* pszColumnName
);
```

Parameters | Description
--- | ---
pszColumnName | Input column name.

Returns

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the expression.

```
AcMap::EErrCode SetExpression(
    const ACHAR* pszExpression
);
```

Parameters Description
pszExpression Input expression.

Returns

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Manages topology polygons. For more information, search for topology in AutoCAD Map Help.

class AcMapTopoPolygon : public AcMapTopoElement;

File
AcMapTopoPolygon.h

Methods

~AcMapTopoPolygon
Retrieves the area of this polygon.

GetArea
Retrieves the boundary of this polygon, represented by a list of rings.

GetBoundary
Retrieves the position of this polygon's centroid.

GetCentroid
Retrieves the object identifier of the AutoCAD entity associated with this polygon.

GetEntity
Retrieves the hierarchical child polygons, if any, of this polygon - this function is not yet implemented in AutoCAD Map.

GetHierChildren
Retrieves the hierarchical parent polygon, if any, of this polygon - this function is not yet implemented in AutoCAD Map.

GetHierParent
Retrieves the unique identifier of this polygon in its topology.

GetID
Retrieves the multi-polygon parent, if any, of this polygon - this function is not yet implemented in AutoCAD Map.

GetParent
Retrieves the perimeter of this polygon.

GetPerimeter
Retrieves the topology that contains this polygon.

GetTopology
Determines whether a specified point is within tolerance of this polygon.

IsOnThisObject
Traverses to another polygon by following a curve - this function is not yet implemented in AutoCAD Map.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links

**Classes**

AcMapTopoRing Class

**Classes**

Manages topology rings. For more information, search for *topology* in AutoCAD Map Help.

class AcMapTopoRing;

File

AcMapTopoRing.h

Methods

- ~AcMapTopoRing: Destroys an instance of this class.
- GetArea: Retrieves the area of this ring.
- GetEdges: Retrieves the edges that make up this ring.
- GetLength: Retrieves the length of this ring.
- GetStartEdge: Retrieves the first edge in this ring.
- IsExterior: Determines whether this ring is exterior (that is, an outer boundary of its polygon).

Created with a commercial version of [Doc-O-Matic](https://www.xtek.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Settings for trace parameters. For more information, search for *topology* in AutoCAD Map Help.

```csharp
class AcMapTraceParameters : public AcMapNetAnalysisParameters;
```

File

AcMapTopology.h

- **Methods**
  - `~AcMapTraceParameters` Destroys an instance of this class.
  - `AcMapTraceParameters` Constructs an instance of this class.

Created with a commercial version of [Doc-O-Matic](https://www.dociotools.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Constructs an instance of this class.

```cpp
AcMapTraceParameters(
    double dMinResistance,
    double dMaxResistance,
    const ACHAR* aNodeResistanceExpression,
    const ACHAR* aLinkDirectionExpression,
    const ACHAR* aLinkResistanceExpression,
    const ACHAR* aLinkReverseResistanceExpression,
    bool bUseReverseDirection,
    const ACHAR* aName = NULL,
    const ACHAR* aDescription = NULL
);```

Parameters
- **dMinResistance**: Input minimum resistance.
- **dMaxResistance**: Input maximum resistance.
- **aNodeResistanceExpression**: Input node resistance expression.
- **aLinkDirectionExpression**: Input link direction expression.
- **aLinkResistanceExpression**: Input link resistance expression.
- **aLinkReverseResistanceExpression**: Input link reverse resistance expression.
- **bUseReverseDirection**: Input true to use reverse direction; otherwise, false.
- **aName**: Input name of the new topology. If this value is NULL, no new topology is created. The default value is NULL.
- **aDescription**: Input description of the new topology. The default value is NULL.

Returns
- Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links

Topology: Class-Based API, Structures

Structs, Records, Enums

Topology: Class-Based API | Structures

Enumerations

- `ETopologyMarkType` Topology error-marker shapes.
- `ETopologyType` Topology types.

Structures

- `AcMapObjectDataField` Information about a field in a topology object data table.
- `AcMapObjectDataTable` Information about a topology object data table.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Information about a field in a topology object data table.

```
struct AcMapObjectDataField {
    ACHAR m_aODFieldName[MAX_FIELD_NAME];
    ACHAR m_aODTableName[MAX_TABLE_NAME];
};
```

File

AcMapTopologyCommonDef.h

Parameters

- `m_aODFieldName[MAX_FIELD_NAME]`  
  Field name. Default MAX_FIELD_NAME is 32.

- `m_aODTableName[MAX_TABLE_NAME]`  
  Table name. Default MAX_TABLE_NAME is 25.

Created with a commercial version of [Doc-O-Matic](https://www.dYNAMO.com/). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Information about a topology object data table.

```c
struct AcMapObjectDataTable {
    ACHAR m_aODTableName[MAX_TABLE_NAME];
    ACHAR m_aODTableDescription[MAX_DESC];
};
```

File

AcMapTopologyCommonDef.h

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>m_aODTableName[MAX_TABLE_NAME]</td>
<td>Table name. Default MAX_TABLE_NAME is 25.</td>
</tr>
<tr>
<td>m_aODTableDescription[MAX_DESC]</td>
<td>Table description. Default MAX_DESC is 512.</td>
</tr>
</tbody>
</table>

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Topology error-marker shapes.

```c
enum ETopologyMarkType {
    eMarkOctagon = 1,
    eMarkTriangle = 2,
    eMarkRhombus = 3,
    eMarkSquare = 4
};
```

File

`AcMapTopologyCommonDef.h`

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>eMarkOctagon</td>
<td>Octagon.</td>
</tr>
<tr>
<td>eMarkTriangle</td>
<td>Triangle.</td>
</tr>
<tr>
<td>eMarkRhombus</td>
<td>Rhombus.</td>
</tr>
<tr>
<td>eMarkSquare</td>
<td>Square.</td>
</tr>
</tbody>
</table>

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links

Structs, Records, Enums

ETopologyType Enumeration

Structs, Records, Enums

Topology types.

```c
enum ETopologyType {
    ePoint = 0x01,
    eLinear = 0x02,
    ePolygon = 0x04,
    eLogical = 0x08,
    eFixed = 0x10
};
```

File

AcMapTopologyCommonDef.h

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ePoint</td>
<td>Node topology.</td>
</tr>
<tr>
<td>eLinear</td>
<td>Network topology.</td>
</tr>
<tr>
<td>ePolygon</td>
<td>Polygon topology.</td>
</tr>
<tr>
<td>eLogical</td>
<td>Geometry/topology mismatch (electrical utility style). Always false.</td>
</tr>
<tr>
<td>eFixed</td>
<td>Topology cannot accept elements of other types. Always true.</td>
</tr>
</tbody>
</table>

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
### Types

<table>
<thead>
<tr>
<th>Class Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>AcMapFullEdgePtrArray</strong></td>
<td>Dynamic array of AcMapTopoFullEdgepointers. For information about member functions, see the template class <code>AcMapTopoElementPtrArray</code> and its parent class <code>AcArray</code>.</td>
</tr>
<tr>
<td><strong>AcMapHalfEdgePtrArray</strong></td>
<td>Dynamic array of AcMapTopoHalfEdgepointers. For information about member functions, see the template class <code>AcMapTopoElementPtrArray</code> and its parent class <code>AcArray</code>.</td>
</tr>
<tr>
<td><strong>AcMapNodePtrArray</strong></td>
<td>Dynamic array of AcMapTopoNodepointers. For information about member functions, see the template class <code>AcMapTopoElementPtrArray</code> and its parent class <code>AcArray</code>.</td>
</tr>
<tr>
<td><strong>AcMapObjectPtrArray</strong></td>
<td>Dynamic array of AcMapTopoElementpointers. For information about member functions, see the template class <code>AcMapTopoElementPtrArray</code> and its parent class <code>AcArray</code>.</td>
</tr>
<tr>
<td><strong>AcMapOverlayDataArray</strong></td>
<td>This is type <code>AcMapOverlayDataArray</code>.</td>
</tr>
<tr>
<td><strong>AcMapPolygonPtrArray</strong></td>
<td>Dynamic array of AcMapTopoPolygonpointers. For information about member functions, see the template class <code>AcMapTopoElementPtrArray</code> and its parent class <code>AcArray</code>.</td>
</tr>
<tr>
<td><strong>AcMapRingPtrArray</strong></td>
<td>Dynamic array of AcMapTopoRingpointers. For information about member functions, see the template class <code>AcMapTopoElementPtrArray</code> and its parent class <code>AcArray</code>.</td>
</tr>
</tbody>
</table>

**AcMapTopologySourceArray** This is type `AcMapTopologySourceArray`. Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links

Types
AcMapFullEdgePtrArray Type

Types

Dynamic array of AcMapTopoFullEdgepointers. For information about member functions, see the template class AcMapTopoElementPtrArray and its parent class AcArray.

typedef AcMapTopoElementPtrArray<AcMapTopoFullEdge> AcMapFullEdgePtrArray;

File

AcMapTopologyCommonDef.h

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Dynamic array of AcMapTopoHalfEdgePointers. For information about member functions, see the template class `AcMapTopoElementPtrArray` and its parent class `AcArray`.

```cpp
typedef AcMapTopoElementPtrArray<AcMapTopoHalfEdge> AcMapHalfEdgePtrArray;
```

File

AcMapTopologyCommonDef.h

Created with a commercial version of `Doc-O-Matic`. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at `support@toolsfactory.com`.
Links

Types
AcMapNodePtrArray Type

Types

Dynamic array of AcMapTopoNodepointers. For information about member functions, see the template class AcMapTopoElementPtrArray and its parent class AcArray.

```cpp
typedef AcMapTopoElementPtrArray<AcMapTopoNode> AcMapNodePtrArray;
```

File

AcMapTopologyCommonDef.h

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Dynamic array of AcMapTopoElement pointers. For information about member functions, see the template class `AcMapTopoElementPtrArray` and its parent class AcArray.

```cpp
typedef AcMapTopoElementPtrArray<AcMapTopoElement> AcMapObjectPtrArray;
```

File

`AcMapTopologyCommonDef.h`

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
This is type AcMapOverlayDataArray.

```cpp
typedef AcArray< AcMapTopoOverlayData > AcMapOverlayDataArray;
```

File

AcMapTopology.h

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Dynamic array of AcMapTopoPolygonpointers. For information about member functions, see the template class AcMapTopoElementPtrArray and its parent class AcArray.

```cpp
typedef AcMapTopoElementPtrArray<AcMapTopoPolygon> AcMapPolygonPtrArray;
```

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Dynamic array of AcMapTopoRingpointers. For information about member functions, see the template class `AcMapTopoElementPtrArray` and its parent class `AcArray`.

```cpp
typedef AcMapTopoElementPtrArray<AcMapTopoRing> AcMapRingPtrArray;
```

File

AcMapTopologyCommonDef.h

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
This is type AcMapTopologySourceArray.

typedef AcArray< AcMapTopologySource > AcMapTopologySourceArray;

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Topology Global-Function API (Deprecated)

Except for drawing cleanup functions, the global-function API for topology is deprecated in favor of the class-based API.

<table>
<thead>
<tr>
<th>Table Title</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Topology Function Catalog</td>
<td>Topology functions sorted by name.</td>
</tr>
<tr>
<td>Topology Function Synopsis</td>
<td>Topology functions sorted by functional group.</td>
</tr>
<tr>
<td>Topology Configuration Variables</td>
<td>Cleanup variables, cleanup-action variables, topology variables.</td>
</tr>
</tbody>
</table>
Topology Function Catalog

Topology Global-Function API

Topology functions sorted by name.

Note Except for drawing cleanup functions, the global-function API for topology is deprecated in favor of the class-based API.

ac | ana | clean | edit | elem | info | iter | mnt | qry | trace | var

tpm_acclose tpm_acexist
tpm_acload
tpm_acopen
tpm_acqty
tpm_acunload
tpm_acupgradeopen
tpm_anabuffer
tpm_anadissolve
tpm_anaoverlay
tpm_cleanactionlistdel
tpm_cleanactionlistgetat
tpm_cleanactionlistins
tpm_cleanactionlistqty
tpm_cleanalloc
tpm_cleananchorss
tpm_cleancancel
tpm_cleancomplete
tpm_cleancreatedss
tpm_cleanend
tpm_cleanerrorcur
tpm_cleanerrordraw
tpm_cleanerrorfix
tpm_cleanerrorget
tpm_cleanerrormark
tpm_cleanerrorset
tpm_cleanfree
tpm_cleangroupdraw
tpm_mntbuild
tpm_mnterase
tpm_mntrebuild
tpm_mntrename

tpm_qrygetresdesc
tpm_qrygetrestopo
tpm_qrygettoponame
tpm_qrygetrestopo
tpm_qrysettoponame

tpm_tracealloc
tpm_tracebestroute
tpm_tracebestroutescan
tpm_tracebestrouteval
tpm_traceelementedit
tpm_traceelementget
tpm_traceelementid
tpm_traceflood
tpm_tracefree
tpm_traceqty
tpm_tracesetmaxres
tpm_tracesetminres
tpm_traceshort
tpm_traceshortscan
tpm_traceshortval

tpm_varalloc
tpm_varfree
tpm_varget
tpm_varlist
tpm_varset
Topology Function Synopsis

Topology Global-Function API

Topology functions sorted by functional group.

**Note** Except for drawing cleanup functions, the global-function API for topology is deprecated in favor of the class-based API.

Access Functions
Analyzing Functions
Building and Erasing Functions
Drawing Cleanup Functions
Editing Functions
Element Information Functions
Iterating Functions
Network Tracing Functions
Topology Information Functions
Topology Query Functions
Topology Variables Functions
Cleanup variables store properties for cleanup models.

Cleanup variables are a subset of the configuration variables data structure.

**Note** Data types cited below are AutoLISP types. The AutoLISP real corresponds to the C++ double. Similarly, integer corresponds to long.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ANCHOROBS_FEATURES</td>
<td>!! New !! Classified objects to include, a comma-separated list of feature names (string), such as &quot;Roads, Rivers, Streets&quot;. Default = &quot;*&quot; (all features).</td>
</tr>
<tr>
<td>ANCHOROBS LAYERS</td>
<td>Anchored entities specified by <code>tpm_cleaninitanchorset</code> will be anchored only if they reside on layers specified here, with multiple layer names separated by commas. For example, &quot;0,Layer1,Layer2&quot;. Default = &quot;*&quot; (any layer)</td>
</tr>
<tr>
<td>APPARENT_INTERSECTION_COLOR</td>
<td>!! New !! Color for apparent-intersection markers, an AutoCAD color index (integer). See Color Index Colors. Default = 7 (white)</td>
</tr>
<tr>
<td>APPARENT_INTERSECTION_MARKER</td>
<td>!! New !! Marker shape for apparent-intersection errors (integer). See Marker Shapes. Default = 2 (triangle)</td>
</tr>
<tr>
<td>ARC_TYPE</td>
<td>Convert arcs to (integer): 0 = Arc (default) 1 = 2D polyline</td>
</tr>
<tr>
<td>CIRCLE_TYPE</td>
<td>Convert circles to (integer): 0 = Circle (default) 1 = 2D polyline 2 = Arc</td>
</tr>
<tr>
<td>CLEAN_TOL</td>
<td>Cleanup tolerance (positive real). Default = 0.01</td>
</tr>
<tr>
<td>CLUSTER_COLOR</td>
<td>Color for cluster markers, an AutoCAD color index (integer). See Color Index Colors.</td>
</tr>
<tr>
<td>Variable</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>CLUSTER_MARKER</td>
<td>Marker shape for cluster errors (integer). See Marker Shapes.</td>
</tr>
<tr>
<td>CONVERT</td>
<td>Flag for the original entities conversion (integer). 0 = Do not convert (default) 1 = Convert</td>
</tr>
<tr>
<td>CORRIDOR</td>
<td>Corridor width (positive real). Default = 0.01</td>
</tr>
<tr>
<td>CROSS_COLOR</td>
<td>Color for cross markers, an AutoCAD color index (integer). See Color Index Colors. Default = 2 (yellow)</td>
</tr>
<tr>
<td>CROSS_MARKER</td>
<td>Marker shape for cross errors (integer). See Marker Shapes.</td>
</tr>
<tr>
<td>DANGL_COLOR</td>
<td>Color for dangling node markers, an AutoCAD color index (integer). See Color Index Colors. Default = 1 (red)</td>
</tr>
<tr>
<td>DANGL_MARKER</td>
<td>Marker shape for dangling node errors (integer). See Marker Shapes.</td>
</tr>
<tr>
<td>DEL_MARKER</td>
<td>Whether to delete previous error markers (integer). 0 = Do not delete 1 = Delete (default)</td>
</tr>
<tr>
<td>DUPL_COLOR</td>
<td>Color for duplicates markers, an AutoCAD color index (integer). See Color Index Colors. Default = 6 (magenta)</td>
</tr>
<tr>
<td>DUPL_MARKER</td>
<td>Marker shape for duplicate errors (integer). See Marker Shapes.</td>
</tr>
<tr>
<td>ENT_PROCESS</td>
<td>Flag for original entities processing (integer). 1 = Convert original 2 = Create new and keep original 3 = Create new and delete original Default = 0</td>
</tr>
</tbody>
</table>
| **GENERALIZE** | Flag for generalization (link simplification) (**integer**).  
0 = Do not generalize (default)  
1 = Generalize  
**Note** If you are using an explicit cleanup action list (specifying cleanup actions using `tpm_cleanactionlists`), this setting is ignored. |
| **INCLUDE_LINEAROBS** | !! New !! Whether to include linear objects while deleting duplicates (**integer**).  
0 = Do not include.  
1 = Include.  
Default = 1. |
| **INCLUDE_POINTS** | !! New !! Whether to include points while deleting duplicates or snapping clustered nodes (**integer**).  
0 = Do not include.  
1 = Include.  
Default = 1. |
| **INCLUDE_BLOCKS** | !! New !! Whether to include blocks while deleting duplicates or snapping clustered nodes (**integer**).  
0 = Do not include.  
1 = Include.  
Default = 0. |
| **INCLUDE_TEXT** | !! New !! Whether to include text while deleting duplicates (**integer**).  
0 = Do not include.  
1 = Include.  
Default = 0. |
| **INCLUDE_MTEXT** | !! New !! Whether to include mtext while deleting duplicates (**integer**).  
0 = Do not include.  
1 = Include.  
Default = 1. |
| **INCLUDE_ROTATION** | !! New !! Whether to include rotation while deleting duplicates (**integer**).  
0 = Do not include.  
1 = Include.  
Default = 0. |
| **INCLUDE_ZVALUES** | !! New !! Whether to include z-values while deleting duplicates (**integer**).  
0 = Do not include.  
1 = Include. |
<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SNAP_TO_NODE</td>
<td>!! New !! Whether to snap to nodes or links (integer). 0 = Snap to nodes. 1 = Snap to links. Default = 0.</td>
</tr>
<tr>
<td>INCLUDEOBS_AUTOSELECT</td>
<td>How entities are specified for cleanup 0 = Select entities manually (that is, by passing a selection set to <code>tpm_cleaninit</code>) 1 = Select all entities in the drawing (in which case the selection set passed to <code>tpm_cleaninit</code>, if any, is ignored)</td>
</tr>
<tr>
<td>INCLUDEOBS_FEATURES</td>
<td>!! New !! Classified objects to include, a comma-separated list of features (string), such as &quot;Roads, Rivers, Streets&quot;. Default = &quot;*&quot; (all features).</td>
</tr>
<tr>
<td>INCLUDEOBS_LAYERS</td>
<td>Entities specified by <code>tpm_cleaninit</code> will be cleaned only if they reside on layers specified here, with multiple layer names separated by commas. For example, &quot;0,Layer1,Layer2&quot;. Default = &quot;*&quot; (any layer)</td>
</tr>
<tr>
<td>LINE_TYPE</td>
<td>Convert lines to (integer) 0 = Line (default) 1 = 2D polyline</td>
</tr>
<tr>
<td>LINK_COLOR</td>
<td>Color for new links, an AutoCAD color index (integer). See Color Index Colors. Negative integer = Current color Default = -1</td>
</tr>
<tr>
<td>LINK_CORRECT</td>
<td>Flag for link error correction (integer). 0 = Do not correct 1 = Correct (default) Note If you are using an explicit cleanup action list (specifying cleanup actions using <code>tpm_cleanactionlistins</code>), this setting is ignored.</td>
</tr>
<tr>
<td>LINK_ERROR</td>
<td>Link error types (integer), sum of the desired options. 0 = None 1 = Short 2 = Cross 4 = Undershoot 8 = Duplicate Default = 15</td>
</tr>
<tr>
<td>LINK_LAYER</td>
<td>Layer name for new links (string). nil = Current layer</td>
</tr>
<tr>
<td>Variable</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| MAINTAIN_MARKERS       | Flag to maintain markers when cleanup ends  
|                        | 0 = Do not maintain  
|                        | 1 = Maintain markers  
|                        | This variable affects only errors that have not been fixed. If an error has been fixed, its marker is deleted at the end of the cleanup process no matter what (by tpm_cleanend). |
| MARKER_HEIGHT          | The height of error markers, a percent of screen height (positive real). Default = 5.0                                                                                                                      |
| NODE_CORRECT           | Flag for node error correction (integer).  
|                        | 0 = Do not correct  
|                        | 1 = Correct (default)  
|                        | **Note** If you are using an explicit cleanup action list (specifying cleanup actions using tpm_cleanactionlistins), this setting is ignored.                                                               |
| NODE_ERROR             | Node error types (integer), sum of the desired options.  
|                        | 0 = None  
|                        | 16 = Cluster  
|                        | 32 = Pseudo  
|                        | 64 = Dangling  
|                        | Default = 48                                                                                                                                      |
| POLY3D_TYPE            | Convert 3D polylines to (integer).  
|                        | 0 = 3D polyline (default)  
<p>|                        | 1 = 2D polyline                                                                                                                                   |
| PSEUDO_COLOR           | Color for pseudo node markers, an AutoCAD color index (integer). See Color Index Colors. Default = 6 (magenta)                                                                                               |
| PSEUDO_MARKER          | Marker shape for pseudo node errors (integer). See Marker Shapes. Default = 2 (triangle)                                                                                                                  |
| SHORT_COLOR            | Color for short markers, an AutoCAD color index (positive integer). See Color Index Colors. Default = 1 (red)                                                                                               |
| SHORT_MARKER           | Marker shape for short errors (integer). See Marker Shapes. Default = 1 (octagon)                                                                                                                         |
| UNDER_COLOR            | Color for undershoot markers, an AutoCAD color index                                                                                               |</p>
<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>UNDER_MARKER</td>
<td>Marker shape for undershoot errors (integer). See Marker Shapes. Default = 1 (octagon)</td>
</tr>
<tr>
<td>WEED_DISTANCE</td>
<td>!! New !! Weeding distance (real). Default = 15.0</td>
</tr>
<tr>
<td>WEED_ANGLE</td>
<td>!! New !! Weed distance (real). Default = 4.0</td>
</tr>
<tr>
<td>WEED_SUPPLEMENT_DISTANCE</td>
<td>!! New !! Weed supplement distance (real). Default = 100.0</td>
</tr>
<tr>
<td>WEED_SUPPLEMENT_BULGE</td>
<td>!! New !! Weed supplement bulge (real). Default = 1.0</td>
</tr>
</tbody>
</table>
Cleanup Action Variables

**Topology Configuration Variables**

Cleanup action variables store properties for cleanup actions, which are assigned to action lists by `tpm_cleanactionlistins`.

Action lists are a feature of cleanup models. Cleanup action variables are a subset of the configuration variables data structure.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
</table>
| **ARC_TYPE** | With the Simplify Objects cleanup action (clean group type 128) only, whether to create arcs (integer).  
1 = Create arcs (default)  
0 = Do not create arcs  
For any action other than 128, Simplify Objects, ARC_TYPE does not matter. |
| **CLEAN_TOL** | Cleanup tolerance (positive real).  
Default = 0.01  
CLEAN_TOL affects the following actions only:  
8 = Delete Duplicates  
1 = Erase Short Objects  
4 = Extend Undershoots  
16 = Snap Clustered Nodes  
64 = Erase Dangling Objects  
128 = Simplify Objects  
With the following actions, the CLEAN_TOL value does not matter:  
2 = Break Crossing Objects  
32 = Dissolve Pseudo Nodes  
256 = Zero Length Objects |
| **CONVERT** | With the Extend Undershoots cleanup action (clean group type 4) only, whether to break the target object (insert a new node) where the extended undershoot intersects it (integer).  
1 = Break target (default)  
0 = Do not break target  
For any action other than 4, Extend Undershoots, CONVERT does not matter. |
Topology Variables

Topology variables store properties for topologies, which are created by `tpm_mntbuild` and the topology analyses functions, `tpm_analxx`.

Topology variables are a subset of the [configuration variables](#) data structure.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>BUILD_TOL</strong></td>
<td>Tolerance for topology building, buffering, and other processes (positive <a href="#">real</a>). Default = 0.01</td>
</tr>
<tr>
<td><strong>CNTR_COLOR</strong></td>
<td>Color for new centroids, an AutoCAD color index (<a href="#">integer</a>). Negative <a href="#">integer</a> = Current color. Default = -1. See <a href="#">Color Index Colors</a>.</td>
</tr>
<tr>
<td><strong>CNTR_LAYER</strong></td>
<td>Layer name for new centroids (<a href="#">string</a>). <code>nil</code> = Current layer. Default = &quot;&quot;.</td>
</tr>
<tr>
<td><strong>CNTR_TYPE</strong></td>
<td>Entity type of new centroids (<a href="#">integer</a>). 1 = Point (default). 2 = Block.</td>
</tr>
<tr>
<td><strong>CNTR_BLOCK</strong></td>
<td>Block name for new centroids (<a href="#">string</a>). Default = &quot;&quot;.</td>
</tr>
<tr>
<td><strong>CREATE_CNTR</strong></td>
<td>Flag for centroids generation (<a href="#">integer</a>). 1 = Generate (default). 0 = Do not generate.</td>
</tr>
<tr>
<td><strong>CREATE_MARKERS</strong></td>
<td>Whether errors should be marked with persistent markers as drawn by <code>tpm_cleanerrormark</code> (<a href="#">integer</a>). 0 = Do not mark (default). 1 = Mark. Marker shapes are set by <a href="#">MARKER_HEIGHT</a> and the <a href="#">XX_MARKER</a> cleanup variables.</td>
</tr>
<tr>
<td><strong>CREATE_NODE</strong></td>
<td>Flag for nodes generation (<a href="#">integer</a>). 1 = Generate (default).</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>CREATE_VIEW</td>
<td>Whether errors should be marked with temporary markers as drawn by <code>tpm_cleanerrordraw</code></td>
</tr>
<tr>
<td></td>
<td>0 = Do not mark</td>
</tr>
<tr>
<td></td>
<td>1 = Mark (default)</td>
</tr>
<tr>
<td>DEF_OFFSET</td>
<td>Default offset (real). Use when offset value cannot be calculated for an object, for example, if the object lacks necessary object data. Default = 1.0</td>
</tr>
<tr>
<td>DUPLICATE_CENTROID_COLOR</td>
<td>Marker color for duplicate centroid errors, an AutoCAD color index (integer). Default = 1 (red)</td>
</tr>
<tr>
<td></td>
<td>See Color Index Colors.</td>
</tr>
<tr>
<td>DUPLICATE_CENTROID_MARKER</td>
<td>Marker shape for duplicate centroid errors (integer). See Marker Shapes. Default = 4 (square)</td>
</tr>
<tr>
<td>IGNORE_INCOMPLETE_AREA</td>
<td>What to do if links are encountered that do not belong to any polygon (integer). 1 = Ignore them 0 = Cancel topology creation process (default)</td>
</tr>
<tr>
<td>INCOMPLETE_AREA_COLOR</td>
<td>Marker color for incomplete area errors, an AutoCAD color index (integer). Default = 2 (yellow)</td>
</tr>
<tr>
<td></td>
<td>See Color Index Colors.</td>
</tr>
<tr>
<td>INCOMPLETE_AREA_MARKER</td>
<td>Marker shape for incomplete area errors (integer). See Marker Shapes. Default = 2 (triangle)</td>
</tr>
<tr>
<td>INTERSECTION_COLOR</td>
<td>Marker color for intersection errors, an AutoCAD color index (integer). Default = 3 (green)</td>
</tr>
<tr>
<td></td>
<td>See Color Index Colors.</td>
</tr>
<tr>
<td>INTERSECTION_MARKER</td>
<td>Marker shape for intersection errors (integer). See Marker Shapes. Default = 1 (octagon)</td>
</tr>
<tr>
<td>MISSING_CENTROID_COLOR</td>
<td>Marker color for missing centroid errors, an AutoCAD color index (integer). Default = 4 (cyan)</td>
</tr>
<tr>
<td></td>
<td>See Color Index Colors.</td>
</tr>
<tr>
<td><strong>MISSING CENTROID_MARKER</strong></td>
<td>Marker shape for missing centroid errors (<strong>integer</strong>). See <a href="#">Marker Shapes</a>. Default = 3 (rhombus)</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>----------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>NODE_BLOCK</strong></td>
<td>Block name for new nodes (<strong>string</strong>). Default = &quot;&quot;</td>
</tr>
<tr>
<td><strong>NODE_COLOR</strong></td>
<td>Color for new nodes, an AutoCAD color index (<strong>integer</strong>): Negative <strong>integer</strong> = The current color Default = -1 See <a href="#">Color Index Colors</a></td>
</tr>
<tr>
<td><strong>NODE_LAYER</strong></td>
<td>Layer name for new nodes (<strong>string</strong>): <strong>nil</strong> The current layer Default = &quot;&quot;</td>
</tr>
<tr>
<td><strong>NODE_TYPE</strong></td>
<td>Entity type of new nodes (<strong>integer</strong>): 1 Point (default) 2 Block</td>
</tr>
<tr>
<td><strong>STOP_AT_MISSING_CNTR</strong></td>
<td>What to do if a polygon has no centroid (<strong>integer</strong>): 0 Create a centroid (default) 1 Cancel topology creation process</td>
</tr>
<tr>
<td><strong>STOP_AT_MULTIPLE_CNTR</strong></td>
<td>What to do if a polygon has more than one centroid (<strong>integer</strong>): 0 Designate one centroid 1 Cancel topology creation process (default)</td>
</tr>
</tbody>
</table>
Allocates a set of configuration variables.

```c
ade_id
tpm_varalloc();
```

Returns a configuration variables ID or ADE_NULLID.

Configuration variables are composed of cleanup variables, cleanup action variables, and topology variables. The variables are initialized to their default values. For a list of these variables and their default values, see Configuration Variables.

You can allocate more than one set of configuration variables.
tpm_varfree

Variables Functions

Frees a set of configuration variables.

```
int tpm_varfree(
    ade_id var_id);
```

Returns RTNORM or an error code.

**var_id**  Configuration variables ID.

Configuration variables are composed of cleanup variables, cleanup action variables, and topology variables. The variables are initialized to their default values. For a list of these variables and their default values, see [Configuration Variables](#).
tpm_varget

Variables Functions

**Get the value of a configuration variable.**

```c
int tpm_varget(
    ade_id var_id,
    char *var_name,
    struct *res);
```

Returns RTNORM or an error code.

<table>
<thead>
<tr>
<th>var_id</th>
<th>Configuration variables ID.</th>
</tr>
</thead>
<tbody>
<tr>
<td>var_name</td>
<td>Variable name.</td>
</tr>
<tr>
<td>res</td>
<td>Variable value.</td>
</tr>
</tbody>
</table>

Configuration variables are composed of cleanup variables, cleanup action variables and topology variables. The variables are initialized to their default values. For a list of these variables and their default values, see [Configuration Variables](#).

**Assigning And Reading Resbuf Values**

The following example demonstrates reading and writing a resbuf using `tpm_varget` and `tpm_varset`:

```c
struct resbuf rb;
ade_id var_id = ADE_NULLID;
int ret = 0;

// Correct ID
var_id = tpm_varalloc();

// Set real value
rb.restype = RTREAL;
rb.resval.rreal = 0.5;
rb.rbnest = NULL;
```
Sample Code

The following sample retrieves the value for the NODE_LAYER configuration variable using `tpm_varget()`. The topology containing the desired information is loaded then opened using `tpm_acload()` and `tpm_acopen()` respectively. A set of configuration variables is allocated using `tpm_varalloc()`, then filled using `tpm_infobuildvar()`. The topology can now be closed and unloaded using `tpm_acclose` and `tpm_acunload` respectively. A `resbuf` is created which will be populated with the configuration variable value using `tpm_varget()`, which is called with all required parameters. The return code is validated against `RTNORM` and appropriate status information is displayed. The `resbuf` is then released as required.

```c
char* pszTopoName = "netTopo";
int topoWriteAccess = 0;
int returnCode = tpm_acload(pszTopoName, 0);
ade_id topoId = tpm_acopen(pszTopoName, topoWriteAccess);
ade_id varId = tpm_varalloc();
returnCode = tpm_infobuildvar(topoId, varId);
tpm_acclose(topoId);
char* pszConfigVarName = "NODE_LAYER";
struct resbuf* pConfigVarsRb = acutBuildList(RTSTR, "", 0);
returnCode = tpm_varget(varId, pszConfigVarName, pConfigVarsRb);
```
if (RTNORM == returnCode){
    acutPrintf(
        "\nThe \"%s\" configuration variables value is: \"%s\"."
        , pszConfigVarName, pConfigVarsRb->resval.rstring);
}
else {
    acutPrintf(
        "\nThe specified configuration variable was not retrieved."));
}
acutRelRb(pConfigVarsRb);
Sets the value of a configuration variable.

```c
int tpm_varset(
    ade_id var_id,
    char *var_name,
    struct resbuf *valres);
```

Returns RTNORM or an error code.

- **var_id**: Configuration variables ID.
- **var_name**: Variable name.
- **valres**: Variable value contained in a resbuf.

Configuration variables are composed of cleanup variables, cleanup action variables, and topology variables. The variables are initialized to their default values. For a list of these variables and their default values, see Configuration Variables.

Examples of building a resbuf for the valres argument.

```c
rb = acutBuildList(RTSHORT, 4, 0);  // integer value
rb = acutBuildList(RTREAL, 1.0, 0); // real value
rb = acutBuildList(RTSTR, "0", 0);  // string value
```

You must release the resbuf.

The following sample allocates a set of configuration variables using tpm_varalloc(). A resbuf is created containing the layer name that will be associated with the LINK_LAYER configuration variable. Tpm_varset() is called with all required parameters and the return code is evaluated for the appropriate display message. The resbuf is then released as required.

```c
ade_id varId = tpm_varalloc();
char* pszConfigVarName = "LINK_LAYER";
struct resbuf* pConfigVarsRb = acutBuildList(RTSTR, "Links", 0);
```
int returnCode = tpm_varset(varId, pszConfigVarName, pConfigVarsRb);
if (RTNORM == returnCode){
    acutPrintf("\nThe specified configuration variable has been set.");
}
else {
    acutPrintf("\nThe specified configuration variable was not set.");
}
acutRelRb(pConfigVarsRb);
tpm_cleanactionlistgetat

Cleanup Functions

Gets the cleanup action at a given list position.

```c
int
*tpm_cleanactionlistgetat
    ade_id clean_var_id,
    long index,
    ade_id action_var_id);
```

Returns a cleanup action as a clean group type. See [tpm_cleangrouptype](#) for a list of types.

- **clean_var_id**: The cleanup variables ID returned by [tpm_varalloc](#).
- **index**: The list position to access.
- **action_var_id**: The cleanup action variables ID returned by [tpm_varalloc](#).

The **clean_var_id** argument references properties for the cleanup operation that you are preparing to initiate (see [Cleanup Variables](#)). These properties include the action list.

The **index** argument is a zero-based position in the action list. A value greater than or equal to the list size or less than 0 returns an error.

The **action_var_id** argument references properties affecting the specific cleanup action that you are getting (see [Cleanup Action Variables](#)). Use [tpm_varget](#) or [tpm_varlist](#) to read them after calling [tpm_cleanactionlistgetat](#).
tpm_infobuildvar

Topology Information Functions

Gets the configuration values of a topology.

```c
int
tpm_infobuildvar(
    ade_id tpm_id,
    ade_id var_id);
```

Returns RTNORM or an error code.

- **tpm_id**  Topology ID.
- **var_id**  Topology variables ID.

The topology variables ID references a set of topology variables in which to store the values that this function gets.

If no topology variables are allocated, call `tpm_varalloc` to allocate a set of them and return their ID.

To read the values that this function gets, use `tpm_varget` or `tpm_varlist`. To build a new topology using these variables, use `tpm_mntbuild`.

The following sample opens a topology, reads its configuration values using `tpm_infobuildvar()`, and gets its node layer. The topology is then closed, unloaded and the `resbuf` is released as required.

```c
char* pszTopoName = "I-95_Corridor";
int resultCode = tpm_acload(pszTopoName, 0);
ade_id topoId = tpm_acopen(pszTopoName, 0);
ade_id topoConfigVarId = tpm_varalloc();
resultCode = tpm_infobuildvar(
    topoId,
    topoConfigVarId);

char* pszConfigVarName = "NODE_LAYER";
struct resbuf* pConfigVarsRb = acutBuildList(RTSTR, ",", 0);
resultCode = tpm_varget(
    topoConfigVarId,

```
pszConfigVarName, pConfigVarsRb);
if (RTNORM == resultCode){
    acutPrintf(
        "The %s variable in the %s topology contains the value \"%s\".
        , pszConfigVarName, pszTopoName, pConfigVarsRb->resval.rstring);
} else {
    acutPrintf(
        "Could not determine the %s variable value."
        , pszConfigVarName);
}
resultCode = tpm_acclose(topoId);
resultCode = tpm_acunload(pszTopoName);
acutRelRb(pConfigVarsRb);
tpm_varlist

**Variables Functions**

Gets all the values in a set of configuration variables.

```
struct resbuf
*tpm_varlist(
    ade_id var_id);
```

Returns a `resbuf` of name-value pairs or `NULL`.

- **var_id**  
  Configuration variables ID.

Configuration variables are composed of cleanup variables, cleanup action variables, and topology variables. The variables are initialized to their default values. For a list of these variables and their default values, see [Configuration Variables](#).

The calling function must release the `resbuf`.

Each `resbuf` has this format:

- variable name, (RTSTR) . value, (RTSTR, RTREAL, RTSHORT)

The following sample allocates a set of configuration variables using `tpm_varalloc()`, then populates a `resbuf` with name-value pairs associated with the current configuration variable set using `tpm_varlist()`. The contents of the `resbuf` are displayed, then the `resbuf` is released as required.

```c
ade_id varId = tpm_varalloc();
struct resbuf* pConfigVarsRb = tpm_varlist(varId);
if (NULL != pConfigVarsRb){
    struct resbuf* rb = pConfigVarsRb;
    while(NULL != rb) {
        if (rb->restype == RTSTR) {
            acutPrintf(
                "The "%s" property contained the value:" ,
                rb->resval.rstring);
        if (NULL != (rb = rb->rbnext)) {
            switch(rb->restype)
```
{  
case RTSTR:
    acutPrintf(" \
"%s"
        , rb->resval.rstring);
    break;
  case RTREAL:
    acutPrintf(" %.2lf"
        , rb->resval.rreal);
    break;
  case RTSHORT:
    acutPrintf(" %d"
        , rb->resval.rint);
    break;
  default:
    break;
  }
}

rb = rb->rbnext;
}

else {
    acutPrintf("\nThe configuration variable list could not been retrieved.");
}

acutRelRb(pConfigVarsRb);
The arrow buttons are for navigation.

Click to navigate down the tree.

Click to navigate up the tree (toward AcMapSession).

The chevron button links to more information.

Click to pop up a note.

Not all classes are members of AutoCAD Map.

NamesInBlack  AutoCAD Map ObjectARX classes. Click a class name to open its Help topic.

NamesInBlue   AutoCAD ObjectARX classes. Refer to AutoCAD ObjectARX Help.

Containment is managed by functions of the containing classes. For example, AcMapSession contains an AcMapAliases object through the AcMapSession::GetAliases function, and AcMapAliases in turn contains one or more AcMapDriveAlias objects through the AcMapAliases::GetAlias function. AcMapAliases has other functions as well for managing containment: functions to add, count, remove, and so on.
Object Model: AcMapSession
Object Model: AcMapODContainer

Inheritance  Legend
Object Model: AcMapQuery

Inheritance  Legend
Inheritance: Query Classes

Object Model

Diagram:
- AcMapQueryUnit
  - AcMapQueryBranch
  - AcMapQueryCondition
    - AcMapDataCondition
    - AcMapLocationCondition
    - AcMapSQLCondition
    - AcMapPropertyCondition
- AcMapPropertyAlteration
  - AcMapHatchAlteration
  - AcMapTextAlteration
Inheritance: Boundary Classes

Object Model

```
AcMapLocationBoundary
  AcMapFenceBoundary
  AcMapAllBoundary
  AcMapPointBoundary
  AcMapPolyLineBoundary

AcMapBufferFenceBoundary
AcMapPolygonBoundary
AcMapWindowBoundary
AcMapCircleBoundary
AcMapClosedPolyLineBoundary
AcMapBufferPolyLineBoundary
```
Header file: MapSession.h.

An instance of the AcMapSession class represents a Map session, which is a single instance of AutoCADMap that is currently running.

An AcMapSession contains the session's error stack (AcMapErrorStack class), its aliases (AcMapAliases class), its projects (AcMapProject class) and its project iterator (AcMapProjectIterator class).

Do not subclass from this class.

To get an AcMapGetSession pointer, call AcMapGetSession, which is declared an extern C function in MapSession.h. AutoCAD Map releases the memory allocated for AcMapGetSession, so do not attempt to delete it yourself.

The following example implements a command that declares a pointer to an AutoCAD Map session, and immediately sets the pointer to NULL. The example calls AcMapGetSession to get the AcMapSession pointer. For simplicity, the code does not show routine error checking.

```c
#include "StdAfx.h"
#include "StdArx.h"
#include <MapArxApi.h>

void asdkhelpcreatesession()
{
    AcMapSession *mapApi = NULL;
    mapApi = AcMapGetSession();
    if (mapApi == NULL)
    {
        acutPrintf ("Can't connect to MAP");
        return;
    }
    else
    {
        acutPrintf("Created AcMapSession");
        acedRetVoid();
    }
}
```
Header file: MapReactors.h.

The AcMapSessionOptionsReactor class is used to represent a session options reactor, which notifies an application when a session option is modified.

To create a specific project options reactor, subclass from AcMapSessionOptionsReactor.

Use AcMapSession::AddOptionsReactor to add an options reactor to a session and AcMapSession::RemoveOptionsReactor to remove it.

You use the new operator to get an AcMySessionOptionsReactor pointer, and call AcMapSession::RemoveOptionsReactor to deallocate memory, as shown in the following example. For simplicity, routine error checking is not shown.

```c++
class AcMySessionOptionsReactor : public AcMapSessionOptionsReactor
{
    // See MapDemoApp.cpp distributed with AutoCAD Map ObjectARX.
};

static AcMySessionOptionsReactor *pSysOptReactor;

AcRx::AppRetCode acrxEntryPoint(AcRx::AppMsgCode msg, void *appId)
{
    AcMapSession *mapApi = NULL;
    switch (msg)
    {
    case AcRx::kInitAppMsg:
        acrxRegisterAppMDIAware(appId);
        //register commands -- code not shown
        pSysOptReactor = new AcMySessionOptionsReactor();
        mapApi = AcMapGetSession();
        if (mapApi)
            mapApi->AddOptionsReactor(pSysOptReactor);
        break;
    case AcRx::kUnloadAppMsg:
        //remove commands -- code not shown
        break;
    case AcRx::kUnloadDwgMsg:
        mapApi = AcMapGetSession();
        if (mapApi)
        {
            AcMapProject *pProj;
            if (mapApi->GetProject(pProj))
                mapApi->RemoveOptionsReactor(pSysOptReactor);
        }
        break;
    }
    return AcRx::kRetOK;
}
```
Header file: MapReactors.h.

The AcMapSessionReactor class is used to represent a session reactor, which notifies an application of session events.

To create a specific session reactor, subclass from AcMapSessionReactor.

Use AcMapSession::AddSessionReactor to add a session reactor to a session and AcMapSession::RemoveSessionReactor to remove it. Delete the pointer to the reactor using the delete operator when you unload the application.
AcMapProject class

Functions  Object Model  See Also

Header file: MapProj.h.

Note This class has new functions added for AutoCAD Map 3D 2005.

An instance of the AcMapProject class, a project, is the principal container for objects used in AutoCAD Map. Within a Map session, a user can open and close multiple projects.

The project contains objects of the following classes:

- AcMapDrawingSet class
- AcMapRangeLibrary class
- AcMapQueryLibrary class
- AcMapSaveSet class
- AcMapODContainer class
- AcMapExpression class
- AcMapQuery class

You cannot instantiate a project programmatically. A project is instantiated when a Map user opens a document. You can get a pointer to a project with the AcMapSession::GetProject function or with the Project functions of various classes. Do not delete memory associate with the project pointer.
Header file: MapProj.h.

An instance of the AcMapProjectIterator class represents a project iterator, which is used to iterate through the projects in a session.

Do not subclass from this class.

You get an AcMapProjectIterator pointer by calling AcMapSession::GetProjectIterator. You are responsible for deleting memory allocated for the iterator using the delete operator.

The following code fragment shows how to create and delete a project iterator. For simplicity, routine error checking is not shown.

```c
AcMapSession *mapApi = NULL;
mapApi = AcMapGetSession();
cMapProjectIterator *pIterator = mapApi->GetProjectIterator();
AcMapProject *pProject;
for (; !pIterator->Done(); pIterator->Next())
{
    pIterator->GetCurrentObject(pProject);
    ...
}
delete pIterator;
```
The AcMapProjectOptionsReactor class is used to represent a project options reactor, which notifies an application when a project option is modified.

To create a specific project options reactor, subclass from AcMapProjectOptionsReactor.

Use AcMapProject::AddOptionsReactor to add an options reactor to a project and AcMapProject::RemoveOptionsReactor to remove it.

For simplicity, the following example does not show routine error checking.

```cpp
class AcMyProjectOptionsReactor : public AcMapProjectOptionsReactor
{
    // See MapDemoApp.cpp distributed with AutoCAD Map ObjectARX.
};

static AcMyProjectOptionsReactor *pPrefReact;

AcRx::AppRetCode acrxEntryPoint(AcRx::AppMsgCode msg, void *appId)
{
    AcMapSession *mapApi = NULL;
    switch (msg)
    {
    case AcRx::kInitAppMsg:
        acrxRegisterAppMDIAware(appId);
        // register commands -- code not shown
        pPrefReact = new AcMyProjectOptionsReactor();
        mapApi = AcMapGetSession();
        if (mapApi)
            pProj->AddOptionsReactor(pPrefReact);
        break;
    case AcRx::kUnloadAppMsg:
        // remove commands -- code not shown
        break;
    case AcRx::kUnloadDwgMsg:
        mapApi = AcMapGetSession();
        if (mapApi)
        {
            AcMapProject *pProj;
            if (mapApi->GetProject(pProj))
                pProj->RemoveOptionsReactor(pPrefReact);
        }
        break;
    }
    return AcRx::kRetOK;
}
The AcMapAliases class represents a list of drive aliases that have been defined for the session. It provides support for the AcMapDriveAlias class, whose instances represent the individual drive aliases used in an AutoCAD Map session.

You must create a drive alias for a drawing before it can be attached.

Do not subclass from this class or delete the AcMapAliases pointer.

You get an pointer to the session's aliases by calling AcMapSession::GetAliases as shown in the following example, which implements a command called accessaliases. For simplicity, the code does not show routine error checking.

```cpp
#include "StdAfx.h"
#include "StdArx.h"
#include <MapArxApi.h>

void asdkhelpaccessaliases()
{
    AcMapSession *mapApi = NULL;
    AcMapAliases *pAliases = NULL;

    mapApi = AcMapGetSession();
    mapApi->GetAliases(pAliases);
    // Use pAliases to access functions of AcMapAliases -- code not shown
    acedRetVoid();
}```
The `AcMapAliasesReactor` class is used to represent a drive alias reactor, which notifies an application of drive alias events.

To create a specific drive aliases reactor, subclass from `AcMapAliasesReactor`.

Use `AcMapAliases::AddReactor` to add the reactor to a drive alias and `AcMapAliases::RemoveReactor` to remove it. Delete the pointer to the reactor using the delete operator when you unload the application, as shown in the following example. For simplicity, error checking is not shown.

```cpp
class AcMyAliasesReactor : public AcMapAliasesReactor
{
    // See MapDemoApp.cpp distributed with AutoCAD Map ObjectARX.
};

static AcMyAliasesReactor *pAliasReact;

AcRx::AppRetCode acrxEntryPoint(AcRx::AppMsgCode msg, void *appId)
{
    AcMapSession *mapApi = NULL;
    switch (msg) {
    case AcRx::kInitAppMsg:
        acrxRegisterAppMDIAware(appId);
        // register commands -- code not shown
        pAliasReact = new AcMyAliasesReactor();
        mapApi = AcMapGetSession();
        AcMapAliases *pAlias = NULL;
        mapApi->GetAliases(pAlias);
        pAlias->AddReactor(pAliasReact);
        break;
    case AcRx::kUnloadAppMsg:
        delete pAliasReact;
        pAliasReact = NULL;
        break;
    case AcRx::kUnloadDwgMsg:
        mapApi = AcMapGetSession();
        AcMapAliases *pAlias = NULL;
        mapApi->GetAliases(pAlias);
        pAlias->RemoveReactor(pAliasReact);
        break;
    }
    return AcRx::kRetOK;
}
```
Header file: MapArxApi.h.

An instance of the AcMapDriveAlias class represents a drive alias used in an AutoCAD Map session.

A drive alias allows you to assign a directory or network path string to a single AutoCAD name, such as a single drive letter, and to substitute the alias name for the directory path string.

The AcMapAliases class contains the AcMapDriveAlias class. You create an AcMapDriveAlias object using the AcMapAliases::AddAlias and FindAlias function. You are responsible for deleting memory allocated for the AcMapDriveAlias using the delete operator.

Do not subclass from this class.
Header file: MapArxApi.h.

**Note** This class has *new functions* added for AutoCAD Map 3D 2005.

An attached drawing, also called a source drawing.

An attached drawing can contain other attached drawings nested within it.

Do not subclass from this class. You create an attached drawing using the AcMapDrawingSet::GetDrawing or AcMapDrawingSet::AttachDrawing. These functions return a pointer by reference to the attached drawing that you are responsible for deleting using the delete operator, as shown in the example.

The following example creates a command that gets a pointer to an attached drawing called, demo.dwg. Delete the memory allocated for the attached drawing, using the delete operator as shown here. The example assumes that you already attached the drawing and its location is specified by ALIAS. For simplicity, routine error checking is not shown.

```cpp
#include "StdAfx.h"
#include "StdArx.h"
#include <MapArxApi.h>
#include <MapProj.h>
#include <MapStringArray.h>

void asdkhelpgetdwg()
{
    AcMapAttachedDrawing *pDwg = NULL;
    AcMapSession *mapApi = NULL;
    AcMapProject *pProj = NULL;
    AcMapDrawingSet *pDSet = NULL;

    mapApi = AcMapGetSession();
    mapApi->GetProject(pProj);
    pProj->GetDrawingSet(pDSet);
    pDSet->GetDrawing(pDwg, "ALIAS:\demo.dwg", Adesk::kTrue);
    // do something with pDwg
    delete pDwg;
    acedRetVoid();
}
```
Header file: MapArxApi.h.

The attached drawings of a project. A project contains a single drawing set.

A drawing set is a tree. Some drawings are attached directly to the drawing set, and each attached drawing can have other attached drawings nested within it.

To alter a drawing set, a user must have the Alter Drawing Set user privilege. Set this privilege in the AutoCAD Map User Administration dialog.

Do not subclass from this class.

To get an AcMapDrawingSet pointer, call AcMapProject::GetDrawingSet, as shown in the following example. Do not attempt to delete the memory associated with the drawing set pointer. For simplicity, the example does not show routine error checking.

```c
AcMapSession *mapApi = NULL;
AcMapProject *pProj = NULL;
AcMapDrawingSet *pDSet = NULL;

mapApi = AcMapGetSession();
mapApi->GetProject(pProj);
pProj->GetDrawingSet(pDSet);
pDSet->ZoomExtents();
```
The AcMapDrawingSetReactor class is used to represent a drawing set reactor, which supplies callbacks to notify an application immediately of drawing set events.

To create a specific drawing set reactor, subclass from AcMapDrawingSetReactor.

Use AcMapDrawingSet::AddReactor to add a reactor to a drawing set and AcMapDrawingSet::RemoveReactor to remove it.

Use AcMapDrawingSet::AddReactor to add the reactor to a drawing set and AcMapDrawingSet::RemoveReactor to remove it. Delete the pointer to the reactor using the delete operator when you unload the application, as shown in the following example. For simplicity, error checking is not shown.

```cpp
class AcMyDrawingSetReactor : public AcMapDrawingSetReactor
{
    // See MapDemoApp.cpp distributed with AutoCAD Map ObjectARX.
};

static AcMyDrawingSetReactor *pDSetReact;

AcRx::AppRetCode acrxEntryPoint(AcRx::AppMsgCode msg, void *appId)
{
    AcMapSession *mapApi = NULL;
    switch (msg) {
    case AcRx::kInitAppMsg:
        acrxRegisterAppMDIAware(appId);
        // register commands -- code not shown
        pDSetReact = new AcMyDrawingSetReactor();
        mapApi = AcMapGetSession();
        AcMapDrawingSet *pDSet = NULL;
        pProj->GetDrawingSet(pDSet);
        pDSet->AddReactor(pDSetReact);
        break;
    case AcRx::kUnloadAppMsg:
        delete pDSetReact;
        pDSetReact = NULL;
        break;
    case AcRx::kUnloadDwgMsg:
        mapApi = AcMapGetSession();
        AcMapDrawingSet *pDSet = NULL;
        pProj->GetDrawingSet(pDSet);
        pDSet->RemoveReactor(pDSetReact);
        break;
    }
    return AcRx::kRetOK;
}
The `AcMapObjectId` class implements an AutoCAD Map database identifier. This class provides identifiers for AutoCAD entities which are persistent across database load and unload operations.

An AutoCAD Map database identifier consists of an object handle and the identifier of the source drawing to which the object belongs.

Do not subclass from this class.
AcMapExpression class

Functions Object Model See Also

Header file: MapArxApi.h.

Note This class has new functions added for AutoCAD Map 3D 2005.

An instance of the AcMapExpression class is an expression, which you can use throughout an AutoCAD Map project. You use AcMapExpression to define, set, get, and execute an expression or manipulate template lines. An expression is a combination of operands and operators that evaluates to a single value. The same expression always evaluates to the same value. For more information about expressions, click Expression Evaluator on the Contents tab of AutoCAD Map Help.

Do not subclass from this class.

You call AcMapProject::DefineExpression to get an AcMapExpression pointer. You are responsible for deleting the memory allocated for the object using the delete operator, as shown in the following example. For simplicity, the code does not show routine error checking.

```cpp
char buff[132] ;
AcMapSession *mapApi = NULL ;
AcMapProject *pProj = NULL ;
AcMapExpression *pExpress = NULL ;
AcDbObjectId Id = 0 ;

AcMapValue Val ;
mapApi = AcMapGetSession();
mapApi->GetProject(pProj);
*buff = EOS ;
acedGetString (1, "Enter an expression: ", buff);
pProj->DefineExpression(pExpress, buff) ;
// Use the expression -- code not shown
delete pExpress ;
```
This class is defined in MapODColumn.h.

The AcMapODColumnDefinition class is used to create a column definition, which can then be passed to AcMapODTableDefinition::AddColumn to build a table definition. The column definition contains the column's data type, name, optional description, and default value.

Do not subclass from this class.

To create a column definition, you instantiate this class using the new operator and release memory allocated for it using the delete operator, as shown in the following example. For simplicity, routine error checking is not shown.

```cpp
AcMapSession *mapApi = NULL;
AcMapProject *pProj = NULL;
AcMapODContainer *pODCont = NULL;
AcMapODTableDefinition *pTabDef = NULL;
AcMapODColumnDefinition *pColDef1 = NULL;

mapApi = AcMapGetSession();
mapApi->GetProject(pProj);
pProj->GetODContainer(pODCont);
pTabDef = new AcMapODTableDefinition();
pColDef1 = new AcMapODColumnDefinition();
pColDef1->SetName("Residential");
pColDef1->SetDescription("Residential R1-R3");
pColDef1->SetType(AcMap::kCharacter);
pColDef1->SetDefaultValue("R1");
pTabDef->AddColumn(*pColDef1);
pODCont->CreateODTable("MinimalTable", *pTabDef, "Simple table", Adesk::kFalse);
if (pColDef1) delete pColDef1;
if (pTabDef) delete pTabDef;
```
This class is defined in MapArxApi.h.

Object data about a source drawing is stored in object data tables that are part of the drawing. The ODContainer class serves as a container for object data tables and their components.

The AutoCAD Map ObjectARX API creates an instance of AcMapODContainer when you open a project. You get a pointer to this instance by calling AcMapProject::GetODContainer to access the project's object data tables. Do not attempt to delete this pointer.

Do not subclass from this class.

The following code fragment gets a pointer to the instance of AcMapODContainer and accesses the project object data tables with it. For simplicity, routine error checking is not shown.

```c
AcMapSession *mapApi = NULL;
AcMapProject *pProj = NULL;
AcMapODContainer *pODCont = NULL;
AcMapODTable *pODTable = NULL;

mapApi = AcMapGetSession();
mapApi->GetProject(pProj);
pProj->GetODContainer(pODCont);
pODCont->GetODTable(pODTable, "Zones");
```
This class is defined in MapODIterator.h.

An instance of the AcMapODRecordIterator class, an object record iterator, is used to iterate through the records in an object data table.

Do not subclass from this class.
AcMapODTable class

This class is defined in MapODTable.h file.
The AcMapODTable class is used to manage object data tables.
Do not subclass from this class.
This class is defined in MapODDefinition.h.

The AcMapODTableDefinition class is used to create a table definition, which can then be passed to AcMapODContainer::CreateODTable to build an object data table. The table definition defines the object data table's columns. You must first add columns to the table definition creating the table.

Do not subclass from this class.

To create a table definition, instantiate this class using the new operator and delete memory allocated for it using the delete operator, as shown in the following example. For simplicity, routine error checking is not shown.

```cpp
AcMapSession *mapApi = NULL;
AcMapProject *pProj = NULL;
AcMapODContainer *pODCont = NULL;
AcMapODTableDefinition *pTabDef = NULL;
AcMapODColumnDefinition *pColDef1 = NULL;
mapApi = AcMapGetSession();
mapApi->GetProject(pProj);
pProj->GetODContainer(pODCont);
pTabDef = new AcMapODTableDefinition();
pColDef1 = new AcMapODColumnDefinition();
pColDef1->SetName("Residential");
pColDef1->SetDescription("Residential R1-R3");
pColDef1->SetType(AcMap::kCharacter);
pColDef1->SetDefaultValue("R1");
pTabDef->addColumn(*pColDef1);
pODCont->CreateODTable("MinimalTable", *pTabDef, "Simple table",
Adesk::kFalse);
if (pColDef1) delete pColDef1;
if (pTabDef) delete pTabDef;
```
AcMapODTableRecord class

This class is defined in MapODRecord.h.
An instance of the AcMapODTableRecord class represents a record in an object data table.
Do not subclass from this class.
This class is defined in MapValue.h.

An instance of the AcMapValue class represents a value in an object data table record.

The value can be integer, double, char *, or point. The value's initial data type is set when the first value is assigned to it. The AcMapValue class has functions for explicit data type casting.

The number and data types of AcMapValue objects that are contained by an AcMapODTableRecord object must correspond with the number and data types of AcMapODColumnDefinition objects that are contained by the corresponding AcMapODTableDefinition object.
Class: MapArxApi.h.

Note This class has new functions added for AutoCAD Map 3D 2005.

An instance of the AcMapQuery class represents a defined query.
Do not subclass from this class.

Creating a Query

The AcMapQuery class does not have a constructor, and you cannot create and modify its objects directly. After you have created the query definition, instantiate an AcMapQuery object using the AcMapProject::CreateQuery function. This creates the query object in the project, making the query available to the application. You are responsible for deleting the query, as shown in the following example.

```c++
// For simplicity, error checking is not shown.
#include "StdAfx.h"
#include "StdArx.h"
#include <MapArxApi.h>
#include <MapProj.h>
#include <MapQuery.h>

AcMapSession *mapApi = NULL;
AcMapProject *pProj = NULL;
AcMapQuery *pQuery = NULL;

mapApi = AcMapGetSession();
mapApi->GetProject(pProj);
pProj->CreateQuery(pQuery, Adesk::kTrue);
// define and execute the query -- code not shown
delete pQuery;
```

Defining a Query

A query definition is built from an AcMapQueryBranch object, which describes the conditions that form the selection criteria of the query. Create the query branch and pass it to the AcMapQuery::Define function to create a query definition. See AcMapQueryBranch class for information about creating the query branch.

Modifying a Query

To modify an existing query, create a new query branch. Then call AcMapQuery::Define to redefine the query using the new query branch.

Executing a Query

Before executing a query, you may also want to set the mode (using AcMapQuery::SetMode), enable or disable property alteration (using AcMapQuery::EnablePropertyAlteration) or create a report template (using AcMapQuery::GetReportTemplate) for the query. before executing a query. Then call AcMapQuery::Run to execute the query.
The following example, which assumes drawings are attached, shows how to create a command that creates, defines, and executes a query. For simplicity, error checking is not shown.

```cpp
#include "StdAfx.h"
#include "StdArx.h"
#include <MapArxApi.h>
#include <MapProj.h>
#include <MapQuery.h>
#include <MapBoundary.h>

void askhelpqueryall()
{
  AcMapSession  *mapApi = NULL;
  AcMapProject  *pProj = NULL;
  AcMapDrawingSet *pDSet = NULL;
  AcMapQueryBranch *pQBranch = NULL;
  AcMapLocationCondition *pLocationCondition = NULL;
  AcMapQuery *pNewQuery = NULL;

  mapApi = AcMapGetSession();
  mapApi->GetProject(pProj);
  pProj->GetDrawingSet(pDSet);
  pQBranch = new AcMapQueryBranch();
  pLocationCondition = new AcMapLocationCondition(AcMap::kOperatorAnd,
                                                AcMap::kLocationInside);
  pProj->CreateQuery(pNewQuery, Adesk::kFalse);
  pNewQuery->Clear(Adesk::kTrue);
  pNewQuery->SetMode(AcMap::kQueryDraw);
  pQBranch->AppendOperand(pLocationCondition);
  pNewQuery->Define(pQBranch);
  pNewQuery->Run();
  pDSet->ZoomExtents();
  delete pLocationCondition;
  delete pQBranch;
  delete pNewQuery;
  aecedRetVoid();
}
```

**Error Codes**

If the return value of a query function is of type AcMap::EErrCode and the function succeeds, it returns AcMap::kOk. Error codes denoting unsuccessful conditions are listed under AcMap::EErrorCode.
This class is defined in MapArxApi.h.
The AcMapQueryAttribute class is used to represent a query that has been saved either with a project or in an external file. This is different from the AcMapQuery class, which is used to represent a query in memory.
To create an AcMapQueryAttribute object, pass NULL as the query to AcMapQueryCategory::GetQuery. Do not subclass from this class.
Header file: MapQuery.h.

The AcMapQueryBranch class is derived from the AcMapQueryUnit class. It is used by the AcMapQuery::Define to build or modify a query definition.

You get an AcMapQueryBranch pointer using the new operator and you are responsible for deleting the allocated memory using the delete operator.

A query branch is a tree constructed of one or more query units. Each query unit in the query branch can be either a subordinate query branch or an instance of one of the following condition classes: AcMapLocationCondition, AcMapSQLCondition, AcMapDataCondition, and AcMapPropertyCondition. The conditions, which hang off the query branch, describe the query’s selection criteria. Structurally, a condition is a simple node in the query branch.

Do not subclass from this class.
This class is defined in MapArxApi.h.

The AcMapQueryCategory class is used to represent a query category, which is used as a container for saved queries in a query library.

Do not subclass from this class.
Header file: MapQuery.h.

A query unit represents either a query condition or a subordinate query branch. It is sometimes referred to as a query node or a query operand.

Every query unit contains a `= 4)` BSPSPopupOnMouseOver(event);` join operator which is used to combine query units logically. The join operator of the first query unit within a query branch is ignored.

Changes to a query unit affect the query only after a call to the AcMapQuery::Define function.

The AcMapQueryBranch and AcMapQueryCondition classes are derived from the AcMapQueryUnit class. Do not subclass from this class.
This class is defined in MapTemplate.h.

The AcMapReportTemplate class is used to represent a report template, which AutoCAD Map uses to format the output of a query in a text file.

A report template contains one or more column definitions, also called template lines.

Do not subclass from this class.
AcMapTemplateLine class

Functions  Object Model  See Also

This class is defined in MapTemplate.h.

The AcMapReportTemplateLine class is used to represent a column definition in a report template. The definition is contained in the column definition expression.

Do not subclass from this class.
Header file: MapArxApi.h.

When you edit objects in a project, your changes are saved to the project file (DWG file) by default. You can also use a save set to save changes back to their source drawings. The save set enables users to make changes to their drawings persistent.

The AcMapSaveSet class is used to create a save set, which contains the drawing database identifiers of the drawing objects whose changes are saved back to their source drawing files.

A project has a single save set. Create a save set with the AcMapProject::GetSaveSet function.

Do not subclass from this class.

During an edit session in a network environment, where the administrator has set the object locking preference, AutoCAD Map locks those objects in source drawing files that are specified in a save set so that other users cannot modify the objects.
AcMapQueryLibrary class

This class is defined in MapArxApi.h.

The AcMapQueryLibrary class is used to represent a query library, which AutoCAD Map uses to save queries with a project. The queries can be saved with the project or in an external file.

A query library is created automatically when a project is created. There is only one query library per project.

Within a query library, queries are be organized into an dynamic array of query catagories.

Do not subclass from this class.
The `AcMapQueryLibraryReactor` class is used to represent a query library reactor, which notifies an application of query library events.

To create a specific query library reactor, subclass from `AcMapQueryLibraryReactor`.

Use `AcMapQueryLibrary::AddReactor` to add a reactor to a query library and `AcMapQueryLibrary::RemoveReactor` to remove it.
The AcMapQueryCondition class is derived from the AcMapQueryUnit class. It is the base class from which the following condition classes are derived:

- `AcMapSQLCondition`
- `AcMapLocationCondition`
- `AcMapDataCondition`
- `AcMapPropertyCondition`

Do not subclass from this class.
Header file: MapQuery.h.

A data condition is used to select drawing objects based on object data, which is non-graphic data stored about the drawing in the drawing file.

A data condition contains a join operator, a data type, a condition operator, and a value. In addition, an associated value expression specifies the table and column in which the drawing’s object data is stored, so that the object data can be attached before the query is executed.

Drawing objects whose related object data values meet the conditions of the data condition expression are selected. For example, to select drawing objects that represent pipes of a diameter of one inch, create a data condition that queries a table called PIPES and select those drawing objects whose DIAMETER column value = 1.0.

After you have built the data condition, add it to the query branch using the AcMapQueryBranch::AppendOperand, AcMapQueryBranch::InsertOperand, or AcMapQueryBranchSubstituteOperand function. Since these functions make a copy of the AcMapDataCondition object, you can release the data condition after adding it to the query branch.

The AcMapDataCondition class is derived from the AcMapQueryCondition class.

Do not subclass from this class.
Header file: MapQuery.h.

A location condition is used to select drawing objects based on their spatial location in a drawing.

A location condition contains a join operator (AND or OR), a location type (either Inside or Crossing), and a boundary.

There are many types of boundaries: location boundaries, all boundaries, circle boundaries, fence boundaries, buffer fence boundaries, polygon boundaries, window boundaries, polyline boundaries, buffer polyline boundaries, and closed polyline boundaries.

Inside and Crossing are not appropriate with all types of boundaries. For example, you cannot use Inside with a fence boundary, only Crossing.

After you have created the location condition, add it to a query branch using the AcMapQueryBranch::AppendOperand, AcMapQueryBranch::InsertOperand, or AcMapQueryBranchSubstituteOperands. Since these functions make a copy of the AcMapLocationCondition object, you can release the location condition after adding it to the query branch.

After you define the query with the query branch, the query will select drawing objects that lie inside or cross the boundary, depending on the location condition’s location type.

The AcMapLocationCondition class is derived from the AcMapQueryCondition class.

Do not subclass from this class.
Header file: MapQuery.h.

A property condition is used to select drawing objects based on their properties in the drawing.

A property condition contains a join operator (AND or OR), a property type, a property value, and a comparative operator, and a value. Drawing objects whose property values meet the conditions of the property condition expression are selected by the query. For example, in the condition

```
COLOR = 'RED' AND ELEVATION > 50
```

the join operator is AND, the property types are COLOR and ELEVATION, the comparative operators are = and >, and the values are "RED" and 50.

After you create a property condition, add it to a query branch using the AcMapQueryBranch::AppendOperand, the AcMapQueryBranch::InsertOperand, or AcMapQueryBranch::SubstituteOperand function. Since these functions make a copy of the AcMapPropertyCondition object, you can release the property condition after adding it to the query branch.

The AcMapPropertyCondition class is derived from the AcMapQueryCondition class. Do not subclass from this class.
AcMapSQLCondition class

Header file: MapQuery.h.

An SQL condition is used to select drawing objects based on nongraphic data that is stored in an external database.

An SQL condition contains a \texttt{= 4) BSPSPopupOnMouseOver(event);} \texttt{;">join operator} and an SQL \texttt{WHERE} condition. In addition, an SQL condition specifies an associated link path name, called a \texttt{= 4) BSPSPopupOnMouseOver(event);} \texttt{;">link template}, to the external database in which the data is stored, so that the drawing and its related records can be compared during query execution.

After you have created an SQL condition, add it to a query branch using the AcMapQueryBranch::AppendOperand, AcMapQueryBranch::InsertOperand, or AcMapQueryBranch::SubstituteOperand function. Since these functions make a copy of the AcMapSQLCondition object, you can release the SQL condition after adding it to the query branch.

After you define the query with the query branch, the query will select drawing objects whose related database record values meet the conditions of the SQL \texttt{WHERE} condition. For example, to select all drawing objects that represent pressure valves, you might create a SQL condition such as VALVE_TYPE = 'PRESSURE'. The \texttt{WHERE} keyword is always omitted from such a statement.

The AcMapSQLCondition class is derived from the AcMapQueryCondition class.

Do not subclass from this class.
AcMapAllBoundary class

Header file: MapBoundary.h.

The AcMapAllBoundary class is derived from the AcMapLocationBoundary class. It represents the All Boundary condition. Use an object of the AcMapAllBoundary class to select all the objects in the active drawing.

Do not subclass from this class.
Header file: MapBoundary.h.

The AcMapBufferFenceBoundary class defines a buffer fence, which is a series of points connected by straight lines, surrounded by a buffer with a specified width. The buffer fence boundary is defined by the coordinates of its points and the width of its buffer. There must be a minimum of two distinct points. If the width of the buffer is 0, the buffer fence boundary is treated as a fence boundary.

A query built on a buffer fence boundary can select all drawing objects that cross or lie inside the boundary.

The AcMapBufferFenceBoundary class is derived from the AcMapFenceBoundary class. To get or set buffer fence boundary points, use the AcMapFenceBoundary::GetPoints or AcMapFenceBoundary::SetPoints function.

Do not subclass from this class.

You create an AcMapBufferFenceBoundary pointer using the new operator and delete allocated memory using the delete operator.
AcMapBufferPolylineBoundary class

Header file: MapBoundary.h.

The AcMapPolylineBoundary defines a buffer polyline boundary, which is one or more connected line segments or circular arcs, all surrounded by a buffer with a specified width. If the width of the buffer is 0, the buffer polyline boundary is treated as a polyline boundary.

A query can select drawing objects that either cross or lie inside the buffer polyline boundary.

Do not subclass from this class.

You get an AcMapBufferPolylineBoundary pointer using the new operator and you are responsible for deleting allocated memory using the delete operator.
Header file: MapBoundary.h.

The AcMapCircleBoundary represents a circle with a center point and a radius.

The AcMapCircleBoundary class is derived from the AcMapPointBoundary class. To get or set the circle’s center, use AcMapPointBoundary::GetPoint or AcMapPointBoundary::SetPoint function.

Do not subclass from this class.

You get an AcMapCircleBoundary pointer using the new operator, so you are responsible for deleting allocated memory with the delete operator.
Header file: MapBoundary.h.

The AcMapPolylineBoundary defines a closed polyline boundary, which is a series of points connected by straight lines and arcs that define a contiguous area. A query can select drawing objects that either cross or lie inside the closed polyline boundary.

The AcMapClosedPolylineBoundary class is derived from the AcMapPolylineBoundary class.

Objects of the AcMapClosedPolylineBoundary class are fully enclosed, in contrast to an AcMapPolylineBoundary object, which may or may not be closed.

Do not subclass from this class.

You get an AcMapClosedPolylineBoundary pointer using the new operator and you are responsible for deleting allocated memory using the delete operator.
Header file: MapBoundary.h.

The AcMapFenceBoundary class defines a fence, which is a series of points connected by straight lines. The fence boundary is defined by the coordinates of its points. There must be a minimum of two distinct points.

A query built on a fence boundary can select all drawing objects that cross the fence boundary. The AcMapFenceBoundary class is derived from the AcMapLocationBoundary class.

The AcMapBufferFenceBoundary, AcMapPolygonBoundary, and AcMapWindowBoundary classes are derived from the AcMapFenceBoundary class.

Do not subclass from this class.

You get an AcMapFenceBoundary pointer using the new operator, so you are responsible for deleting allocated memory with the delete operator.
Header file: MapBoundary.h.

The AcMapLocationBoundary class is an abstract base class from which all other location boundary classes are derived. Classes derived from the AcMapLocationBoundary class are used to create instances of different types of boundaries. Boundary types include all boundaries, fence, circle, point, polygon, polyline, and window.

A location condition displays objects based on location relative to a specified boundary, represented by an instance of a location boundary subclass. The query can specify whether objects must be completely inside the boundary or cross and have any part inside the boundary.

To use a location boundary in a query, instantiate an object of one of the AcMapLocationBoundary subclasses, associate it with a location condition using the AcMapLocationCondition::SetBoundary function, and add the location condition to a query branch.

Do not subclass from this class.
The AcMapPointBoundary represents the point boundary condition, which is used to define a point and then build a query that selects all the objects that cross the point.

With some location boundaries, you can specify that objects are selected that either cross (AcMap::kLocationCrossing) or lie inside of the boundary (AcMap::kLocationInside). However, with the AcMapPointBoundary class, you can only specify AcMap::kLocationCrossing.

The AcMapPointBoundary class is derived from the AcMapLocationBoundary class.

Do not subclass from this class.

You create an AcMapPointBoundary pointer using the new operator, so you are responsible for deleting allocated memory using the delete operator.
Header file: MapBoundary.h.

The AcMapPolygonBoundary defines a series of points connected by straight lines that define a polygon. Objects of the AcMapPolygonBoundary class are always fully enclosed, unlike members of the AcMapPolylineBoundary class, which may or may not be closed.

A query can select drawing objects that either cross or lie inside the polygon boundary.

The AcMapPolygonBoundary class is derived from the AcMapFenceBoundary class. To get an AcMapPolygonBoundary object's points, use AcMapFenceBoundary::GetPoints.

Do not subclass from this class.

You get an AcMapPolygonBoundary pointer using the new operator and you are responsible for deleting allocated memory with the delete operator.
AcMapPolylineBoundary class

Functions  Object Model  See Also

Header file: MapBoundary.h.

The AcMapPolylineBoundary defines a series of points connected by straight lines that define a polyline. An AcMapPolylineBoundary object may or may not be fully closed. If it is not closed, a query can select only those drawing objects that cross the polyline. If it is closed, a query can select drawing objects that either cross or lie inside the polyline boundary.

Do not subclass from this class.

You get an AcMapPolylineBoundary pointer using the new operator, and you are responsible for deleting allocated memory using the delete operator.
Header file: MapBoundary.h.

The AcMapWindowBoundary represents a rectangle based on two points that specify its left bottom and right top corners. A query can select drawing objects that either cross or lie inside the window boundary.

The AcMapWindowBoundary class is derived from the AcMapPolygonBoundary class, which is derived from the AcMapFenceBoundary class. To get window boundary points, use AcMapFenceBoundary::GetPoints.

Do not subclass from this class.

You create an AcMapWindowBoundary pointer using the new operator and you are responsible for deleting allocated memory using the delete operator. For simplicity, error checking is not shown.
An instance of the AcMapPropertyAlteration class represents a property alteration. A property alteration has an alteration type, which identifies the property to be altered (color, layer, line type, and so on), and an expression, which describes how the property is to be altered when the results of a query are displayed with property alteration enabled.

Created by calling AcMapPropertyAlterationDefinition::AddAlteration. This function returns a pointer to AcMapPropertyAlteration that you are responsible for deleting with the delete operator.

A collection of property alterations constitutes a property alteration definition, which is represented by an AcMapPropertyAlterationDefinition object. The property alteration definition is contained by the query to which it applies.

There are two other classes for representing property alterations: AcMapTextAlteration and AcMapHatchAlteration. Note that these two are subclassed from AcMapPropertyAlteration, and that both of them use the SetExpression function of the base class to set their respective expressions—to specify the string for the text alteration, or the hatch pattern name for the hatch alteration.

Do not subclass from this class.

The following code alters the color of queried objects to red. This example assumes that a drawing is already attached. For simplicity, error checking is not shown.

```
AcMapSession *mapApi = NULL;
AcMapProject *pProj = NULL;
AcMapDrawingSet *pDSet = NULL;
AcMapQueryBranch *pQBranch = NULL;
AcMapLocationCondition *pLocationCondition = NULL;
AcMapQuery *pQuery = NULL;
mapApi = AcMapGetSession();
mapApi->GetProject(pProj);
pProj->GetDrawingSet(pDSet);
pQBranch = new AcMapQueryBranch();
pLocationCondition = new AcMapLocationCondition(AcMap::kOperatorAnd,
    AcMap::kLocationInside);
pLocationCondition->SetBoundary((&AcMapAllBoundary;()));
pProj->CreateQuery(pQuery, Adesk::kFalse);
pQuery->Clear(Adesk::kTrue);
pQuery->SetMode(AcMap::kQueryDraw);
pQBranch->AppendOperand(pLocationCondition);
AcMapPropertyAlterationDefinition *pDef = NULL;
AcMapPropertyAlteration *pcAlt = NULL;
pQuery->GetPropertyAlteration(pDef);
pDef->AddAlteration(pcAlt, AcMap::kAlterationColor);
pcAlt->SetExpression("RED");
pQuery->EnablePropertyAlteration(Adesk::kTrue);
pQuery->Define(pQBranch);
pQuery->Run();
pDSet->ZoomExtents();
delete pcAlt;
delete pLocationCondition;
```
delete pQBranch;
delete pQuery;
Header file: MapAlteration.h.

**Note** This class has [new functions](#) added for AutoCAD Map 3D 2005.

An instance of the `AcMapPropertyAlterationDefinition` class represents a property alteration definition, which is an ordered collection of property alterations. These property alterations affect the appearance of objects displayed in the results of a query if property alteration for the query is enabled.

There are three classes for representing property alterations: `AcMapPropertyAlteration`, `AcMapTextAlteration`, and `AcMapHatchAlteration`. The latter two are subclassed from the first, and both of them use the parent's `SetExpression` function to specify the string for the text alteration or a hatch pattern name for the hatch alteration.

To create or get the property alteration definition of a query, use the `AcMapQuery::GetPropertyAlteration` function. If property alteration is not yet defined for the query, calling this function creates an empty definition that you can complete by adding property alterations.

Do not subclass from this class. Do not delete the `AcMapPropertyAlterationDefinition` memory.

The following code alters the color of queried objects to red. This example assumes that a drawing is already attached. For simplicity, error checking is not shown.

```cpp
AcMapSession  *mapApi = NULL;
AcMapProject  *pProj = NULL;
AcMapDrawingSet *pDSet = NULL;
AcMapQueryBranch *pQBranch = NULL;
AcMapLocationCondition *pLocationCondition = NULL;
AcMapQuery *pQuery = NULL;

mapApi = AcMapGetSession();
pProj->GetProject(pProj);
pProj->GetDrawingSet(pDSet);
pQBranch = new AcMapQueryBranch();
pLocationCondition = new AcMapLocationCondition(AcMap::kOperatorAnd,
                                             AcMap::kLocationInside);
pLocationCondition->SetBoundary((&AcMapAllBoundary;()));
pProj->CreateQuery(pQuery, Adesk::kFalse);
pQuery->Clear(Adesk::kTrue);
pQuery->SetMode(AcMap::kQueryDraw);
pQBranch->AppendOperand(pLocationCondition);
AcMapPropertyAlterationDefinition *pDef = NULL;
AcMapPropertyAlteration *pcAlt = NULL;
pQuery->GetPropertyAlteration(pDef);
pDef->AddAlteration(pcAlt, AcMap::kAlterationColor);
pAlt->SetExpression("RED");
pQuery->EnablePropertyAlteration(Adesk::kTrue);
pQuery->Define(pQBranch);
pQuery->Run();
pDSet->ZoomExtents();
delete pcAlt ;
delete pLocationCondition;
delete pQBranch;
delete pQuery;
```
AcMapHatchAlteration class

Functions  Object Model  See Also

Header file: MapAlteration.h.

Note This class has new functions added for AutoCAD Map 3D 2005.

An instance of the AcMapHatchAlteration class represents a hatch alteration object, which provides hatch patterns for closed figures when they are queried into the project drawing if property alteration is enabled for the query.

To access members of this class, create an AcMapHatchAlteration pointer using AcMapPropertyAlterationDefinition::AddAlteration and cast it to the AcMapHatchAlteration* type as follows:

```cpp
pDef->AddAlteration(pcAlt, AcMap::kAlterationHatch);

// ...
AcMapTextAlteration *pHatch = NULL;
pHatchAlt = (AcMapHatchAlteration*)pcAlt;
pHatchAlt->SetColor("RED");
```

You are responsible for deleting the pcAlt pointer. Do not attempt to delete the pDef pointer.

The AcMapHatchAlteration class is a subclass of the AcMapPropertyAlteration class. Note that it inherits certain functions of the parent class without overloading them. In particular, note that it uses the parent's SetExpression function to specify the hatch pattern name for the hatch alteration.

Do not subclass from this class.

You create an AcMapHatchAlteration pointer by adding an alteration to the property alteration definition. Next, you specify the hatch pattern using AcMapTextAlteration::SetExpression. Default values are used to alter the pattern unless you access members of AcMapHatchAlteration to you change these values, as shown in the following example, which changes the color of the hatch pattern on the WATER layer. This example assumes you have attached citymap7.dwg from the MAPTUT directory.

```cpp
// For simplicity, error checking is not shown.

AcMapProject *pProj = NULL;
AcMapQueryBranch *pQBranch = NULL;
AcMapLocationCondition *pLocationCondition = NULL;
AcMapQuery *pQuery = NULL;

AcMapSession *mapApi = AcMapGetSession();
mapApi->GetProject(pProj);
pQBranch = new AcMapQueryBranch();
pLocationCondition = new AcMapLocationCondition(AcMap::kOperatorAnd,
                                             AcMap::kLocationInside);
pLocationCondition->SetBoundary((&AcMapAllBoundary;()));
pProj->CreateQuery(pQuery, Adesk::kFalse);
pQuery->CreateQuery(pQuery, Adesk::kFalse);
pQuery->SetMode(AcMap::kQueryDraw);
pQBranch->AppendOperand(pLocationCondition);
AcMapPropertyAlterationDefinition *pDef = NULL;
AcMapPropertyAlteration *pcAlt = NULL;
pQuery->GetPropertyAlteration(pDef);
```
pDef->AddAlteration(pcAlt, AcMap::kAlterationHatch);
AcMapHatchAlteration *pHatchAlt = NULL;
pHatchAlt = (AcMapHatchAlteration*)pcAlt;
pHatchAlt->SetLayer("WATER");
pHatchAlt->SetColor("CYAN");
pcAlt->SetExpression("Solid");
pQuery->EnablePropertyAlteration(Adesk::kTrue);
pQuery->Define(pQBranch);
pQuery->Run();
delete pcAlt;
delete pLocationCondition;
delete pQBranch;
delete pQuery;
AcMapTextAlteration class

Functions  Object Model  See Also

Header file: MapAlteration.h.

**Note** This class has [new functions](#) added for AutoCAD Map 3D 2005.

An instance of the AcMapTextAlteration class represents a text alteration, which provides text labels for objects queried into the project drawing if property alteration is enabled for the query.

The AcMapTextAlteration class is a subclass of the AcMapPropertyAlteration class. Do not subclass from this class. Note that it inherits certain functions of the base class without overloading them. AcMapTextAlteration inherits AcMapTextAlteration::GetType from AcMapPropertyAlteration and it returns kAlterationTextEntity.

To access the members of this class, create an AcMapTextAlteration pointer using AcMapPropertyAlterationDefinition::AddAlteration and cast it to the AcMapTextAlteration* type as follows:

```cpp
pDef->AddAlteration(pcAlt, AcMap::kAlterationTextEntity);
...
AcMapTextAlteration *pTextAlt = NULL;
pTextAlt = (AcMapTextAlteration*)pcAlt;
pTextAlt->SetColor("RED");
```

You are responsible for deleting the pcAlt pointer. Do not attempt to delete the pDef pointer.

You create an AcMapTextAlteration pointer by adding an alteration to the property alteration definition. Next, you specify the text to be altered using AcMapTextAlteration::SetExpression. Default values are used to alter the specified text unless you access members of AcMapTextAlteration, as shown in the following example that changes the color of text. This example assumes you have attached citymap7.dwg from the MAPTUT directory.

```cpp
// For simplicity, error checking is not shown.
AcMapSession *mapApi = NULL;
AcMapProject *pProj = NULL;
AcMapQueryBranch *pQBranch = NULL;
AcMapLocationCondition *pLocationCondition = NULL;
AcMapQuery *pQuery = NULL;

mapApi = AcMapGetSession();
mapApi->GetProject(pProj);
pQBranch = new AcMapQueryBranch();
pLocationCondition = new AcMapLocationCondition(AcMap::kOperatorAnd,
                                                AcMap::kLocationInside);
pLocationCondition->SetBoundary((&AcMapAllBoundary;()));
pProj->CreateQuery(pQuery, Adesk::kFalse);
pQuery->Clear(Adesk::kTrue);
pQuery->SetMode(AcMap::kQueryDraw);
pQBranch->AppendOperand(pLocationCondition);
AcMapPropertyAlterationDefinition *pDef = NULL;
AcMapPropertyAlteration *pcAlt = NULL;
pQuery->GetPropertyAlteration(pDef);
```
pDef->AddAlteration(pcAlt, AcMap::kAlterationTextEntity);
    AcMapTextAlteration *pTextAlt = NULL;
    pTextAlt = (AcMapTextAlteration*)pcAlt;
    pTextAlt->SetColor("RED");
    pcAlt->SetExpression("@NAME@WATER_BODIES");
    pQuery->EnablePropertyAlteration(Adesk::kTrue);
    pQuery->Define(pQBranch);
    pQuery->Run();
    delete pcAlt;
    delete pLocationCondition;
    delete pQBranch;
    delete pQuery;
AcMapRangeLibrary class

The AcMapRangeLibrary class represents a range library, which contains all the range tables available to a project.

A range library is instantiated automatically when a project is instantiated. There is only one range library per project. You get a pointer to the project's range library with the AcMapProject::GetRangeLibrary function. Do not delete the memory associated with the AcMapRangeLibrary pointer.

Do not subclass from this class.

The following example shows how to use this class. It assumes that you have already attached citymap7.dwg from the MAPTUT directory. It queries all objects in the drawing, and labels each polyline and line. For simplicity, error checking is not shown.

```c
AcMapSession *mapApi = NULL;
AcMapProject *pProj = NULL;
AcMapDrawingSet *pDSet = NULL;
AcMapQueryBranch *pQBranch = NULL;
AcMapLocationCondition *pLocationCondition = NULL;
AcMapQuery *pQuery = NULL;
AcMapPropertyAlterationDefinition *pDef = NULL;
AcMapPropertyAlteration *pcAlt = NULL;
AcMapRangeLibrary *pRangeLib = NULL;
AcMapRangeTable *pTable = NULL;

mapApi = AcMapGetSession();
mapApi->GetProject(pProj);
pProj->GetDrawingSet(pDSet);
pQBranch = new AcMapQueryBranch();
pLocationCondition = new AcMapLocationCondition(AcMap::kOperatorAnd,
                                              AcMap::kLocationInside);
pLocationCondition->SetBoundary((&AcMapAllBoundary;()));
pProj->CreateQuery(pQuery, Adesk::kFalse);
pQuery->Clear(Adesk::kTrue);
pQuery->SetMode(AcMap::kQueryDraw);
pQBranch->AppendOperand(pLocationCondition);
pQuery->GetPropertyAlteration(pDef);
pDef->AddAlteration(pcAlt, AcMap::kAlterationTextEntity);
pProj->GetRangeLibrary(pRangeLib);
pRangeLib->Clear();
const char *pcName = "MyTypeRangeTable";
const char *pcDsc = "Table for types";
pRangeLib->AddRangeTable(pTable, pcName, pcDsc);
pTable->AddRangeLine(AcMap::kRangeEq, "polyline", "PLINE");
pTable->AddRangeLine(AcMap::kRangeEq, "line", "LINE");
pTable->AddRangeLine(AcMap::kRangeEq, "lwpolyline", "LWPOLYLINE");
pcAlt->SetExpression("(Range .TYPE MyTypeRangeTable)");
pQuery->EnablePropertyAlteration(Adesk::kTrue);
pQuery->Define(pQBranch);
pQuery->Run();
pDSet->ZoomExtents();
```
delete pTable;
delete pcAlt;
delete pLocationCondition;
delete pQBranch;
delete pQuery;
The AcMapRangeLine class represents a range line, also called a range expression. A range line is a line in a range table consisting of a condition to compare to an object property or object data value and the value to return if the condition evaluates to True.

The condition portion of the range line consists of an operator and a simple expression value. The operator (AcMap::ERangeOperator) can be a comparative operator or a special otherwise operator (AcMap::kRangeOtherwise) which covers conditions not specified by the other range lines in the range table. The expression value is called the range definition.

For example, in the range line

```
> 1000 SMALL
```

> is the range operator, 1000 is the range definition, and SMALL is the return value. In the UI, the user would see this range line displayed as

```
If > 100 Return: SMALL
```

Create a range line using the AcMapRangeTable::AddRangeLine function. Get a pointer to AcMapRangeLine using the AcMapRangeTable::GetRangeLine function. You are responsible for deleting the memory allocated for the range line using the delete operator.

To associate range lines in a range table with a property alteration, pass an expression that evaluates to the range table to AcMapPropertyAlteration::SetExpression. For example:

```c++
//Add a range table to the range library
AcMapRangeTable *pTable = NULL;
const char *pcName = "MyTypeRangeTable";
const char *pcDsc = "Table for types";

if(pRangeLib->AddRangeTable(pTable, pcName, pcDsc) == AcMap::kOk)
{
    // Add range lines that will add text next to the entities
    // that match the input criteria.
    pTable->AddRangeLine(AcMap::kRangeEq, "10e", "LOW");
    pTable->AddRangeLine(AcMap::kRangeEq, "20", "MEDIUM");
    pTable->AddRangeLine(AcMap::kRangeEq, "30", "HIGH");
    pTable->AddRangeLine(AcMap::kRangeOtherwise, NULL,
    "Out of range.");
}

//Add a Text Alteration
AcMapPropertyAlteration *pPropAltObj = NULL;
if(pPADef->AddAlteration(pPropAltObj, AcMap::kAlterationTextEntity) == AcMap::kOk)
{
    AcMapTextAlteration *pTextAlt = NULL;

    //cast the property alteration to a text alteration
    pTextAlt = (AcMapTextAlteration*)pPropAltObj;

    //set some attributes
    pTextAlt->SetColor("Magenta");
```

Do not subclass from this class.

For more information, click . Also click Using AutoCAD Map > Queries > Altering the Properties of Queried Objects > Creating a Range Table on the Contents tab of AutoCAD Map Help.
Header file: MapArxApi.h.

The AcMapRangeTable class represents a range table, which contains a collection of range lines (also called range expressions).

Get an AcMapRangeTable pointer using the AcMapRangeLibrary::AddRangeTable function, and delete the memory allocated by this function using the delete operator.

Do not subclass from this class.

The following example shows how to use this class. It assumes that you have already attached citymap7.dwg from the MAPTUT directory. It queries all objects in the drawing, and labels each polyline and line. For simplicity, error checking is not shown.

```cpp
AcMapSession *mapApi = NULL;
AcMapProject *pProj = NULL;
AcMapDrawingSet *pDSet = NULL;
AcMapQueryBranch *pQBranch = NULL;
AcMapLocationCondition *pLocationCondition = NULL;
AcMapQuery *pQuery = NULL;
AcMapPropertyAlterationDefinition *pDef = NULL;
AcMapPropertyAlteration *pcAlt = NULL;
AcMapRangeLibrary *pRangeLib = NULL;
AcMapRangeTable *pTable = NULL;

mapApi = AcMapGetSession();
mapApi->GetProject(pProj);
pProj->GetDrawingSet(pDSet);
pQBranch = new AcMapQueryBranch();
pLocationCondition = new AcMapLocationCondition(AcMap::kOperatorAnd,
    AcMap::kLocationInside);
pLocationCondition->SetBoundary((&AcMapAllBoundary;()));
pProj->CreateQuery(pQuery, Adesk::kFalse);
pQuery->Clear(Adesk::kTrue);
pQuery->SetMode(AcMap::kQueryDraw);
pQBranch->AppendOperand(pLocationCondition);
pQuery->GetPropertyAlteration(pDef);
pDef->AddAlteration(pcAlt, AcMap::kAlterationTextEntity);
pProj->GetRangeLibrary(pRangeLib);
pRangeLib->Clear();
const char *pcName = "MyTypeRangeTable";
const char *pcDsc = "Table for types";
pRangeLib->AddRangeTable(pTable, pcName, pcDsc);
pTable->AddRangeLine(AcMap::kRangeEq, "polyline", "PLINE");
pTable->AddRangeLine(AcMap::kRangeEq, "line", "LINE");
pTable->AddRangeLine(AcMap::kRangeEq, "lwpolyline", "LWPOLYLINE");
pcAlt->SetExpression("(Range .TYPE MyTypeRangeTable)");
pQuery->EnablePropertyAlteration(Adesk::kTrue);
pQuery->Define(pQBranch);
pQuery->Run();
pDSet->ZoomExtents();
delete pTable;
```
delete pcAlt;
delete pLocationCondition;
delete pQBranch;
delete pQuery;
Header file: MapArxApi.h.

An instance of the AcMapErrorEntry class represents an error. Error entries are pushed onto an error stack where you can retrieve them.

You do not create error entries explicitly. AutoCAD Map automatically generates an error entry whenever it encounters an error. You call AcMapErrorStack::GetEntry to get an error entry object pointer.

Do not subclass from this class.
Header file: MapArxApi.h.

An instance of the AcMapErrorParameter class stores information about an error entry parameter. You get an AcMapErrorParameter pointer by calling AcMapErrorEntry::GetParameter, and you must delete the memory allocated for the object using the delete operator.

Do not subclass from this class.
Header file: MapArxApi.h.

An instance of the AcMapErrorStack class represents the error stack for an AutoCAD Map session. The error stack is a last-in, first-out data structure, where the most recent error is pushed onto the top of the stack. The AcMapErrorStack::GetEntry function pops the most recent error from the top of the stack.

AutoCAD Map clears the error stack after completing a command. AutoCAD Map automatically generates an error entry whenever it encounters an error and pushes the error entry onto the error stack. An application can also push error entries onto the error stack.

Do not subclass from this class.
This template class is defined in MapColl.h.

```
template <class ApiObj> class AcMapObjArray
```

The AcMapObjArray class is used for dynamic object arrays. It stores ApiObj objects in the member data. To use this template for collecting objects of a particular class, you must define the operator= function for the class.

The AutoCAD Map ARX API uses AcMapObjArray and AcMapObjPtrArray instead of MFC collections to provide array services.

A dynamic object array can grow as needed at runtime.
AcMapObjPtrArray class

Header file: MapColl.h.

template <class ApiObj> class AcMapObjPtrArray

The AcMapObjPtrArray class is used for dynamic object pointer arrays. It stores pointers to ApiObj objects in the member data. The application is responsible for allocating and freeing these objects.

To use this template for collecting objects of a particular class, you must define the operator= function for the class.

The AutoCAD Map ARX API uses AcMapObjPtrArray and AcMapObjArray instead of MFC collections to provide array services.

A dynamic object array can grow as needed at runtime.
Header file: MapStringArray.h.
The AcMapStringArray class is a dynamic array used to represent character strings. A dynamic string array can grow as needed at runtime.
The AutoCAD Map ARX API uses AcMapStringArray instead of MFC CstringList for dynamic character strings.
Gets the AutoCAD Map application object.
Include MapSession.h.

```
AcMapSession* AcMapGetSession();
```

Returns the application object.

Before you can do anything with AutoCAD Map, you must get the application object.

```
AcMapSession* pSession = AcMapGetSession();
```
Attached drawing status codes.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>kDwgInactive</td>
</tr>
<tr>
<td>1</td>
<td>kDwgActive</td>
</tr>
<tr>
<td>4</td>
<td>kDwgLocked</td>
</tr>
</tbody>
</table>
## AcMap::EAdeDwgUpdateStatus

### Drawing update status.

<table>
<thead>
<tr>
<th>Status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>kDwgNonUpdated</td>
</tr>
<tr>
<td>1</td>
<td>kDwgShouldBeReloaded</td>
</tr>
<tr>
<td>4</td>
<td>kDwgShouldBeRequeried</td>
</tr>
</tbody>
</table>
### Property alteration types.

<table>
<thead>
<tr>
<th>Value</th>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>kAlterationBlockName</td>
<td>Block name</td>
</tr>
<tr>
<td>01</td>
<td>kAlterationColor</td>
<td>Color</td>
</tr>
<tr>
<td>02</td>
<td>kAlterationLayer</td>
<td>Layer name</td>
</tr>
<tr>
<td>03</td>
<td>kAlterationRotation</td>
<td>Rotation</td>
</tr>
<tr>
<td>04</td>
<td>kAlterationElevation</td>
<td>Z coordinate in the user coordinate system</td>
</tr>
<tr>
<td>05</td>
<td>kAlterationHeight</td>
<td>Text height</td>
</tr>
<tr>
<td>06</td>
<td>kAlterationLineType</td>
<td>Line type</td>
</tr>
<tr>
<td>07</td>
<td>kAlterationScale</td>
<td>Scaling factor. For example &quot;1.2&quot; = 120%</td>
</tr>
<tr>
<td>08</td>
<td>kAlterationStyle</td>
<td>Text style</td>
</tr>
<tr>
<td>09</td>
<td>kAlterationText</td>
<td>Text value</td>
</tr>
<tr>
<td>10</td>
<td>kAlterationThickness</td>
<td>Thickness</td>
</tr>
<tr>
<td>11</td>
<td>kAlterationWidth</td>
<td>Line width</td>
</tr>
<tr>
<td>12</td>
<td>kAlterationTextEntity</td>
<td>Text entity definition.</td>
</tr>
<tr>
<td>13</td>
<td>kAlterationHatch</td>
<td>Hatch definition.</td>
</tr>
</tbody>
</table>
Run-time class identification for query condition classes

**Location boundaries**

<table>
<thead>
<tr>
<th>Location Boundary</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>01 kLocationBoundary</td>
<td></td>
</tr>
<tr>
<td>02 kAllBoundary</td>
<td></td>
</tr>
<tr>
<td>03 kPointBoundary</td>
<td></td>
</tr>
<tr>
<td>04 kCircleBoundary</td>
<td></td>
</tr>
<tr>
<td>05 kFenceBoundary</td>
<td></td>
</tr>
<tr>
<td>06 kBufferFenceBoundary</td>
<td></td>
</tr>
<tr>
<td>07 kPolygonBoundary</td>
<td></td>
</tr>
<tr>
<td>08 kWindowBoundary</td>
<td></td>
</tr>
<tr>
<td>09 kPolylineBoundary</td>
<td></td>
</tr>
<tr>
<td>10 kBufferPolylineBoundary</td>
<td></td>
</tr>
<tr>
<td>11 kClosedPolylineBoundary</td>
<td></td>
</tr>
</tbody>
</table>

**Query units**

<table>
<thead>
<tr>
<th>Query Unit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>12 kQueryUnit</td>
<td>Query unit of undetermined type.</td>
</tr>
<tr>
<td>13 kQueryCondition</td>
<td>Query condition of undetermined type.</td>
</tr>
<tr>
<td>14 kLocationCondition</td>
<td>Location condition.</td>
</tr>
<tr>
<td>15 kPropertyCondition</td>
<td>Property condition.</td>
</tr>
<tr>
<td>16 kSQLCondition</td>
<td>SQL condition.</td>
</tr>
<tr>
<td>17 kDataCondition</td>
<td>Data condition.</td>
</tr>
<tr>
<td>18 kLocationConditionImp</td>
<td>Not intended for public use.</td>
</tr>
<tr>
<td>19 kPropertyConditionImp</td>
<td>Not intended for public use.</td>
</tr>
<tr>
<td></td>
<td>Class Name</td>
</tr>
<tr>
<td>---</td>
<td>-------------------------</td>
</tr>
<tr>
<td>20</td>
<td>kSQLConditionImp</td>
</tr>
<tr>
<td>21</td>
<td>kDataConditionImp</td>
</tr>
<tr>
<td>22</td>
<td>kQueryBranch</td>
</tr>
</tbody>
</table>
Comparison operators for query conditions.

**Note** The only valid operator in a string context is AcMap::kCondEq.

<table>
<thead>
<tr>
<th></th>
<th>Operator</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>kCondEq</td>
<td>Equal.</td>
</tr>
<tr>
<td>2</td>
<td>kCondGT</td>
<td>Greater than.</td>
</tr>
<tr>
<td>3</td>
<td>kCondGTorEq</td>
<td>Greater than or equal.</td>
</tr>
<tr>
<td>4</td>
<td>kCondLT</td>
<td>Less than.</td>
</tr>
<tr>
<td>5</td>
<td>kCondLTorEq</td>
<td>Less than or equal.</td>
</tr>
<tr>
<td>6</td>
<td>kCondNotEq</td>
<td>Not equal.</td>
</tr>
</tbody>
</table>
Data query types.

<table>
<thead>
<tr>
<th></th>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>kDataIRD</td>
<td>Internal resources dictionary data tables, object data fields, and an expression</td>
</tr>
<tr>
<td>1</td>
<td>kDataAttribute</td>
<td>Blocks and attribute tags</td>
</tr>
<tr>
<td>2</td>
<td>kDataLinkTemplate</td>
<td>Link templates and key columns for tables</td>
</tr>
<tr>
<td>3</td>
<td>kDataEED</td>
<td>Extended entity data</td>
</tr>
<tr>
<td>4</td>
<td>kDataFeature</td>
<td>Feature data</td>
</tr>
</tbody>
</table>
Data types for object data fields.

<table>
<thead>
<tr>
<th></th>
<th>AcMap::EDataType</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>kUnknownType</td>
</tr>
<tr>
<td>1</td>
<td>kInteger</td>
</tr>
<tr>
<td>2</td>
<td>kReal</td>
</tr>
<tr>
<td>3</td>
<td>kCharacter</td>
</tr>
<tr>
<td>4</td>
<td>kPoint</td>
</tr>
</tbody>
</table>
AcMap::EErrCode enumerators represent error codes.

Errors on the error stack are represented by AcMapErrorEntry objects. Error codes are returned by the AcMapErrorEntry::ErrorCode function.

<table>
<thead>
<tr>
<th>Expression</th>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>kErrXEDValueFail</td>
<td>–15</td>
<td>Internal error.</td>
</tr>
<tr>
<td>kErrRngTabEvalFail</td>
<td>–14</td>
<td>Internal error.</td>
</tr>
<tr>
<td>kErrNoRngTabFound</td>
<td>–13</td>
<td>Internal error.</td>
</tr>
<tr>
<td>kErrNoRngTabLibFound</td>
<td>–12</td>
<td>Internal error.</td>
</tr>
<tr>
<td>kErrLispValueFail</td>
<td>–11</td>
<td>Internal error.</td>
</tr>
<tr>
<td>kErrGetAttrFail</td>
<td>–10</td>
<td>Internal error.</td>
</tr>
<tr>
<td>kErrGetSQLFail</td>
<td>–09</td>
<td>Internal error.</td>
</tr>
<tr>
<td>kErrGetEEDFail</td>
<td>–08</td>
<td>Internal error.</td>
</tr>
<tr>
<td>kErrInvalidProperty</td>
<td>–07</td>
<td>Internal error.</td>
</tr>
<tr>
<td>kErrExpInvalidOperand</td>
<td>–06</td>
<td>Internal error.</td>
</tr>
<tr>
<td>kErrExpADS</td>
<td>–05</td>
<td>Internal error.</td>
</tr>
<tr>
<td>kErrExpNoMemforOperand</td>
<td>–04</td>
<td>Internal error.</td>
</tr>
<tr>
<td>kErrExpMathOverFlow</td>
<td>–03</td>
<td>Internal error.</td>
</tr>
<tr>
<td>kErrExpInvalidOper</td>
<td>–02</td>
<td>Internal error.</td>
</tr>
<tr>
<td>kErrExpSyntaxErr</td>
<td>–01</td>
<td>Internal error.</td>
</tr>
</tbody>
</table>
For more Expression errors, which begin at 2900, click □.

**Common Usage**

<table>
<thead>
<tr>
<th>Code</th>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>kOk</td>
<td>General ADE return value: call to ADE object is not successful.</td>
</tr>
<tr>
<td>01</td>
<td>kErr</td>
<td>Invalid function argument.</td>
</tr>
<tr>
<td>02</td>
<td>kErrBadInput</td>
<td></td>
</tr>
<tr>
<td>03</td>
<td>kErrObjectNotFound</td>
<td></td>
</tr>
<tr>
<td>04</td>
<td>kErrOutOfMemory</td>
<td></td>
</tr>
<tr>
<td>05</td>
<td>kErrObjNotInitialized</td>
<td></td>
</tr>
<tr>
<td>06</td>
<td>kErrWrongType</td>
<td></td>
</tr>
<tr>
<td>07</td>
<td>kErrWrongProject</td>
<td></td>
</tr>
<tr>
<td>08</td>
<td>kErrEOB</td>
<td>Invalid function argument type.</td>
</tr>
<tr>
<td>09</td>
<td>kErrADSError</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>kErrAdsNameConversionFails</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>kErrWrongArgument</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>kErrWriteBufFails</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>kErrReadBufFails</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>kErrXDataCorrupted</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>kErrNoEnvironment</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>kErrUsrBreak</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>kErrUncomparable</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>kErrPermissionDenied</td>
<td>No permission to perform some action. When using the following commands, the message has different meanings: ADESAVEOBS = can't be executed in demonstration mode; ADEDRAWINGS = User has no rights to update the set; ADESELOBJS, ADDREMOBJS = User has no rights to edit; ADEDWGMAINT = User needs to be superuser to remove the foreign locks; MAPOPTIONS: System Preferences = can't set &quot;Force User Login&quot; flag</td>
</tr>
</tbody>
</table>
and change "Object Locking" flag if drawing set contains active or locked drawings. Deactivate or unlock the drawings before trying to set these preferences.

<table>
<thead>
<tr>
<th>Code</th>
<th>Error Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>19</td>
<td>kErrWrongSyntax</td>
</tr>
<tr>
<td>20</td>
<td>kErrDuplicate</td>
</tr>
<tr>
<td>21</td>
<td>kErrInvalidPathOrFileName</td>
</tr>
<tr>
<td>22</td>
<td>kErrInvalidVersion</td>
</tr>
<tr>
<td>23</td>
<td>kErrFileIOFatalError</td>
</tr>
</tbody>
</table>

**External Subsystems**

<table>
<thead>
<tr>
<th>Code</th>
<th>Error Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>kErrAcDbError</td>
</tr>
<tr>
<td>201</td>
<td>kErrIRDNotInitialized</td>
</tr>
<tr>
<td>202</td>
<td>kErrIRDError</td>
</tr>
<tr>
<td>203</td>
<td>kErrASENotInitialized</td>
</tr>
<tr>
<td>204</td>
<td>kErrASEError</td>
</tr>
<tr>
<td>205</td>
<td>kErrASIError</td>
</tr>
<tr>
<td>206</td>
<td>kErrAsiNotInitialized</td>
</tr>
<tr>
<td>207</td>
<td>kErrAsiConnectToEnvironmentFailed</td>
</tr>
</tbody>
</table>

**Transactional Manager**

<table>
<thead>
<tr>
<th>Code</th>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1000</td>
<td>kErrClosed</td>
<td>Repeated attempts to close previously closed ADE object. Call support.</td>
</tr>
<tr>
<td>1001</td>
<td>kErrWasErased</td>
<td>Attempt to work with erased ADE object. Call support.</td>
</tr>
<tr>
<td>1002</td>
<td>kErrOpenForRead</td>
<td>Attempt to update ADE objects opened for read. Call support.</td>
</tr>
<tr>
<td>1003</td>
<td>kErrOpenForWrite</td>
<td>Attempts to get multiple access to ADE object opened for write. Call support.</td>
</tr>
<tr>
<td>1004</td>
<td>kErrWrongMode</td>
<td>Incorrect mode of the ADE objects should be opened. Call support.</td>
</tr>
<tr>
<td>1005</td>
<td>kErrClone</td>
<td>Exception at the time of cloning ADE objects. Call support.</td>
</tr>
<tr>
<td>Code</td>
<td>Error Code</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>---------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>1006</td>
<td>kErrResponse</td>
<td>Incorrect attempt to start ADE transaction. Call support.</td>
</tr>
<tr>
<td>1007</td>
<td>kErrObjIsAbsent</td>
<td>Attempt to work with erased ADE object. Call support.</td>
</tr>
<tr>
<td>1008</td>
<td>kErrAccess</td>
<td>Type of work with ADE does not correspond to its status. Call support.</td>
</tr>
<tr>
<td>1009</td>
<td>kErrMultipleUsage</td>
<td>Attempts to get multiple access to ADE object opened for write. Call support.</td>
</tr>
<tr>
<td>1010</td>
<td>kErrUpgrade</td>
<td>Incorrect attempt to update opening mode of the ADE object. Call support.</td>
</tr>
<tr>
<td>1011</td>
<td>kErrNotClosed</td>
<td>Object was not closed during current ADE transaction. Call support.</td>
</tr>
</tbody>
</table>

**CAdeList**

<table>
<thead>
<tr>
<th>Code</th>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1100</td>
<td>kErrGetIteratorFails</td>
<td></td>
</tr>
<tr>
<td>1101</td>
<td>kErrListIsEmpty</td>
<td></td>
</tr>
</tbody>
</table>

**CAdeListIterator**

<table>
<thead>
<tr>
<th>Code</th>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1102</td>
<td>kErrListEnd</td>
<td></td>
</tr>
<tr>
<td>1103</td>
<td>kErrListObjectIsAbsent</td>
<td></td>
</tr>
</tbody>
</table>

**Drawing**

<table>
<thead>
<tr>
<th>Code</th>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1200</td>
<td>kErrQueriedEntity</td>
<td>Ignoring queried entity for saving selection. When using ADESSAVEOBSJS, the message means queried objects that are selected for save back are ignored.</td>
</tr>
<tr>
<td>1201</td>
<td>kErrDwgNotActive</td>
<td>Attempt to read the object locked from an inactive drawing. When using ADEWHOHASIT, the message means the drawing from which the object was queried is no longer active. ADE is unable to determine if the object selected is currently locked. Activate the specified drawing and reenter the command.</td>
</tr>
<tr>
<td>1202</td>
<td>kErrReadDwgFileFails</td>
<td>ADE fails to read external drawing. If using ADEDRAWINGS, this message means that ADE is unable to read the specified drawing. Perhaps the</td>
</tr>
<tr>
<td>Code</td>
<td>Description</td>
<td>Message Description</td>
</tr>
<tr>
<td>-------</td>
<td>-----------------------------------------------------------------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>1203</td>
<td>kErrMultipleUsers</td>
<td>ADE fails to access external drawing.. If using ADEDRAWINGS, this message means ADE is unable to access the specified drawing because of file locking problems. Check the error message stack for more information.</td>
</tr>
<tr>
<td>1204</td>
<td>kErrEntityLockingFails</td>
<td>ADE fails to lock. If using ADESELOBJS, this message means ADE is unable to lock an object. The object may already be locked. Use the ADEWHOHASIT to determine if the object is locked, perhaps by another user. Check the error message stack for more information.</td>
</tr>
<tr>
<td>1205</td>
<td>kErrUnlockedEntity</td>
<td>Entity is unlocked by another ADE user. If using ADEWHOHASIT, this message indicates that the selected object is not currently locked.</td>
</tr>
<tr>
<td>1206</td>
<td>kErrLockedEntity</td>
<td>Entity has been locked by another ADE user. If using ADESELOBJS, this message indicates that the object is already locked by another user. Use the ADEWHOHASIT command to identify the user.</td>
</tr>
<tr>
<td>1207</td>
<td>kErrAlreadyInSaveSet</td>
<td></td>
</tr>
<tr>
<td>1208</td>
<td>kErrAlreadyNotInSaveSet</td>
<td></td>
</tr>
<tr>
<td>1209</td>
<td>kErrMultipleReaders</td>
<td></td>
</tr>
<tr>
<td>1210</td>
<td>kErrOpenPrefDictionaryFails</td>
<td>ADE fails to open ADE preferences dictionary. Call support.</td>
</tr>
<tr>
<td>1211</td>
<td>kErrSavePrefDictionaryFails</td>
<td>ADE fails to save ADE preferences dictionary. Call support.</td>
</tr>
<tr>
<td>1212</td>
<td>kErrRestoreDSetFails</td>
<td>ADE fails to restore the drawing set. Call support.</td>
</tr>
<tr>
<td>1213</td>
<td>kErrOpenDSetDictionaryFails</td>
<td>ADE fails to open ADE drawing set dictionary. Call support.</td>
</tr>
<tr>
<td>1214</td>
<td>kErrSaveDSetDictionaryFails</td>
<td>ADE fails to save ADE drawing set in dictionary. Call support.</td>
</tr>
<tr>
<td>1215</td>
<td>kErrOpenQueryDictionaryFails</td>
<td>ADE fails to open ADE query library dictionary. Call support.</td>
</tr>
<tr>
<td>Code</td>
<td>Error Code</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>----------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>1216</td>
<td>kErrSaveQueryDictionaryFails</td>
<td>ADE fails to save ADE query library in dictionary. Call support.</td>
</tr>
<tr>
<td>1217</td>
<td>kErrOpenRTableDictionaryFails</td>
<td>ADE fails to open ADE range table dictionary. Call support.</td>
</tr>
<tr>
<td>1218</td>
<td>kErrSaveRTableDictionaryFails</td>
<td>ADE fails to save ADE range table in dictionary. Call support.</td>
</tr>
<tr>
<td>1219</td>
<td>kErrRestoreRTableDictionaryFails</td>
<td>ADE fails to restore ADE range table in dictionary. Call support.</td>
</tr>
<tr>
<td>1220</td>
<td>kErrOpenDocViewDictionaryFails</td>
<td>ADE fails to open ADE Doc View information dictionary. Call support.</td>
</tr>
<tr>
<td>1221</td>
<td>kErrSaveDocViewDictionaryFails</td>
<td>ADE fails to save ADE Doc View information in dictionary. Call support.</td>
</tr>
<tr>
<td>1222</td>
<td>kErrOpenKeyViewDictionaryFails</td>
<td>ADE fails to open ADE Key View information dictionary.</td>
</tr>
<tr>
<td>1223</td>
<td>kErrSaveKeyViewDictionaryFails</td>
<td>ADE fails to save ADE Key View information in dictionary. Call support.</td>
</tr>
<tr>
<td>1224</td>
<td>kErrSaveProjectionFails</td>
<td>ADE fails to save projection code in the drawing. Call support.</td>
</tr>
<tr>
<td>1225</td>
<td>kErrCopyHardPointerFails</td>
<td>ADE fails to apply property alteration for some symbol table. Call support.</td>
</tr>
<tr>
<td>1226</td>
<td>kErrDwgToBeReloaded</td>
<td></td>
</tr>
<tr>
<td>1227</td>
<td>kErrDwgHasBeenModified</td>
<td>There were objects queried from the drawing that will be treated as new objects. If using ADEDRAWINGS, this message means that when a drawing from which objects have been queried is detached, ADE converts the objects into newly created objects. When you use the ADEWHOHASIT command to see the origin of these objects, it says that they have not been queried.</td>
</tr>
<tr>
<td>1228</td>
<td>kErrOnLockedLayer</td>
<td>Objects from a locked layer have been selected. If using ADESELOBJS, ADEREMOBJS, or ADESAVEOBJS, this message means you selected objects from a locked layer and they cannot be added to, saved to, or removed from the the save set or saved to source.</td>
</tr>
<tr>
<td>1229</td>
<td>kErrDwgSaveFales</td>
<td>ADE fails to save the source drawing. If using</td>
</tr>
<tr>
<td>Error Code</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>------------</td>
<td>-------------</td>
<td></td>
</tr>
<tr>
<td>1230</td>
<td>kErrDwgLocksLeft</td>
<td>ADE fails to remove object locks (if present) by the end of ADE session. If using OPEN, NEW, or QUIT, or if you're configuring options, or modifying objects, you may get this message. Call support. NOTE: Use this message exactly as spelled here.</td>
</tr>
<tr>
<td>1231</td>
<td>kErrLinkWillBeLost</td>
<td>ADE does not save links between queried objects and source drawings between ADE sessions. Detach source drawing with queried objects; Use the SAVE command with queried objects or no objects in the save set.</td>
</tr>
<tr>
<td>1232</td>
<td>kErrDwgDiskFull</td>
<td>ADE custom object is selected to add to save set or to save. If using ADESAVEOBJS, ADESELOBJS, or ADEDWGMAINT, this message means entity modification occurred. Call support.</td>
</tr>
</tbody>
</table>

**Current Session**

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1300</td>
<td>kErrGetCPointFails</td>
</tr>
<tr>
<td>1301</td>
<td>kErrSetCPointFails</td>
</tr>
<tr>
<td>1302</td>
<td>kErrTextInsPointMissed</td>
</tr>
<tr>
<td>1303</td>
<td>kErrTextAllignPointMissed</td>
</tr>
<tr>
<td>1304</td>
<td>kErrTextHeightMissed</td>
</tr>
<tr>
<td>1305</td>
<td>kErrTextStringMissed</td>
</tr>
<tr>
<td>1306</td>
<td>kErrIndexUpdateFails</td>
</tr>
<tr>
<td>1307</td>
<td>kErrIgnorePreview</td>
</tr>
</tbody>
</table>
ADEREMOBS this message has the following meaning: When doing a Preview Query, ADE creates a special object called a PREVIEW object, used to display the queried objects. PREVIEW cannot be saved back to a source drawing. When a user selects this object for adding to the save set or when saving back, ADE detects this and prevents the operation.

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Error Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1308</td>
<td>kErrIgnorePSpace</td>
</tr>
<tr>
<td>1309</td>
<td>kErrEraseIRDOObjectFails</td>
</tr>
<tr>
<td>1310</td>
<td>kErrRenameIRDOObjectFails</td>
</tr>
<tr>
<td>1311</td>
<td>kErrAlterIRDOObjectFails</td>
</tr>
<tr>
<td>1312</td>
<td>kErrIrdDuplicateTableName</td>
</tr>
<tr>
<td>1313</td>
<td>kErrIrdNotIdenticaFormat</td>
</tr>
<tr>
<td>1314</td>
<td>kErrBHatchUnit</td>
</tr>
</tbody>
</table>
| 1315       | kErrReQuery | ADE queried one or more objects twice. ADE does not support UNDO for this operation. If using ADEQUERY or ADERUNXQUERY this message has the following meaning: If a drawing file has been
modified by another ADE user and if a queried object matches another query, ADE removes the old copy and queries a new copy. This operation can't be undone.

<table>
<thead>
<tr>
<th>Code</th>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1316</td>
<td>kErrCantAccessFont</td>
<td></td>
</tr>
<tr>
<td>1317</td>
<td>kErrCantAccessImageFile</td>
<td></td>
</tr>
<tr>
<td>1318</td>
<td>kErrUnableRedefineXrefBlock</td>
<td></td>
</tr>
</tbody>
</table>

**Drawing Set**

<table>
<thead>
<tr>
<th>Code</th>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1500</td>
<td>kErrRestoreDrawingSetFails</td>
<td>ADE fails to read Drawing Set from DWG file. You encounter this message during ADE initialization or when using Open drawing file or ADEDRAWINGS. If using ADEDRAWINGS - attach drawing, the message means &quot;activate on attach&quot; is ON or ade_dsattach(), ade_dsreadattach() drawing is corrupted or old ADE version is in use.</td>
</tr>
<tr>
<td>1501</td>
<td>kErrDuplicatedDrawing</td>
<td>Attempt to attach the same drawing twice. If using ADEDRAWINGS, this error appears when a drawing with the same name has already been attached to the work session.</td>
</tr>
<tr>
<td>1502</td>
<td>kErrNestedDrawing</td>
<td>Prohibited attempt to edit properties of the drawing in the nested drawing set. If using ADEDRAWINGS, this message means a user is not allowed to modify the transformation and save back extents of nested drawings in the work session. These properties can only be modified for top level drawings.</td>
</tr>
<tr>
<td>1503</td>
<td>kErrActivateDrawingFails</td>
<td>ADE fails to activate source drawing. ADE was unable to activate a drawing Perhaps the drawing does not exist or it is locked by another user, or the current user doesn't have permission to read the specified drawing file. Check the error message stack for more information.</td>
</tr>
<tr>
<td>1504</td>
<td>kErrDeactivateDrawingFails</td>
<td>ADE fails to deactivate source drawing. ADE is unable to deactivate a drawing. Perhaps the drawing is locked by another user, the drawing no longer exists, or there are locked objects in the drawing. Check the error message stack for more information.</td>
</tr>
<tr>
<td>1505</td>
<td>kErrLongDrawingDescription</td>
<td>Specified drawing description exceeds 133 symbols. Shorten description.</td>
</tr>
<tr>
<td>Code</td>
<td>Error Code</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>-----------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>1506</td>
<td>kErrEntityHasBeenLocked</td>
<td>Drawing with locked entities can't be deactivated. ADEDRAWINGS - the specified drawing cannot be deactivated because it contains locked objects. Remove locks and deactivate.</td>
</tr>
<tr>
<td>1507</td>
<td>kErrPreviewNotSupported</td>
<td>ADE 2.0 ignores Tolerance, Body, Ellipse, 3dSolid, Region and Mline. If using ADEQUERY (Preview), ADEQVIEWDWGS, or ADEKEYVIEW, this message means ADE does not support Preview Query of Tolerance, Body, Ellipse, 3dSolid, Region and Mline.</td>
</tr>
<tr>
<td>1508</td>
<td>kErrAliasIsInUse</td>
<td>A drive alias of the same name already exists.</td>
</tr>
<tr>
<td>1509</td>
<td>kErrActivateDrawingCancelled</td>
<td>AcMapDrawingSetReactor::DrawingToBeActivated returns Adesk::kFalse, the attach operation is cancelled, and the error is returned to the application to handle.</td>
</tr>
<tr>
<td>1510</td>
<td>kErrAttachDrawingCancelled</td>
<td>AcMapDrawingSetReactor::DrawingToBeAttached returns Adesk::kFalse, the attach operation is cancelled, and the error is returned to the application to handle.</td>
</tr>
</tbody>
</table>

**Feature Alteration**

<table>
<thead>
<tr>
<th>Code</th>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1800</td>
<td>kErrInvalidFeatureType</td>
<td>Attempt to set an invalid property alteration type. If calling API functions ade_altpsetprop or ade_altpdefine, check for a mistake in property type.</td>
</tr>
<tr>
<td>1801</td>
<td>kErrNoListId</td>
<td>Property alteration internal list is invalid. Call support.</td>
</tr>
<tr>
<td>1802</td>
<td>kErrInvalidExpType</td>
<td>Attempt made to set an invalid property alteration expression type. Call support.</td>
</tr>
<tr>
<td>1803</td>
<td>kErrNoExpression</td>
<td>Property alteration internal object is invalid. Call support.</td>
</tr>
<tr>
<td>1804</td>
<td>kErrTextCreationFailed</td>
<td>Property alteration was unable to create a new text object. This message occurs when using queries that alter properties. Check expressions in the text property alteration definition.</td>
</tr>
<tr>
<td>1805</td>
<td>kErrHatchCreationFailed</td>
<td>Property alteration was unable to create a new hatch object. This message occurs when using queries that alter properties. Check expressions in the hatch property alteration definition.</td>
</tr>
<tr>
<td>1806</td>
<td>kErrInvalidColor</td>
<td>Invalid color passed to property alteration. This message occurs when using queries that alter properties. Check expressions evaluates to a valid AutoCAD color.</td>
</tr>
<tr>
<td>Code</td>
<td>Error Code</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-----------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>1807</td>
<td>kErrInvalidLayer</td>
<td>Invalid layer name. This message occurs when using queries that alter properties. Check expressions evaluates to a valid AutoCAD layer.</td>
</tr>
<tr>
<td>1808</td>
<td>kErrInvalidStyle</td>
<td>Invalid style name. This message occurs when using queries that alter properties. Check expressions evaluates to a valid style.</td>
</tr>
<tr>
<td>1809</td>
<td>kErrInvalidJustification</td>
<td>The expression for justification in a text property alteration did not evaluate to a valid justification. This message occurs when using queries that alter properties. Check expressions evaluates to a valid AutoCAD justification.</td>
</tr>
<tr>
<td>1810</td>
<td>kErrInvalidScale</td>
<td>ADE internal object is invalid. Call support.</td>
</tr>
<tr>
<td>1811</td>
<td>kErrNoRangeId</td>
<td>ADE internal object is invalid. Call support.</td>
</tr>
</tbody>
</table>

### Mapping

<table>
<thead>
<tr>
<th>Code</th>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1900</td>
<td>kErrMapCoincPoint</td>
<td>Coincident points. If using ADERSHEET or ADETRANSFORM this message means either old or new points are coincident. They must be different.</td>
</tr>
<tr>
<td>1901</td>
<td>kErrMapWrongScale</td>
<td>Invalid scale. Call support.</td>
</tr>
<tr>
<td>1902</td>
<td>kErrMapTransform</td>
<td>Can't transform entity. ADERSHEET, ADETRANSFORM, ADEQUERY, ADESSAVEOBJS An error appeared at the time of entity transformation. It is high-level error. There must be another error in the stack with more specific information.</td>
</tr>
<tr>
<td>1903</td>
<td>kErrMapWrongExtents</td>
<td>Invalid entity extents. If you're using ADETEXTLOC, ADERSHEET, ADEQUERY, ADESSAVEOBJS, or calling AcDbEntity::getGeomExtents() function you may get this message.</td>
</tr>
<tr>
<td>1904</td>
<td>kErrMapWrongPoints</td>
<td>Invalid points number. If you're using ADERSHEET, this message means that the numbers of old and new points are different, or less than 2. Dialog doesn't allow this. C.</td>
</tr>
<tr>
<td>1905</td>
<td>kErrMapWrongSelSet</td>
<td>Invalid selection set. Call support.</td>
</tr>
<tr>
<td>1906</td>
<td>kErrMapWrongEntityName</td>
<td>Invalid entity name. The entity is open. For</td>
</tr>
<tr>
<td>Code</td>
<td>Error Code</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>----------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>1907</td>
<td>kErrMapOpenEntity</td>
<td>Can't open entity. The entity is open. For example, another application opened the entity.</td>
</tr>
<tr>
<td>1908</td>
<td>kErrMapUpgradeEntity</td>
<td>Can't upgrade open. Entity modification occurred.</td>
</tr>
<tr>
<td>1909</td>
<td>kErrMapMoveStretchPoints</td>
<td>Can't modify stretch points. Call support.</td>
</tr>
<tr>
<td>1910</td>
<td>kErrMapEntityPoint</td>
<td>Can't modify entity points. Call support.</td>
</tr>
<tr>
<td>1911</td>
<td>kErrMapCmdecho</td>
<td>Can't change CMDECHO variable. Using ADEFILLPOLYG may produce this message.</td>
</tr>
<tr>
<td>1912</td>
<td>kErrMapCecolor</td>
<td>Can't change CECOLOR variable. Using ADEFILLPOLYG may produce this message.</td>
</tr>
<tr>
<td>1913</td>
<td>kErrMapHatch</td>
<td>Error in hatch command. Using ADEFILLPOLYG may produce this message.</td>
</tr>
<tr>
<td>1914</td>
<td>kErrMapWrongIntersectForPoints</td>
<td>Can't find intersection. Using ADEQUERY or calling AcDbentity::IntersectWith() function may produce this message.</td>
</tr>
<tr>
<td>1915</td>
<td>kErrMapWrongHandle</td>
<td>Wrong entity handle. Call support.</td>
</tr>
<tr>
<td>1916</td>
<td>kErrMapNotPolyline</td>
<td>Entity isn't polyline. Call support.</td>
</tr>
<tr>
<td>1917</td>
<td>kErrMapIterator</td>
<td>Can't create iterator. Call support.</td>
</tr>
<tr>
<td>1918</td>
<td>kErrMapWriteXData</td>
<td>Can't write Xdata. Ensure that Xdata size is 16 KB or less.</td>
</tr>
<tr>
<td>1919</td>
<td>kErrMapBuffer</td>
<td>Can't create buffer. If you're using ADEQUERY, to make a location query using a bufferfence, you may get this message.</td>
</tr>
<tr>
<td>1920</td>
<td>kErrMapStretchPoints</td>
<td>Can't get stretch points. If you're using ADETRANSFORM or ADEQUERY or calling AcDbEntity::getStretchPoints() you may get this message.</td>
</tr>
</tbody>
</table>

**Topology**

<table>
<thead>
<tr>
<th>Code</th>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>2000</td>
<td>kErrTopInvalidName</td>
<td>Invalid topology name. Occurs during topology creation.</td>
</tr>
<tr>
<td>2001</td>
<td>kErrTopExist</td>
<td>Topology already exists. Occurs during topology creation.</td>
</tr>
<tr>
<td>Code</td>
<td>Error Code</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>2003</td>
<td>kErrTopBuildPolygon</td>
<td>Error building polygon topology. Occurs during topology creation.</td>
</tr>
<tr>
<td>2004</td>
<td>kErrTopBuildNode</td>
<td>Can't create node. Occurs during topology creation.</td>
</tr>
<tr>
<td>2005</td>
<td>kErrTopBuildArc</td>
<td>Can't create link. Occurs during topology creation.</td>
</tr>
<tr>
<td>2006</td>
<td>kErrTopBuildCntr</td>
<td>Can't create centroid. Occurs during topology creation.</td>
</tr>
<tr>
<td>2007</td>
<td>kErrTopAPIReg</td>
<td>Can't register topology API. Occurs during ADE loading.</td>
</tr>
<tr>
<td>2008</td>
<td>kErrTopFuncNotAvail</td>
<td>Function isn't available. Occurs if you're using topology functions of the API.</td>
</tr>
<tr>
<td>2009</td>
<td>kErrTopWriteData</td>
<td>Error writing Xdata. Occurs during topology creation and modification.</td>
</tr>
<tr>
<td>2010</td>
<td>kErrTopNotExist</td>
<td>Topology doesn't exist. Occurs if you're using topology functions of the API.</td>
</tr>
<tr>
<td>2011</td>
<td>kErrTopOverlayType</td>
<td>Wrong overlay type.</td>
</tr>
<tr>
<td>2012</td>
<td>kErrTopMakeLayer</td>
<td>Can't create new layer. Using ADEDWGCLEAN produces this message.</td>
</tr>
<tr>
<td>2013</td>
<td>kErrTopBlockNotExist</td>
<td>Block doesn't exist.</td>
</tr>
<tr>
<td>2014</td>
<td>kErrTopNotOpenForWrite</td>
<td>Topology isn't open for write. Occurs when editing topology.</td>
</tr>
<tr>
<td>2015</td>
<td>kErrTopOpenIrdTable</td>
<td>Can't open object data table. Occurs when loading and editing topology.</td>
</tr>
<tr>
<td>2016</td>
<td>kErrTopWrongIrdAttr</td>
<td>Invalid object data table. Occurs when loading and editing topology.</td>
</tr>
<tr>
<td>2017</td>
<td>kErrTopLoaded</td>
<td>Topology is already loaded. Occurs when loading topology.</td>
</tr>
<tr>
<td>2018</td>
<td>kErrTopIncompleteElem</td>
<td>Incomplete topological element. Occurs when editing topology.</td>
</tr>
<tr>
<td>2019</td>
<td>kErrTopInvalidColor</td>
<td>Invalid color number. Occurs when using ADEDWGCLEAN and creating topology.</td>
</tr>
<tr>
<td>Year</td>
<td>Error Code</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>---------------------</td>
<td>-------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>2020</td>
<td>kErrTopInvalidFlag</td>
<td>Invalid flag. Occurs when using ADEDWGCLEAN.</td>
</tr>
<tr>
<td>2021</td>
<td>kErrTopInvalidTolerance</td>
<td>Invalid tolerance. Occurs when using ADEDWGCLEAN and creating topology.</td>
</tr>
<tr>
<td>2022</td>
<td>kErrTopInvalidCorridor</td>
<td>Invalid corridor width. Occurs when using ADEDWGCLEAN.</td>
</tr>
<tr>
<td>2023</td>
<td>kErrTopInvalidOffset</td>
<td>Invalid offset. Occurs when using buffering.</td>
</tr>
<tr>
<td>2024</td>
<td>kErrTopInvalidHeight</td>
<td>Invalid marker height. Occurs when using ADEDWGCLEAN.</td>
</tr>
<tr>
<td>2025</td>
<td>kErrTopInvalidMarkerType</td>
<td>Invalid marker type. Occurs when using ADEDWGCLEAN.</td>
</tr>
<tr>
<td>2026</td>
<td>kErrTopInvalidEntityTypeName</td>
<td>Invalid type for new entities. Occurs when using ADEDWGCLEAN.</td>
</tr>
<tr>
<td>2027</td>
<td>kErrTopInvalidErrorType</td>
<td>Invalid error type. Occurs when using ADEDWGCLEAN.</td>
</tr>
<tr>
<td>2028</td>
<td>kErrTopIntersection</td>
<td>Intersections detected. Occurs when creating and editing polygon topology.</td>
</tr>
<tr>
<td>2029</td>
<td>kErrTopOverlayItself</td>
<td>Can't overlay topology with itself.</td>
</tr>
<tr>
<td>2030</td>
<td>kErrTopSourceDwgAccess</td>
<td>Can't access source drawing.</td>
</tr>
<tr>
<td>2031</td>
<td>kErrTopSourceDwgNotActive</td>
<td>Source drawing isn't active.</td>
</tr>
<tr>
<td>2032</td>
<td>kErrTopSourceDatabaseAccess</td>
<td>Can't access source drawing database.</td>
</tr>
<tr>
<td>2033</td>
<td>kErrTopSourceObjectId</td>
<td>Can't get object ID by handle in source drawing database.</td>
</tr>
<tr>
<td>2034</td>
<td>kErrTopNotLoaded</td>
<td>Topology isn't loaded.</td>
</tr>
<tr>
<td>2035</td>
<td>kErrTopImplicitNode</td>
<td>Node object doesn't exist in node topology.</td>
</tr>
<tr>
<td>2036</td>
<td>kErrTopMisplacedNode</td>
<td>Wrong node coordinates.</td>
</tr>
<tr>
<td>2037</td>
<td>kErrTopUnreferencedNode</td>
<td>Node isn't referenced in links.</td>
</tr>
<tr>
<td>2038</td>
<td>kErrTopUnexistentNode</td>
<td>Link references nonexistent node.</td>
</tr>
<tr>
<td>2039</td>
<td>kErrTopMismatchStartNode</td>
<td>Link has invalid ID at the start node.</td>
</tr>
<tr>
<td>2040</td>
<td>kErrTopMismatchEndNode</td>
<td>Link has invalid ID at the end node.</td>
</tr>
<tr>
<td>2041</td>
<td>kErrTopMisplacedCentroid</td>
<td>Wrong centroid coordinates.</td>
</tr>
<tr>
<td>Code</td>
<td>Error Code</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>------------------------------------------------</td>
<td>------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>2042</td>
<td>kErrTopMismatchLeftPoly</td>
<td>Link has invalid ID for the left polygon.</td>
</tr>
<tr>
<td>2043</td>
<td>kErrTopMismatchRightPoly</td>
<td>Link has invalid ID for the right polygon.</td>
</tr>
<tr>
<td>2044</td>
<td>kErrTopUnexistentCentroid</td>
<td>Centroid isn't inside polygon.</td>
</tr>
<tr>
<td>2045</td>
<td>kErrTopMultiplyCentroid</td>
<td>Polygon has several centroids inside.</td>
</tr>
<tr>
<td>2046</td>
<td>kErrTopWrongPolyQty</td>
<td>Some polygons are incorrect.</td>
</tr>
<tr>
<td>2047</td>
<td>kErrTopMismatchPolyArea</td>
<td>Incorrect polygon area.</td>
</tr>
<tr>
<td>2048</td>
<td>kErrTopMismatchPolyPerimeter</td>
<td>Incorrect polygon perimeter.</td>
</tr>
<tr>
<td>2049</td>
<td>kErrTopOpenSourceDwgTopo</td>
<td>Topology loaded from source drawings can't be open for write.</td>
</tr>
<tr>
<td>2050</td>
<td>kErrTopOpenTempTopo</td>
<td>Temporary topology can't be open for write.</td>
</tr>
<tr>
<td>2051</td>
<td>kErrTopIdNotExist</td>
<td>Current drawing doesn't have OD table with information about last ID.</td>
</tr>
<tr>
<td>2052</td>
<td>kErrTopEmpty</td>
<td>Can't create or load empty topology.</td>
</tr>
<tr>
<td>2053</td>
<td>kErrTopWasModified</td>
<td>Topological objects were modified by AutoCAD commands.</td>
</tr>
<tr>
<td>2054</td>
<td>kErrTopMultiple</td>
<td>Object belongs to multiple topologies and can't be erased.</td>
</tr>
<tr>
<td>2055</td>
<td>kErrTopCalculateOffset</td>
<td>Can't calculate offset. Use default. Occurs when using buffering.</td>
</tr>
<tr>
<td>2056</td>
<td>kErrTopZeroOffset</td>
<td>Zero offset. Can't build buffer.</td>
</tr>
<tr>
<td>2057</td>
<td>kErrTopDifferentOffset</td>
<td>Offset has different sign for some objects. Can't build buffer.</td>
</tr>
<tr>
<td>2058</td>
<td>kErrTopInvalidSelSet</td>
<td>Invalid selection set. Occurs when using the API.</td>
</tr>
<tr>
<td>2059</td>
<td>kErrTopCleanNotInit</td>
<td>Cleadup model isn't initialized. Occurs when using the API.</td>
</tr>
<tr>
<td>2060</td>
<td>kErrTopCleanNoGroup</td>
<td>There is no current group. Occurs when using the API.</td>
</tr>
<tr>
<td>2061</td>
<td>kErrTopCleanInvalidIndex</td>
<td>Invalid error index. Occurs when using the API.</td>
</tr>
<tr>
<td>2062</td>
<td>kErrTopCleanNoError</td>
<td>Current error isn't set. Occurs when using the API.</td>
</tr>
<tr>
<td>2063</td>
<td>kErrTopTraceLinkNotExist</td>
<td>Link doesn't exist in tracing model. Occurs when</td>
</tr>
</tbody>
</table>
### Topology API

<table>
<thead>
<tr>
<th>Code</th>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>2064</td>
<td>kErrTopTraceNodeNotExist</td>
<td>Node doesn't exist in tracing model. Occurs when using the API.</td>
</tr>
<tr>
<td>2065</td>
<td>kErrTopTraceNoPath</td>
<td>Result path isn't calculated. Occurs when using the API.</td>
</tr>
<tr>
<td>2066</td>
<td>kErrTopTraceInvalidIndex</td>
<td>Invalid element index. Occurs when using the API.</td>
</tr>
<tr>
<td>2067</td>
<td>kErrTopInvalidExpression</td>
<td>Can't process ADE expression. Occurs when using overlay, buffer, dissolve, or tracing command.</td>
</tr>
<tr>
<td>2068</td>
<td>kErrTopLockedTable</td>
<td>Can't write into topology OD table. Occurs when using dissolve command.</td>
</tr>
<tr>
<td>2069</td>
<td>kErrTopCreateTable</td>
<td>Can't create OD table. Occurs when using topology creation, overlay, buffer, or dissolve commands.</td>
</tr>
<tr>
<td>2070</td>
<td>kErrTopCreateTableColumn</td>
<td>Can't add column to OD table. Occurs when using topology creation, overlay, buffer, or dissolve commands.</td>
</tr>
<tr>
<td>2071</td>
<td>kErrTopTraceNodesEqual</td>
<td>Start and end nodes are the same. Occurs doing shortest path tracing.</td>
</tr>
<tr>
<td>2072</td>
<td>kErrTopTracePathNotExist</td>
<td>Empty path. Occurs during shortest path tracing.</td>
</tr>
<tr>
<td>2073</td>
<td>kErrTopTraceFloodNotExist</td>
<td>Empty path. Occurs when tracing floods.</td>
</tr>
<tr>
<td>2074</td>
<td>kErrTopRenameDisabled</td>
<td>Can't rename topology, because current drawing has queried objects with OD.</td>
</tr>
<tr>
<td>2075</td>
<td>kErrTopDeleteDisabled</td>
<td>Can't delete topology, because current drawing has queried objects with OD.</td>
</tr>
<tr>
<td>2076</td>
<td>kErrTopInvalidExtents</td>
<td></td>
</tr>
</tbody>
</table>

### Tracing

<table>
<thead>
<tr>
<th>Code</th>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>2100</td>
<td>kErrTopApiErrWrongInput</td>
<td>Missing or invalid parameter.</td>
</tr>
<tr>
<td>2101</td>
<td>kErrTopApiWrongId</td>
<td>Invalid ID.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Code</th>
<th>Error Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>2150</td>
<td>kErrTopSprErr</td>
</tr>
<tr>
<td>Code</td>
<td>Error Code</td>
</tr>
<tr>
<td>-------</td>
<td>-------------------------------------</td>
</tr>
<tr>
<td>2200</td>
<td>kErrUnexpectedBuffChar</td>
</tr>
<tr>
<td>2201</td>
<td>kErrInvalidIndex</td>
</tr>
<tr>
<td>2202</td>
<td>kErrInvalidQueryLine</td>
</tr>
<tr>
<td>2203</td>
<td>kErrInvalidName</td>
</tr>
<tr>
<td>2204</td>
<td>kErrEntryAlreadyExists</td>
</tr>
<tr>
<td>2205</td>
<td>kErrEntryInOtherCategory</td>
</tr>
<tr>
<td>2206</td>
<td>kErrEntryAndFileAlreadyExist</td>
</tr>
<tr>
<td>2207</td>
<td>kErrASIConnectFailed</td>
</tr>
<tr>
<td>2208</td>
<td>kErrASISmtPrepareFailed</td>
</tr>
<tr>
<td>2209</td>
<td>kErrASICsrAllocFailed</td>
</tr>
<tr>
<td>2210</td>
<td>kErrASICsrOpenFailed</td>
</tr>
<tr>
<td>2211</td>
<td>kErrInvalidDOName</td>
</tr>
<tr>
<td>Code</td>
<td>Error Code</td>
</tr>
<tr>
<td>--------</td>
<td>--------------------</td>
</tr>
<tr>
<td>2212</td>
<td>kErrLPInitFailed</td>
</tr>
<tr>
<td>2213</td>
<td>kErrColNotFound</td>
</tr>
<tr>
<td>2214</td>
<td>kErrQDefNotInTM</td>
</tr>
<tr>
<td>2215</td>
<td>kErrQryDefnExists</td>
</tr>
<tr>
<td>2216</td>
<td>kErrInvalidOperator</td>
</tr>
<tr>
<td>2217</td>
<td>kErrInvalidPtnOperator</td>
</tr>
<tr>
<td>2218</td>
<td>kErrInvalidField</td>
</tr>
<tr>
<td>2219</td>
<td>kErrInvalidNotBranch</td>
</tr>
<tr>
<td>2220</td>
<td>kErrInvalidBranch</td>
</tr>
<tr>
<td>2221</td>
<td>kErrUndefinedValue</td>
</tr>
<tr>
<td>2222</td>
<td>kErrInvalidLocationType</td>
</tr>
<tr>
<td>2223</td>
<td>kErrCantLoadExternQuery</td>
</tr>
</tbody>
</table>

**Query Manager**

<table>
<thead>
<tr>
<th>Code</th>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>2300</td>
<td>kErrIntersectFailed</td>
<td>A call to CAseLinkSel::intersectPartialKey failed. Look at the ASE error displayed.</td>
</tr>
<tr>
<td>2301</td>
<td>kErrNoTemplate</td>
<td>The query type was specified as report but no report options were defined. Define report options.</td>
</tr>
<tr>
<td>Code</td>
<td>Error Code</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>---------------</td>
<td>----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>2302</td>
<td>kErrASIStoreValueFailed</td>
<td>A call to CAsiData ::storeValue failed. Look at the ASI error displayed.</td>
</tr>
<tr>
<td>2303</td>
<td>kErrASIGetValueFailed</td>
<td>A call to CAsiData ::getValue failed. Look at the ASI error displayed.</td>
</tr>
</tbody>
</table>

**Utility**

<table>
<thead>
<tr>
<th>Code</th>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>2400</td>
<td>kErrLicFatal</td>
<td>Fatal error in ADE license. Call support.</td>
</tr>
<tr>
<td>2401</td>
<td>kErrFileNotFound</td>
<td>Can't find associated document. Occurs when using ADEDOCVIEW.</td>
</tr>
<tr>
<td>2402</td>
<td>kErrPathNotFound</td>
<td>Can't find executable file. Occurs when using ADEDOCVIEW.</td>
</tr>
<tr>
<td>2403</td>
<td>kErrBadFormat</td>
<td>Syntax error in the command line. Occurs when using ADEDOCVIEW.</td>
</tr>
<tr>
<td>2404</td>
<td>kErrConvErr</td>
<td>Error converting ADE 1.0 data to ADE 2.0 data. Occurs when using ADECONVERT.</td>
</tr>
</tbody>
</table>

**Data Dialogs**

<table>
<thead>
<tr>
<th>Code</th>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>2450</td>
<td>kErrIRDMismatch</td>
<td></td>
</tr>
<tr>
<td>2451</td>
<td>kErrIRDInvalidName</td>
<td></td>
</tr>
<tr>
<td>2452</td>
<td>kErrIRDTTableExists</td>
<td></td>
</tr>
<tr>
<td>2453</td>
<td>kErrInvalidTableName</td>
<td></td>
</tr>
<tr>
<td>2454</td>
<td>kErrInvalidAttrName</td>
<td></td>
</tr>
<tr>
<td>2455</td>
<td>kErrTopoName</td>
<td></td>
</tr>
<tr>
<td>2456</td>
<td>kErrQueriedAndNotNew</td>
<td></td>
</tr>
<tr>
<td>2457</td>
<td>kErrNotAdministrator</td>
<td></td>
</tr>
</tbody>
</table>

**GenLink**

<table>
<thead>
<tr>
<th>Code</th>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>2500</td>
<td>kErrTagNotFound</td>
<td></td>
</tr>
<tr>
<td>2501</td>
<td>kErrTagValueAbsent</td>
<td></td>
</tr>
<tr>
<td>2502</td>
<td>kErrIllegalFormat</td>
<td></td>
</tr>
</tbody>
</table>
### Environment

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>2600</td>
<td>kErrInitEnv  An error occurred during the initialization of ADE. The cause of this error may be due to errors in loading/initializing ADE user preferences, system preferences, log file, or user list.</td>
</tr>
<tr>
<td>2601</td>
<td>kErrCantFindAdeExePath</td>
</tr>
<tr>
<td>2602</td>
<td>kErrINIWrite</td>
</tr>
<tr>
<td>2603</td>
<td>kErrInvalidUserName  The user name specified does not exist in the user list. Use a user name that already exists in the user list or define a new one using User Administration.</td>
</tr>
<tr>
<td>2604</td>
<td>kErrLoadUserList</td>
</tr>
<tr>
<td>2605</td>
<td>kErrSaveUserList</td>
</tr>
<tr>
<td>2606</td>
<td>kErrInvalidPswd  The password specified does not match the one specified in the user list for this user. Use the correct password.</td>
</tr>
</tbody>
</table>

### Rx

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>2700</td>
<td>kErrRxAseLoad  ASE isn't loaded Can't initialize ASE API.</td>
</tr>
<tr>
<td>2701</td>
<td>kErrRxAselInit  Object Data module isn't loaded.</td>
</tr>
<tr>
<td>2702</td>
<td>kErrRxIrdLoad  Can't initialize Object Data API.</td>
</tr>
<tr>
<td>2703</td>
<td>kErrRxIrdInit  Specified Coord system category not found in the library. Call support.</td>
</tr>
</tbody>
</table>

### Projection

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>2800</td>
<td>kErrNoProjCatFound  Specified Coord system category not found in the library. Call support.</td>
</tr>
<tr>
<td>2801</td>
<td>kErrNoDatumFound  Specified Coord system datum not found. Call support.</td>
</tr>
<tr>
<td>2802</td>
<td>kErrNoElipFound  Ellipsoid not found in the ellipsoid list. Call support.</td>
</tr>
<tr>
<td>2803</td>
<td>kErrNoCoordFound  Specified Coord Sys not found. Call support.</td>
</tr>
<tr>
<td>Code</td>
<td>Error Code</td>
</tr>
<tr>
<td>-------</td>
<td>-----------------------------------</td>
</tr>
<tr>
<td>2804</td>
<td>kErrFaileOpenDatumFile</td>
</tr>
<tr>
<td>2805</td>
<td>kErrFaileOpenElipFile</td>
</tr>
<tr>
<td>2806</td>
<td>kErrNoneCoord</td>
</tr>
<tr>
<td>2900</td>
<td>kErrNoExpressionFound</td>
</tr>
<tr>
<td>2901</td>
<td>kErrGetPropFail</td>
</tr>
<tr>
<td>2902</td>
<td>kErrExpEvalFail</td>
</tr>
<tr>
<td>2903</td>
<td>kErrExpMissingQuote</td>
</tr>
<tr>
<td>2904</td>
<td>kErrExpMissingCParen</td>
</tr>
<tr>
<td>2905</td>
<td>kErrExpExceedThreeOper</td>
</tr>
<tr>
<td>2906</td>
<td>kErrRngTabNameExist</td>
</tr>
<tr>
<td>3000</td>
<td>kErrInvalidIndexVersion</td>
</tr>
</tbody>
</table>
The index in the drawing is out-of-date. Regenerate the index using drawing maintenance.

### Validation

<table>
<thead>
<tr>
<th>Code</th>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>3001</td>
<td>kErrIndexOutOfRange</td>
<td>The index in the drawing is out-of-date. Regenerate the index using drawing maintenance.</td>
</tr>
<tr>
<td>3002</td>
<td>kErrTypeAllObjects</td>
<td></td>
</tr>
<tr>
<td>3003</td>
<td>kErrTypeNoOneObject</td>
<td></td>
</tr>
<tr>
<td>3100</td>
<td>kErrWrongSymbolName</td>
<td></td>
</tr>
<tr>
<td>3101</td>
<td>kErrWrongSymbol</td>
<td></td>
</tr>
<tr>
<td>3102</td>
<td>kErrWrongStrLength</td>
<td></td>
</tr>
<tr>
<td>3103</td>
<td>kErrDirDoesNotExist</td>
<td></td>
</tr>
<tr>
<td>3104</td>
<td>kErrDirReadOnly</td>
<td></td>
</tr>
<tr>
<td>3105</td>
<td>kErrAccessDenied</td>
<td></td>
</tr>
<tr>
<td>3106</td>
<td>kErrFileDoesNotExist</td>
<td></td>
</tr>
<tr>
<td>3107</td>
<td>kErrFileAlreadyExists</td>
<td></td>
</tr>
<tr>
<td>3108</td>
<td>kErrFileOpenFailed</td>
<td></td>
</tr>
<tr>
<td>3109</td>
<td>kErrFileReadOnly</td>
<td></td>
</tr>
<tr>
<td>3110</td>
<td>kErrInvalidString</td>
<td></td>
</tr>
<tr>
<td>3111</td>
<td>kErrOutOfRange</td>
<td></td>
</tr>
<tr>
<td>3112</td>
<td>kErrWrongColor</td>
<td></td>
</tr>
<tr>
<td>3113</td>
<td>kErrIncorrectParameters</td>
<td>One of ADE validation functions recognized incorrect input parameters. This error is an internal ADE error.</td>
</tr>
<tr>
<td>3114</td>
<td>kErrFileOpenLimit</td>
<td></td>
</tr>
<tr>
<td>3115</td>
<td>kErrShareViolation</td>
<td></td>
</tr>
<tr>
<td>3116</td>
<td>kErrNetAccessDenied</td>
<td></td>
</tr>
<tr>
<td>3117</td>
<td>kErrPathDoesNotExist</td>
<td></td>
</tr>
</tbody>
</table>

## File Locking
<table>
<thead>
<tr>
<th>Code</th>
<th>Error Description</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>3200</td>
<td>kErrDwkFileDoesNotExist</td>
<td>ADE lock file is locked. Occurs when using ADEDRAWINGS and ADEQUERY commands and when ADE is running in a multi-user environment.</td>
</tr>
<tr>
<td>3201</td>
<td>kErrOpenDwkFileFailed</td>
<td>ADE was unable to open the .DWK lock file. Call support.</td>
</tr>
<tr>
<td>3202</td>
<td>kErrFileLockedByAcad</td>
<td>Attempt to remove a user who does not exist from the lock file. Call support.</td>
</tr>
<tr>
<td>3203</td>
<td>kErrOldMapLockFile</td>
<td>ADE was unable to create the .DWK lock file. Call support.</td>
</tr>
<tr>
<td>3204</td>
<td>kErrFileIsNotDwk</td>
<td>ADE tried to lock a file for write that was already locked for read. Call support.</td>
</tr>
<tr>
<td>3205</td>
<td>kErrSpecifiedUserDoesNotExist</td>
<td>ADE tried to lock a file for read that was already locked for write. Occurs during query operations in a multi-user environment.</td>
</tr>
<tr>
<td>3206</td>
<td>kErrCreateDwkFileFailed</td>
<td>ADE tried to open and read a file that was not a valid .DWK file. Call support.</td>
</tr>
<tr>
<td>3207</td>
<td>kErrFileIsLockedForRead</td>
<td>ADE was unable to unlock the lock file. Call support.</td>
</tr>
<tr>
<td>3208</td>
<td>kErrFileIsLockedForWrite</td>
<td>ADE tried to attach a file that is already open by AutoCAD. Occurs when using ADEDRAWINGS with ATTACH operations if the file is open in an AutoCAD project.</td>
</tr>
<tr>
<td>3209</td>
<td>kErrInvalidLockStateSpecified</td>
<td>ADE internal object is invalid. Call support.</td>
</tr>
<tr>
<td>3210</td>
<td>kErrNotOwnerOfWLH</td>
<td>ADE tried to remove a write lock when the user did not have a write lock. Call support.</td>
</tr>
<tr>
<td>3211</td>
<td>kErrUserIsNotWriter</td>
<td></td>
</tr>
<tr>
<td>3212</td>
<td>kErrUserIsNotReader</td>
<td></td>
</tr>
<tr>
<td>3213</td>
<td>kErrUserHasReadLock</td>
<td></td>
</tr>
<tr>
<td>3214</td>
<td>kErrLockFileIsFull</td>
<td></td>
</tr>
<tr>
<td>3215</td>
<td>kErrDwgFileDoesNotExist</td>
<td>ADE tried to unlock a file but the .DWK file was missing. Occurs if the .dwk file was erased after a file was attached.</td>
</tr>
<tr>
<td>3216</td>
<td>kErrNotAnADELockFile</td>
<td>ADE internal object is invalid. Call support.</td>
</tr>
<tr>
<td>3217</td>
<td>kErrFileMayHaveBeenModified</td>
<td>Existing .DWK file does not belong to ADE. the lock</td>
</tr>
</tbody>
</table>
file exists and can be read by ADE, but ADE does not own the file.

<table>
<thead>
<tr>
<th>Code</th>
<th>Error Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>3218</td>
<td>kErrFileHasLocks</td>
</tr>
<tr>
<td></td>
<td>ADE tried to remove a lock file but it was not found. Occurs if the .dwk file was erased after a file was attached.</td>
</tr>
</tbody>
</table>

**Unicode Support**

<table>
<thead>
<tr>
<th>Code</th>
<th>Error Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>4000</td>
<td>kErrUnicodeInsufficientBufferToConvert</td>
</tr>
<tr>
<td>4001</td>
<td>kErrUnicodeInvalidFlagsToConvert</td>
</tr>
<tr>
<td>4002</td>
<td>kErrUnicodeInvalidParameterToConvert</td>
</tr>
<tr>
<td>4003</td>
<td>kErrUnicodeNoTranslation</td>
</tr>
<tr>
<td>4004</td>
<td>kErrUnicodeCodePageNotAvailable</td>
</tr>
</tbody>
</table>

**Double-Byte Support**

<table>
<thead>
<tr>
<th>Code</th>
<th>Error Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>4005</td>
<td>kErrNoMBCSAllowed</td>
</tr>
</tbody>
</table>
AcMap::EErrType enumerators represent error types.

Errors on the error stack are represented by AcMapErrorEntry objects. Error types are returned by the AcMapErrorEntry::ErrorType function.

<table>
<thead>
<tr>
<th>Code</th>
<th>Enum</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>kAdeNoMessage</td>
<td>Error condition does not exist.</td>
</tr>
<tr>
<td>01</td>
<td>kAdeWarning</td>
<td>ADE (AutoCAD Data Extension) execution warning.</td>
</tr>
<tr>
<td>02</td>
<td>kAdeError</td>
<td>ADE execution error.</td>
</tr>
<tr>
<td>03</td>
<td>kAseWarning</td>
<td>ASE (AutoCAD SQL Extension) execution warning.</td>
</tr>
<tr>
<td>04</td>
<td>kAseError</td>
<td>ASE execution error.</td>
</tr>
<tr>
<td>05</td>
<td>kAcWarning</td>
<td>AutoCAD execution warning.</td>
</tr>
<tr>
<td>06</td>
<td>kAcError</td>
<td>AutoCAD execution error.</td>
</tr>
<tr>
<td>07</td>
<td>kAsiWarning</td>
<td>ASI (AutoCAD SQL Interface) execution warning.</td>
</tr>
<tr>
<td>08</td>
<td>kAsiError</td>
<td>ASI execution error.</td>
</tr>
<tr>
<td>09</td>
<td>kIRDWarning</td>
<td>Extended object data (Xdata) warning.</td>
</tr>
<tr>
<td>10</td>
<td>kIRDError</td>
<td>Xdata error.</td>
</tr>
<tr>
<td>13</td>
<td>kMapError</td>
<td>AutoCAD Map execution error.</td>
</tr>
<tr>
<td>14</td>
<td>kMentorError</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>kApplicationError</td>
<td>Operating system level application error.</td>
</tr>
<tr>
<td>16</td>
<td>kDiagMessage</td>
<td>Diagnostic message returned.</td>
</tr>
</tbody>
</table>
Join operators for query conditions.

<table>
<thead>
<tr>
<th>First</th>
<th>First</th>
<th>First</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>kOperatorOr</td>
<td></td>
</tr>
</tbody>
</table>

AcMap::EJoinOperator
Location condition types.

<table>
<thead>
<tr>
<th>First</th>
<th>First</th>
<th>First</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>kLocationCrossing</td>
<td></td>
</tr>
<tr>
<td>AcMap::EOpenMode</td>
<td>Value</td>
<td></td>
</tr>
<tr>
<td>--------------------</td>
<td>---------</td>
<td></td>
</tr>
<tr>
<td>kAdeClosed</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>kOpenForWrite</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>kOpenForRead</td>
<td>2</td>
<td></td>
</tr>
</tbody>
</table>
Option types.

<table>
<thead>
<tr>
<th>Option</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>kWSpaceType</td>
<td>Workspace.</td>
</tr>
<tr>
<td>1</td>
<td>kWSessType</td>
<td>Work session.</td>
</tr>
<tr>
<td>2</td>
<td>kQryType</td>
<td>Query.</td>
</tr>
<tr>
<td>3</td>
<td>kSvBkType</td>
<td>Save back.</td>
</tr>
<tr>
<td>4</td>
<td>kExtDbType</td>
<td>Database.</td>
</tr>
<tr>
<td>5</td>
<td>kCoordXformType</td>
<td>Coordinate transformation.</td>
</tr>
<tr>
<td>6</td>
<td>kSystemType</td>
<td>System.</td>
</tr>
<tr>
<td>7</td>
<td>kCoordsysType</td>
<td>Coordinate system.</td>
</tr>
</tbody>
</table>
Source for preview definitions.

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>kProject</td>
</tr>
<tr>
<td>2</td>
<td>kSource</td>
</tr>
<tr>
<td>3</td>
<td>kLibrary</td>
</tr>
</tbody>
</table>
Types of preferences.

<table>
<thead>
<tr>
<th></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>kQryPrefDefn</td>
</tr>
<tr>
<td>01</td>
<td>kSvBkPrefDefn</td>
</tr>
<tr>
<td>02</td>
<td>kCoordXformPrefDefn</td>
</tr>
</tbody>
</table>
Property condition types.

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>kArea</td>
</tr>
<tr>
<td>01</td>
<td>kBlockName</td>
</tr>
<tr>
<td>02</td>
<td>kColor</td>
</tr>
<tr>
<td>03</td>
<td>kElevation</td>
</tr>
<tr>
<td>04</td>
<td>kEntType</td>
</tr>
<tr>
<td>05</td>
<td>kGroup</td>
</tr>
<tr>
<td>06</td>
<td>kLayer</td>
</tr>
<tr>
<td>07</td>
<td>kLength</td>
</tr>
<tr>
<td>08</td>
<td>kLineType</td>
</tr>
<tr>
<td>09</td>
<td>kTextStyle</td>
</tr>
<tr>
<td>10</td>
<td>kTextValue</td>
</tr>
<tr>
<td>11</td>
<td>kThickness</td>
</tr>
<tr>
<td>12</td>
<td>kFeature</td>
</tr>
<tr>
<td>13</td>
<td>kLineweight</td>
</tr>
<tr>
<td>14</td>
<td>kPlotstyle</td>
</tr>
</tbody>
</table>
Query dialog options.

<table>
<thead>
<tr>
<th></th>
<th>AcMap::EQueryDialogOptions</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>kQueryDialogSuppressNothing</td>
</tr>
<tr>
<td>01</td>
<td>kQueryDialogSuppressExecute</td>
</tr>
<tr>
<td>02</td>
<td>kQueryDialogSuppressPreview</td>
</tr>
<tr>
<td>04</td>
<td>kQueryDialogSuppressReport</td>
</tr>
<tr>
<td>08</td>
<td>kQueryDialogSuppressAlteration</td>
</tr>
<tr>
<td>16</td>
<td>kQueryDialogSuppressDrawings</td>
</tr>
<tr>
<td>32</td>
<td>kQueryDialogSuppressMore</td>
</tr>
</tbody>
</table>
Query modes.

<table>
<thead>
<tr>
<th></th>
<th>EQueryType</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>kQueryDraw</td>
</tr>
<tr>
<td>1</td>
<td>kQueryPreview</td>
</tr>
<tr>
<td>2</td>
<td>kQueryReport</td>
</tr>
</tbody>
</table>
Comparison operators for range lines.

<table>
<thead>
<tr>
<th></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>kRangeEq</td>
</tr>
<tr>
<td>2</td>
<td>kRangeGT</td>
</tr>
<tr>
<td>3</td>
<td>kRangeGTorEq</td>
</tr>
<tr>
<td>4</td>
<td>kRangeLT</td>
</tr>
<tr>
<td>5</td>
<td>kRangeLTorEq</td>
</tr>
<tr>
<td>6</td>
<td>kRangeNotEq</td>
</tr>
<tr>
<td>7</td>
<td>kRangeOtherwise</td>
</tr>
</tbody>
</table>
Save query options.

<table>
<thead>
<tr>
<th></th>
<th>kSaveDrawingSet</th>
<th>Saves drawing objects.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>kSaveLocationCoordinates</td>
<td>Saves location coordinates.</td>
</tr>
<tr>
<td>2</td>
<td>kSavePropertyAlteration</td>
<td>Saves property alterations.</td>
</tr>
<tr>
<td>4</td>
<td>kAutoExecute</td>
<td>Executes a query when it is loaded.</td>
</tr>
</tbody>
</table>
Symbol table types.

<table>
<thead>
<tr>
<th>Symbol ID</th>
<th>Table Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>kBlockTable</td>
<td>Block names.</td>
</tr>
<tr>
<td>1</td>
<td>kLayerTable</td>
<td>Layer names.</td>
</tr>
<tr>
<td>2</td>
<td>kLinetypeTable</td>
<td>Line types.</td>
</tr>
<tr>
<td>3</td>
<td>kStyleTable</td>
<td>Text styles.</td>
</tr>
<tr>
<td>4</td>
<td>kRegappTable</td>
<td>AutoCAD regapps.</td>
</tr>
<tr>
<td>5</td>
<td>kGroupTable</td>
<td>Group names.</td>
</tr>
<tr>
<td>6</td>
<td>kLinkTemplate</td>
<td>Link template names.</td>
</tr>
<tr>
<td>7</td>
<td>kODD</td>
<td>Object data table names.</td>
</tr>
<tr>
<td>8</td>
<td>kMLinestyleTable</td>
<td>Line styles.</td>
</tr>
<tr>
<td>9</td>
<td>kFeatureTable</td>
<td>Feature names.</td>
</tr>
<tr>
<td>10</td>
<td>kLineweightTable</td>
<td>Line weights.</td>
</tr>
<tr>
<td>11</td>
<td>kPlotstyleTable</td>
<td>Plot styles.</td>
</tr>
</tbody>
</table>
User rights.

<table>
<thead>
<tr>
<th>#</th>
<th>AcMap::kRightsSuperUser</th>
<th>Superuser</th>
</tr>
</thead>
<tbody>
<tr>
<td>02</td>
<td>AcMap::kRightsAltDwgSet</td>
<td>Can alter drawing set</td>
</tr>
<tr>
<td>04</td>
<td>AcMap::kRightsEditDwg</td>
<td>Can edit drawings</td>
</tr>
<tr>
<td>08</td>
<td>AcMap::kRightsDrawQuery</td>
<td>Can run Draw queries</td>
</tr>
<tr>
<td>16</td>
<td>AcMap::kRightsAltClass</td>
<td>Can edit Feature Class definitions</td>
</tr>
</tbody>
</table>
Types of the objects in the save set.

<table>
<thead>
<tr>
<th></th>
<th>kQueriedExisted</th>
<th>Queried objects that are unchanged or were altered by the query.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>kQueriedExisted</td>
<td>Queried objects that are unchanged or were altered by the query.</td>
</tr>
<tr>
<td>2</td>
<td>kQueriedErased</td>
<td>Queried objects that were deleted.</td>
</tr>
<tr>
<td>4</td>
<td>kNewlyCreated</td>
<td>New objects added to model space.</td>
</tr>
</tbody>
</table>
The coordinate transformation functions begin with *ade_proj*.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ade_projentitybackward</strong></td>
<td>Transforms an entity from the destination coordinate system to the source coordinate system.</td>
</tr>
<tr>
<td><strong>ade_projentityforward</strong></td>
<td>Transforms an entity from the source coordinate system to the destination coordinate system.</td>
</tr>
<tr>
<td><strong>ade_progetctgynme</strong></td>
<td>Identifies the category that a coordinate system belongs to.</td>
</tr>
<tr>
<td><strong>ade_progetinfo</strong></td>
<td>Gets information about a projection system.</td>
</tr>
<tr>
<td><strong>ade_progetwscode</strong></td>
<td>Gets the project drawing's coordinate system code.</td>
</tr>
<tr>
<td><strong>ade_projlistctgy</strong></td>
<td>Lists available coordinate system categories.</td>
</tr>
<tr>
<td><strong>ade_projlistcrdsyts</strong></td>
<td>Lists available coordinate systems in a given category.</td>
</tr>
<tr>
<td><strong>ade_projptbackward</strong></td>
<td>Computes new coordinates for a source point.</td>
</tr>
<tr>
<td><strong>ade_projptforward</strong></td>
<td>Computes new coordinates for a destination point.</td>
</tr>
<tr>
<td><strong>ade_projsetdest</strong></td>
<td>Sets the destination coordinate system.</td>
</tr>
<tr>
<td><strong>ade_projsetsrc</strong></td>
<td>Sets the source coordinate system.</td>
</tr>
<tr>
<td><strong>ade_projsetwscode</strong></td>
<td>Sets the project drawing's coordinate system.</td>
</tr>
<tr>
<td><strong>ade_projwsgeodistance</strong></td>
<td>Measures the geodetic distance between two points.</td>
</tr>
</tbody>
</table>
ade_aliasadd

Drive Alias Functions

Creates a drive alias.

```c
int ade_aliasadd(
    char* alias_name,
    char* path_name);
```

Returns RTNORM or an error code.

- **alias_name** The name of the drive alias, which can be up to 31 characters. The name must be unique, contain no spaces, and start with an alphanumeric character.
- **path_name** The drive and path to which the drive alias refers.

The function adds a new drive alias to the drive alias list. For example:

```c
ade_aliasadd (    
    "ADSRX_SAMPLE",
    "C:\ADSRX\Drawings\PlotMapSet");
```
Delete a drive alias.

```c
int ade_aliasdelete(char* alias_name);
```

Returns RTNORM or an error code.

**alias_name**
The name of the drive alias, which can be up to 31 characters. The name must be unique, contain no spaces, and start with an alphanumeric character.
ade_aliasgetlist

Drive Alias Functions

Lists all drive aliases in the current drawing and all attached source drawings.

struct resbuf*  
ade_aliasgetlist();

Returns the list of drive aliases or NULL.

The following sample populates a resbuf with drive aliases using ade_aliasgetlist(). The contents of the resbuf are displayed and the resbuf is released as required.

```c
struct resbuf* pDriveAliasRb = ade_aliasgetlist();
if(pDriveAliasRb != NULL) {
    struct resbuf* rb = pDriveAliasRb;
    while(rb != NULL) {
        if (rb->restype == RTSTR) {
            acutPrintf(
                "\n\nThe \"%s\" alias contained the value:\n" , rb->resval.rstring);
            if (NULL != (rb = rb->rbnext)) {
                acutPrintf(
                    " \"%s\""
                    , rb->resval.rstring);
            }
        }
        rb = rb->rbnext;
    }
} else{
    acutPrintf(
        "\nNo alias could be determined.\n");
}
acutRelRb(pDriveAliasRb);
```
Assigns a new drive and path to a drive alias.

```c
int ade_aliasupdate(
    char* alias_name,
    char* path_name);
```

Returns RTNORM or an error code.

- **alias_name**: The name of the drive alias, which can be up to 31 characters. The name must be unique, contain no spaces, and start with an alphanumeric character.
- **path_name**: The new drive and path for the specified drive alias.
ade_altpclear

**Property Alteration Functions**

Clears the current property alteration definition.

```c
int ade_altpclear();
```

Returns RTNORM or an error code.

A property alteration definition is a list of one or more property alteration expressions. See [ade_altpdefine](#) for information about property alteration expressions.
ade_altpdefine

**Property Alteration Functions**

Creates a property alteration expression.

```c
ade_id ade_altpdefine(
    char* property,
    struct resbuf* value);
```

Returns a property alteration expression ID or **ADE_NULLID**.

- **property** : Property to alter. See the Alterable Properties table below.
- **value** : New value (type varies), or a range table expression that determines the new value. See Using a Range Table later in this topic.

**Note** If you use a range table expression instead of an explicit value argument, you must represent it in the resbuf as all one string.

You must release the resbuf.

**Using a Range Table**

The following code generates a property alteration expression that adds a yellow text object to each queried entity. The text object displays which layer the entity came from. Note how the value argument is prepared by using `ads_buildlist` to define and populate the required resbuf.

```c
char* propName = "textobject";
struct resbuf* pPropValueRb = ads_buildlist
    (RTLB,
    RTLB,
    RTSTR, "color", RTSTR, "yellow",
    RTDOTE,
    RTLB,
    RTSTR, "textvalue", RTSTR, ".Layer",
    RTDOTE,
    RTLE, 0
    );
```
ade_id propAltID = ade_altpdefine(propName, pPropValueRb);
if (ADE_NULLID != propAltID) {
    acutPrintf(
        "%The property alteration expression ID is: %.0lf"
        , propAltID);
} else {
    acutPrintf(
        "%No property alteration expression ID was set");
} acutRelRb(pPropValueRb);

A list of one or more property alteration expressions constitutes a property alteration definition. If there is a current property alteration definition when you create a property alteration expression, the new expression is added to it. When you execute a Draw query, each queried entity is altered in accord with the current property alteration definition.

The following table lists the alterable properties:

**Alterable Properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>blockname</td>
<td>Block name (RTSTR)</td>
</tr>
<tr>
<td>color</td>
<td>Color (RTSTR)</td>
</tr>
<tr>
<td>elevation</td>
<td>Z coordinate (RTPOINT) in the user coordinate system</td>
</tr>
<tr>
<td>height</td>
<td>Text height (RTREAL)</td>
</tr>
<tr>
<td>layer</td>
<td>Layer name (RTSTR)</td>
</tr>
<tr>
<td>linetype</td>
<td>Line type (RTSTR)</td>
</tr>
<tr>
<td>rotation</td>
<td>Rotation (RTREAL)</td>
</tr>
<tr>
<td>scale</td>
<td>Scaling factor (RTREAL). For example, 1.2 = 120%</td>
</tr>
<tr>
<td>style</td>
<td>Text style (RTSTR)</td>
</tr>
<tr>
<td>width</td>
<td>Line width (RTREAL)</td>
</tr>
<tr>
<td>textvalue</td>
<td>Text value (RTSTR)</td>
</tr>
<tr>
<td>thickness</td>
<td>Thickness (RTREAL)</td>
</tr>
<tr>
<td>hatch</td>
<td>List of dotted pairs that define the hatch properties. See Hatch properties below</td>
</tr>
</tbody>
</table>
To add a hatch pattern to each queried entity, as long as it is a closed polygon, specify "hatch" for the property argument. The value argument is then a list of dotted pairs. Each dotted pair is composed of a hatch property and a string value.

**Hatch Properties**

<table>
<thead>
<tr>
<th>property</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pattern</td>
<td>Hatch pattern name (RTSTR)</td>
</tr>
<tr>
<td>scale</td>
<td>Scaling factor (RTSTR). For example, &quot;1.2&quot; = 120%</td>
</tr>
<tr>
<td>rotation</td>
<td>Rotation of the hatch pattern (RTSTR)</td>
</tr>
<tr>
<td>layer</td>
<td>Name of the layer that contains the hatch pattern (RTSTR)</td>
</tr>
<tr>
<td>color</td>
<td>Hatch pattern color (RTSTR)</td>
</tr>
</tbody>
</table>

To create a text object for each queried entity, specify "textobject" for the property argument. The value argument is then a list of dotted pairs. Each dotted pair is composed of a text object property and a string value. The value element in the dotted pair can be an explicit value or a range table expression that determines a value.

**Text Object Properties**

<table>
<thead>
<tr>
<th>property</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>textvalue</td>
<td>Text to display (RTSTR)</td>
</tr>
<tr>
<td>height</td>
<td>Text height (RTSTR)</td>
</tr>
<tr>
<td>inspt</td>
<td>Point where text is inserted (expression as a RTSTR)</td>
</tr>
<tr>
<td>justify</td>
<td>Text alignment (RTSTR). For example, &quot;center&quot;.</td>
</tr>
<tr>
<td>style</td>
<td>Text style (RTSTR)</td>
</tr>
<tr>
<td>layer</td>
<td>Name of the layer on which the text object resides (RTSTR)</td>
</tr>
<tr>
<td>color</td>
<td>Text color (RTSTR)</td>
</tr>
<tr>
<td>rotation</td>
<td>Rotation of the text object (RTSTR)</td>
</tr>
</tbody>
</table>
ade_altpdelprop

Property Alteration Functions

Deletes a property alteration expression.

int
ade_altpdelprop(
    ade_id altp_id);

Returns RTNORM or an error code.

altp_id Property alteration expression ID.
ade_altpgetprop

Property Alteration Functions

Gets a property alteration expression.

```c
struct resbuf*
ade_altpgetprop(
    ade_id altp_id);
```

Returns a property alteration expression ID or NULL.

altp_id Property alteration expression ID.

You must release the resbuf.

See ade_altpdefine for information about property alteration expressions.

The following example creates an initial property alteration expression using ade_altpdefine(). This expression is based on a "Text Object" type and sets the color property to yellow and the text property to .layer. The property alteration expression ID returned by ade_altpdefine() is used by ade_altpgetprop() and automatically assumes that a "Text Object" type is going to be accessed. The property names and the corresponding values associated with the "Text Object" type are displayed, then the resbufs are released as required.

```c
char* propName = "textobject";
struct resbuf* pPropValueRb = acutBuildList(
    RTLB,
    RTSTR, "color",
    RTDOTE,
    RTSTR, "yellow",
    RTLE,
    RTLB,
    RTSTR, "textvalue",
    RTDOTE,
    RTSTR, ".Layer",
    RTLE, 0
);
ade_id propAltID = ade_altpdefine(propName, pPropValueRb);
```
if (ADE_NULLID != propAltID) {
    acutPrintf("The property alteration expression ID is: %.0lf", propAltID);
} else {
    acutPrintf("No property alteration expression ID was set");
}
acutRelRb(pPropValueRb);
struct resbuf* pPropAltExprRb = ade_altpgetprop(propAltID);
if (NULL != pPropAltExprRb) {
    acutPrintf("The "%s" property contains the following value pairs:" , pPropAltExprRb->resval.rstring);
    if (NULL != (pPropAltExprRb = pPropAltExprRb->rbnext)) {
        struct resbuf* rb = pPropAltExprRb;
        while(rb != NULL) {
            switch(rb->restype) {
            case RTSTR:
                acutPrintf("The "%s" property" , rb->resval.rstring, rb->resval.rstring);
                break;
            case RTDOTE:
                if (NULL != (rb = rb->rbnext))
                    acutPrintf(" contains the value " , rb->resval.rstring);
                break;
            default:
                break;
            }
            rb = rb->rbnext;
        }
    }
} else {
    acutPrintf("No property alteration expression information was retrieved");
}
acutRelRb(pPropAltExprRb);
ade_altplist

Property Alteration Functions

Lists the IDs of the current property alteration expressions.

```c
struct resbuf*
ade_altplist();
```

Returns a list of property alteration IDs or NULL.

This list of property alteration expressions returned by this function constitutes the current property alteration definition.

You must release the resbuf.

The following sample parses the resbuf returned by ade_altplist() and prints a list of property expression ID's in the current project. Then it releases the resbuf, as required.

```c
struct resbuf* pAltPropListRb = NULL;
pAltPropListRb = ade_altplist();
if (NULL != pAltPropListRb) {
    int nAltProp = 0;
    struct resbuf* rb = pAltPropListRb;
    while(rb != NULL) {
        ++nAltProp;
        acutPrintf( 
            "%nThe property alteration expression id %d is: %.0f"
            ,nAltProp, rb->resval.rreal);
        rb = rb->rbnext;
    }
} else {
    acutPrintf(
        "%nThere are no property alteration expressions in this project.");
}
acutRelRb(pAltPropListRb);
```
ade_altpsetprop

Property Alteration Functions

Modifies a property alteration expression.

```c
int
ade_altpsetprop(
    ade_id propId,
    char* property,
    struct resbuf* value);
```

Returns `RTNORM` or an error code.

- **altp_id**: Property alteration expression ID.
- **property**: Property to alter.
- **value**: The new value (type varies).

See `ade_altpdefine` for information about properties and values.

The following example creates an initial property alteration expression using `ade_altpdefine()`. This expression is based on a "Text Object" type and sets the color property to yellow and the text property to .layer. The property alteration expression ID returned by `ade_altpdefine()` is used by `ade_altpsetprop()` and automatically assumes that a "Text Object" type is going to be modified. The property param identifies a specific property of the "Text Object" type and the value `resbuf` contains the new value for that specific property. If the operation is successful, information is displayed about the modification and the `resbuf(s)` are released as required.

```c
char* pszPropertyType = "textobject";
struct resbuf* pPropValuePairRb = acutBuildList(
    RTLB,
    RTSTR, "color",
    RTDOTE,
    RTSTR, "yellow",
    RTLE,
    RTLB,
    RTSTR, "textvalue",
);
ade_id propAltID = ade_altpdefine(pszPropertyType, pPropValuePairRb);
char* pszPropertyName = "color";
struct resbuf* pNewPropValueRb = acutBuildList(RTSTR, "green", 0);
int resultCode = ade_altpsetprop(propAltID, pszPropertyName, pNewPropValueRb);
if (RTNORM == resultCode) {
    acutPrintf("\nThe \"%s\" property list contains the property \"%s\" which was assigned a new value of \"%s\"",
        pszPropertyType, pszPropertyName, pNewPropValueRb->resval.rstring);
}
else {
    acutPrintf("\nNo property modification was performed.");
}
acutRelRb(pPropValuePairRb);
acutRelRb(pNewPropValueRb);
**ade_dsattach**

**Drawing Set Functions**

Attaches a drawing to the current drawing.

*ade_id*

*ade_dsattach(*

  char* *dwgname*);

Returns the ID of the drawing attached or ADE_NULLID.

*dwgname*  The path alias and file name of the drawing to attach.

This function returns an ID even if the drawing does not exist. A system administrator can use this function to define a drawing set before the drawing files it references are created or installed. A drawing must exist before you can make it active.

```c
ade_aliasadd("ADSRX_SAMPLE", "C:\ADSRX\Drawings");
ade_id dwgId101 = ade_dsattach("ADSRX_SAMPLE:\101.dwg");
if (ADE_NULLID == dwgId101) {
    acutPrintf("\nThe specified drawing was not attached.");
} else {
    acutPrintf("\nThe specified drawing was successfully attached.");
}
```
**ade_dsdetach**

*Drawing Set Functions*

Detaches a drawing from the current drawing.

```c
int ade_dsdetach(
    ade_id dwg_id);
```

Returns **RTNORM** or an error code.

**dwg_id**  The drawing ID to detach.

The following sample detaches the drawing attached in the `ade_dsattach` sample:

```c
int resultCode = ade_dsdetach("ADSRX_SAMPLE:\101.dwg");
if (RTNORM == resultCode) {
    acutPrintf("\nThe specified drawing was detached.\n");
}
else {
    acutPrintf("\nThe specified drawing was not detached.\n");
}
```
ade_dsisnested

**Drawing Set Functions**

Checks if a drawing has nested drawings.

```c
int ade_dsisnested(
    ade_id dwg_id);
```

Returns **ADE_TRUE** if the drawing has drawings attached, or **ADE_FALSE**.

- **dwg_id** The drawing ID of the drawing to check.

If the drawing ID has the value **ADE_NULLID**, the current drawing is checked to see if it has drawings attached.

Verify that the drawing in question is active before calling `ade_dsisnested`. It is not possible to determine if an inactive drawing has nested drawings. If `dwg_id` is not specified or is **NULL**, the function checks the current drawing to see if it has drawings attached.

The following sample checks the current project for attached drawings using `ade_dslist()` and if they exist the attached drawing is checked for attached, (nested) drawings using `ade_dsisnested()`. Status messages are displayed based on the result of the checks and the `resbuf` is released as required.

```c
ade_boolean bAllNestedDwgs = ADE_FALSE;
struct resbuf* pAttachedDwgListRb = ade_dslist(
    ADE_NULLID,
    bAllNestedDwgs);

if (NULL != pAttachedDwgListRb) {
    ade_id dwgId = pAttachedDwgListRb->resval.rreal;
    int nNestedDwgs = ade_dsisnested(dwgId);
    if (nNestedDwgs) {
        acutPrintf(
            "\nThe current project has an attached drawing which has a drawing attached to it."
        );
    } else{
        acutPrintf(
```
"No nested drawings were found.

}
}
else{
    acutPrintf(
        "No drawings were attached to the current project.");
}
acutRelRb(pAttachedDwgListRb);
ade_dslist

**Drawing Set Functions**

Lists the drawings attached to a specified drawing.

```c
struct resbuf*
ade_dslist(
    ade_id dwg_id,
    ade_boolean nested);
```

Returns a list of drawing IDs or **NULL**.

- **dwg_id**  The drawing ID or **ADE_NULLID**.
- **nested**  Drawing is nested or not. Values can be: **ADE_TRUE** or **ADE_FALSE**.

If the **dwg_id** argument is **ADE_NULLID**, this function returns drawing IDs for the drawings attached to the current drawing.

If the **nested** argument is **ADE_TRUE**, the list includes drawings that are directly attached and all nested drawings at every level below them. If the argument is **ADE_FALSE**, the list includes only drawings that are directly attached.

You must release the **resbuf**.

The following sample parses the **resbuf** returned by **ade_dslist()** and prints a list of attached drawing ID's. Then it releases the **resbuf**, as required.

```c
struct resbuf* pAttachedDwsRb = NULL;
pAttachedDwsRb = ade_dslist(ADE_NULLID, 1);
if (NULL != pAttachedDwsRb) {
    int nDwg = 0;
    struct resbuf* rb = pAttachedDwsRb;
    while(rb != NULL) {
        ++nDwg;
        acutPrintf("\nAttached drawing # %d has the ID of: %.0lf", nDwg, rb->resval.rreal);
        rb = rb->rbnext;
    }
}
```
} 
{}
else {
    acutPrintf(
        "\nThere are no attached drawings in this project."
    );
}
acurRelRb(pAttachedDwgsRb)
ade_dsproplist

**Drawing Set Functions**

Lists all values found in the drawing set for a given drawing property.

```c
struct resbuf*
ade_dsproplist(
    char* property);
```

Returns a list of values or NULL.

**property**  
Drawing property. See Drawing Properties below.

The function searches all active source drawings and returns a list of the values it finds for the given drawing property.

The following table shows property names and return values.

### Drawing Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>object_type</code></td>
<td>AutoCAD object types (string)</td>
</tr>
<tr>
<td><code>blockname</code></td>
<td>Block names (string)</td>
</tr>
<tr>
<td><code>linetype</code></td>
<td>Line type names (string)</td>
</tr>
<tr>
<td><code>textstyle</code></td>
<td>Text style names (string)</td>
</tr>
<tr>
<td><code>attrib</code></td>
<td>Attribute name tags (string)</td>
</tr>
<tr>
<td><code>extents</code></td>
<td>Computed extents: the lower-left and upper-right points in the set of active source drawings. For example: ((2.20286 4.99866) (20.4689 12.3563))</td>
</tr>
<tr>
<td><code>group</code></td>
<td>Group names (string)</td>
</tr>
<tr>
<td><code>layer</code></td>
<td>Layer names (string)</td>
</tr>
<tr>
<td><code>lpn</code></td>
<td>Link templates (string). Note that link path names (LPNs) have been replaced by link templates in AutoCAD Map.</td>
</tr>
<tr>
<td><strong>objdata</strong></td>
<td>Names of object data tables. Table names can be up to 25 characters long (string). Must be unique, contain no spaces, and start with an alphanumeric character</td>
</tr>
<tr>
<td><strong>mlinestyle</strong></td>
<td>Mline style (string)</td>
</tr>
<tr>
<td><strong>feature</strong></td>
<td>Feature name (string)</td>
</tr>
<tr>
<td><strong>lineweight</strong></td>
<td>Line weight (string)</td>
</tr>
<tr>
<td><strong>plotstyle</strong></td>
<td>Plot style (string)</td>
</tr>
</tbody>
</table>

The following sample parses the `resbuf` returned by `ade_dsproplist()` and prints a list of object data tables. Then it releases the `resbuf`, as required.

```c
char* pszPropValue = "objdata";
struct resbuf* pDwgSetPropertyRb = ade_dsproplist(pszPropValue);
if(NULL != pDwgSetPropertyRb) {
    struct resbuf* rb = pDwgSetPropertyRb;
    while(rb != NULL) {
        acutPrintf("The current drawing set contains the following %s information: %s", pszPropValue, rb->resval.rstring);
        rb = rb->rbnext;
    }
} else {
    acutPrintf("No data could be retrieved.");
}
acutRelRb(pDwgSetPropertyRb);
```
activates a drawing.

int
ade_dwgactivate(ade_id dwg_id);

returns RTNORM or an error code.

dwg_id drawing ID.

you can attach a drawing that does not yet exist, but you cannot activate it. see ade_dsattach.

the following sample gets a list, (resbuf) containing drawing IDs of drawings directly attached to the current project, (no nested drawings) using ade_dslist(). If attached drawings exist, the list is processed and each drawing in the list is deactivated using ade_dwgdeactivate(). The resbuf is then released as required.

```c
struct resbuf* pAttachedDwgsRb = ade_dslist(ADE_NULLID, ADE_FALSE);
int resultCode = RTERROR;
if (NULL != pAttachedDwgsRb) {
    struct resbuf* rb = pAttachedDwgsRb;
    while(NULL != rb) {
        resultCode = ade_dwgdeactivate(rb->resval.rreal);
        rb = rb->rbnext;
    }
} else {
    acutPrintf("\nThere are no attached drawings in this project.\n");
}
acutRelRb(pAttachedDwgsRb);
```
ade_dwgactualpath

**Drawing Functions**

Returns the actual path of a drawing.

```c
char*
ade_dwgactualpath(
    ade_id dwg_id);
```

Returns the actual full path (without an alias) of the specified drawing or **NULL**. You must call the `free` function to deallocate the returned buffer.

- **dwg_id**  Drawing ID.
ade_dwgaliaspath

Drawing Functions

Returns the alias path of a drawing.

char*
ade_dwgaliaspath(
    const char* dwgPath);

Returns the drawing alias path or NULL. You must call the free function to deallocate the returned buffer.

dwg_path The actual path of the drawing.
Returns a list of attribute tags for the specified block name.

```
struct resbuf*
ade_dwgattriblist(
    ade_id dwg_id,
    char* block_name);
```

Returns a list of attribute tags or NULL.

**dwg_id**  The drawing ID of the attached drawing containing the "block_name".

**block_name**  The name of the block for which to get attribute tags.

This function returns a list of the attribute tags, given a block name from the specified drawing.

The following sample retrieves the ID of an attached drawing using `ade_dwggetid()` then parses the `resbuf` returned by `ade_dwgattriblist()` and prints a list of block attributes. Then it releases the `resbufs`, as required.

```
char* pszDwgPathName = "MYDWGS:\994049-2blk.dwg";
ade_id dwgId = ade_dwggetid(pszDwgPathName);
char* pszBlockName = "1BANK-TR";
struct resbuf* pBlkAttrbRb = ade_dwgattriblist(dwgId, pszBlockName);
if(pBlkAttrbRb != NULL) {
    struct resbuf* rb = pBlkAttrbRb;
    acutPrintf("A block named %s was found in %s and contains the following attributes: \n\n", 
            pszBlockName, pszDwgPathName);
    while(rb != NULL) {
        acutPrintf("\n%", rb->resval.rstring);
        rb = rb->rbnext;
    }
} else{
    acutPrintf(
```
"\n%s was not found in %s",
pszBlockName, pszDwgPathName);
}
acutRelRb(pBlkAttrbRb);
ade_dwgdeactivate

Drawing Functions

Deactivates a drawing.

int ade_dwgdeactivate(
    ade_id dwg_id);

Returns RTNORM or an error code.

dwg_id Drawing ID.

The following sample deactivates all the drawings in the drawing set.

```
struct resbuf* pAttachedDwgsRb = ade_dslist(ADE_NULLID, ADE_FALSE);
int resultCode = RTERROR;
if (NULL != pAttachedDwgsRb) {
    struct resbuf* rb = pAttachedDwgsRb;
    while(NULL != rb) {
        resultCode = ade_dwgdeactivate(rb->resval.rreal);
        rb = rb->rbnext;
    }
} else {
    acutPrintf("\nThere are no attached drawings in this project.");
}
acutRelRb(pAttachedDwgsRb);
```
ade_dwggetid

Drawing Functions

Gets the drawing ID of a drawing.

```c
ade_id
ade_dwggetid
    char* dwgpathname);
```

Returns a drawing ID or ADE_NULLID.

```
dwg_pathname       The path alias and drawing file name.
```

The following sample creates a drive alias using ade_aliasadd() then obtains a drawing id using ade_dwggetid(). Information pertaining to the results of the operation is then displayed. Note, The drawing in question must be attached to the current project.

```c
int resultCode = ade_aliasadd(  
    "ADSRX_SAMPLE",  
    "C:\My Documents\_MapAPI\ADSRX\Drawings");
char* pszDwgPathName = "ADSRX_SAMPLE:\ForQuery.dwg";
adde_id dwgId = ade_dwggetid(pszDwgPathName);
if (ADE_NULLID != dwgId) {
    acutPrintf(
        "%nThe specified drawing "%s" has a drawing ID of: %.0lf"
       , pszDwgPathName, dwgId);
} 
else {
    acutPrintf(
        "%nNo ID could be obtained for the specified drawing.");
}
```
ade_dwggetsetting

Drawing Functions

Gets a drawing setting value.

```c
struct resbuf*
ade_dwggetsetting(
    ade_id dwg_id,
    char* setting);
```

Returns the value of the given drawing setting or NULL.

- **dwg_id** Drawing ID.
- **setting** The drawing setting name. See Drawing Setting Names below

### Drawing Setting Names

<table>
<thead>
<tr>
<th>Setting Name</th>
<th>Return Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>dwgname</td>
<td>Drawing name; a path alias and file name, such as &quot;myfiles:\mydwg.dwg&quot;</td>
</tr>
<tr>
<td>dwgdesc</td>
<td>Drawing description.</td>
</tr>
<tr>
<td>t_scale</td>
<td>Simple transform scale. For example, 1.2 = 120%</td>
</tr>
<tr>
<td>t_rotate</td>
<td>Simple transform rotation direction. Value depends on the AutoCAD ANGDIR setting (real). Values: 0 = counterclockwise, 1 = clockwise</td>
</tr>
<tr>
<td>t_xoffset</td>
<td>Simple transform X offset.</td>
</tr>
<tr>
<td>t_yoffset</td>
<td>Simple transform Y offset.</td>
</tr>
<tr>
<td>t_apply</td>
<td>Flag value. Values: 1 = apply all simple transformations defined for the given drawing, 0 = do not apply transformations</td>
</tr>
<tr>
<td>saveback</td>
<td>Save back coordinates, a sequence of corner points, in this order: lower left, lower</td>
</tr>
</tbody>
</table>
The following sample gets the drawing ID from an attached drawing using `ade_dwggetid()` then parses the resbuf returned by `ade_dwggetsetting()` and prints the information relevant to the setting specified. Then it releases the resbuf, as required.

```c
ade_id dwgId;
char* pszDwgPathName = "MYDWGS:\994049-2blk.dwg";
dwgId = ade_dwggetid(pszDwgPathName);

struct resbuf* pDwgSettingRb;
char* pszDwgSetting = "dwgdesc";
pDwgSettingRb = ade_dwggetsetting(dwgId, pszDwgSetting);
while(pDwgSettingRb != NULL) {
    acutPrintf("\nThe %s information associated with attached drawing %s is: %s\n", pszDwgSetting, pszDwgPathName, pDwgSettingRb->resval.rstring);
    pDwgSettingRb = pDwgSettingRb->rbnext;
}
acutRelRb(pDwgSettingRb);
```
ade_dwghaslocks

Drawing Functions

Checks if a drawing has locked objects.

```c
int ade_dwghaslocks(
    ade_id dwg_id);
```

Returns ADE_TRUE or ADE_FALSE.

**dwg_id**  Drawing ID.
ade_dwgindex
Drawing Functions

Applies specified index operations to a drawing.

rtype
ade_dwgindex(
    ade_id dwgID);

Returns RTNORM or an error code.

dwgID The drawing ID for the drawing on which index operations will be performed.

When you call ade_dwgindex, indexes in the specified drawing are created or removed in accordance with the current index operation definition created by one or more calls to ade_dwgindexdef. If there is no index operation definition, ade_dwgindex has no effect on the drawing and returns ADE_NULLID.

The following sample, first clears any existing index definitions using ade_dwgindexdef() with an empty string, (""") for the indexType parameter. A location index and an object data index are then defined and executed against all active drawings in the current project using ade_dwgindex(). Resbufs are then released as required. For more samples, see Creating Index Operation Expressions.

    char* pszIndexName = "";
    struct resbuf* indexParams = acutBuildList(RTNIL, 0);
    int resultCode = ade_dwgindexdef(
        pszIndexName,// An empty string "" clears
        0,// remove index
        indexParams);

    char* pszPropIndexName = "Location";
    resultCode = ade_dwgindexdef(
        pszPropIndexName,
        1,
        indexParams);

    struct resbuf* pOdiIndexParamsRb = acutBuildList(
        RTLB,
char* pszOdIndexName = "ObjData";
resultCode = ade_dwgindexdef(
    pszOdIndexName,
    1,
    pOdIndexParamsRb);

struct resbuf* pAttachedDwgsRb = ade_dslist(ADE_NULLID, ADE_FALSE);
if (NULL != pAttachedDwgsRb) {
    struct resbuf* rb = pAttachedDwgsRb;
    while (NULL != rb) {
        resultCode = ade_dwgindex(rb->resval.rreal);
        rb = rb->rbnext;
    }
} else {
    acutPrintf("nThere are no attached drawings in this project.");
}
acutRelRb(indexParams);
acutRelRb(pAttachedDwgsRb);
Specifies which indexes are to be created or removed.

```c
int ade_dwgindexdef(
    struct resbuf* indexType,
    short indexOper,
    struct resbuf* indexParams);
```

Returns RTNORM or an error code.

- **indexType**: Type of index to be operated on: "Location", "Property", "EED", "SQLLINKS", "ObjData".
- **indexOper**: Index operator: 1 = create, 0 = remove.
- **indexParams**: Additional parameters required to specify which index is to be manipulated or NULL if none. Only the Object Data index supports additional parameters.

The list of indexes to be created or removed and associated information constitutes an index operation expression. A list of one or more index operation expressions constitutes an index operation definition. If there is a current index operation definition when you create an index operation expression, the new expression is added to it. When you call ade_dwgindex, indexes in the specified drawing are created or removed in accordance with the current index operation definition.

If you use "" (the empty string) for the `indexType` argument (in which case the arguments for the other two parameters do not matter so long as they are valid), the current index operation definition is cleared.

The `indexParams` argument is a list of tables and fields on which object data indexes will be created or removed (depending on the `indexOper` argument). The list consists of an object data table name followed by the fields to be processed. If this argument is NULL, all object data tables and fields will be processed.

The following sample gets the drawing ID from an attached drawing using `ade_dwgettid()` then parses the `resbuf` used by `ade_userlist()` and prints a status of the type of index created for the specified drawing. Then it releases the `resbuf`, as required. For more samples, see [Creating Index Operation Expressions](#).

```c
char* pszDwgPathName = "MYDWGS:\994049-2blk.dwg";
ade_id dwgId = ade_dwgettid (pszDwgPathName);
```
int indexOper = 1;
struct resbuf* pDwgIndexOnRb = acutBuildList(
    RTSTR,"Property",
    RTSTR,"Location",
    RTSTR,"EED",
    0;
)
struct resbuf* rb = pDwgIndexOnRb;
while(rb != NULL) {
    int returnCode = ade_dwgindexdef(rb->resval.rstring, indexOper, NULL);
    if (RTNORM == returnCode) {
        acutPrintf("The %s index has been successfully created for %s",
            rb->resval.rstring, pszDwgPathName);
    } else {
        acutPrintf("The index could not be created.");
    }
    rb = rb->rbnext;
}
acutRelRb(pDwgIndexOnRb);
**ade_dwgisactive**

*Drawing Functions*

Checks if a drawing is active.

```c
int ade_dwgisactive(
    ade_id dwg_id);
```

Returns **ADE_TRUE** or **ADE_FALSE**.

**dwg_id**  Drawing ID.

The function returns **ADE_TRUE** if the specified drawing is active. If the drawing is not active or the drawing ID is invalid, the function returns **ADE_FALSE**.
Checks if a drawing is directly attached to the current drawing.

```c
int ade_dwgistoplevel(ade_id dwg_id);
```

Returns **ADE_TRUE** if the drawing is attached directly to the current drawing, or **ADE_FALSE**.

*dwg_id* Drawing ID.
ade_dwgproplist

**Drawing Functions**

Lists all values found in a drawing for a given drawing property.

```
struct resbuf*
ade_dwgproplist(
    ade_id dwg_id,
    char* property);
```

Returns a *resbuf* list or NULL.

**dwg_id** Drawing ID.

**property** Property name. See Property Names below.

You must release the *resbuf*.

The function searches the given drawing and returns a list of the values it finds for the given drawing property.

**Property Names**

<table>
<thead>
<tr>
<th>Name</th>
<th>Return Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>object_type</td>
<td>AutoCAD object types.</td>
</tr>
<tr>
<td>blockname</td>
<td>Block names.</td>
</tr>
<tr>
<td>linetype</td>
<td>Line type names.</td>
</tr>
<tr>
<td>textstyle</td>
<td>Text style names.</td>
</tr>
<tr>
<td>attrib</td>
<td>Attribute tag names.</td>
</tr>
<tr>
<td>extents</td>
<td>Computed extents. The most lower-left point and the most upper-right point in the drawing For example: ((2.20286 4.99866) (20.4689 12.3563))</td>
</tr>
<tr>
<td>group</td>
<td>Group names.</td>
</tr>
<tr>
<td>layer</td>
<td>Layer names.</td>
</tr>
<tr>
<td>---------</td>
<td>------------------------------------------</td>
</tr>
<tr>
<td>lpn</td>
<td>Link templates. Note that link path names (LPNs) have been replaced by link templates in AutoCAD Map</td>
</tr>
<tr>
<td>objdata</td>
<td>Names of object data tables.</td>
</tr>
<tr>
<td>mlinestyle</td>
<td>Mline style.</td>
</tr>
<tr>
<td>feature</td>
<td>Feature name.</td>
</tr>
<tr>
<td>lineweight</td>
<td>Line weight.</td>
</tr>
<tr>
<td>plotstyle</td>
<td>Plot style.</td>
</tr>
</tbody>
</table>

The following sample gets a drawing ID from an attached drawing using `ade_dwggetid` then parses the `resbuf` returned by `ade_dwgproplist()` and prints a list of object data tables found in the attached drawing. Then it releases the `resbuf`, as required.

```c
char* pszDwgPathName = "MYDWGS:\SanFranMarina_BlkGrp_OD.dwg";
ade_id dwgId = ade_dwggetid(pszDwgPathName);
char* pszPropValue = "objdata";
struct resbuf* pDwgPropertyRb = ade_dwgproplist(dwgId, pszPropValue);
if(pDwgPropertyRb != NULL) {
    struct resbuf* rb = pDwgPropertyRb;
    while(rb != NULL) {
        acutPrintf("The current drawing set contains the following \%s information: \%s", pszPropValue, rb->resval.rstring);
        rb = rb->rbnext;
    }
} else{
    acutPrintf("\nNo information could be retrieved.");
}
acutRelRb(pDwgPropertyRb);
```
Displays a quick view of a drawing.

```
int ade_dwgquickview(ade_id dwg_id);
```

Returns RTNORM or an error code.

**dwg_id**  Drawing ID.

The following sample gets a list of drawings attached to the current project using `ade_dslist()`. Then displays quick views of all attached drawings and releases the `resbuf` as required.

```c
struct resbuf* pAttachedDwgsRb = ade_dslist(ADE_NULLID, ADE_FALSE);
int resultCode = RTERROR;
if (NULL != pAttachedDwgsRb) {
    struct resbuf* rb = pAttachedDwgsRb;
    while(NULL != rb) {
        resultCode = ade_dwgquickview(rb->resval.rreal);
        rb = rb->rbnext;
    }
} else {
    acutPrintf(
        "\nThere are no attached drawings in this project."\n    );
}
acutRelRb(pAttachedDwgsRb);
```
ade_dwgselectdlg

**Drawing Functions**

Displays the Select Drawings dialog box.

```c
struct resbuf*
ade_dwgselectdlg(
    void* pParent,
    const char* pCaption);
```

Returns a resbuf list or NULL.

- **pParent**
  A value defining the pointer to the Select Drawings dialog box parent window, which is expected to be represented by a CWnd object. 0 means that the parent window is not defined.

- **pCaption**
  Text that is shown before the current directory in the caption of the Select Drawings dialog box.

You must release the resbuf.

The following sample populates a resbuf based on the files selected using ade_dwgselectdlg(). The contents of the resbuf are displayed and then released as required.

```c
void* pParentCWnd = 0;
const char* pszDlgCaption = "Dialog Caption";
struct resbuf* pSelectedDwgsRb = ade_dwgselectdlg(pParentCWnd, pszDlgCaption);
if (NULL != pSelectedDwgsRb) {
    acutPrintf("The following files were selected: ");
    struct resbuf* rb = pSelectedDwgsRb;
    while(NULL != rb) {
        acutPrintf("\n\n\"%s\"", rb->resval.rstring);
        rb = rb->rbnext;
    }
} else {
```
acutPrintf("\nNo files were selected.");
}
acutRelRb(pSelectedDwgsRb);
ade_dwgsetof

**Drawing Functions**

Identifies the drawings to which a given drawing is attached.

```c
struct resbuf*
ade_dwgsetof(
    ade_id dwg_id);
```

Returns a *resbuf* list or NULL.

**dwg_id**  
Drawing ID.

You must release the *resbuf*.

You cannot use this function to check if a drawing is attached to the current drawing. Use *ade_dwgistoplevel* instead. If a drawing is attached to both the current drawing and to other drawings, this function returns a list of the IDs of the other drawings only.

The following sample returns the project name to which a specific drawing is attached. *Ade_dwggetid()* is used to obtain the drawing id of a file which is part of a drawing set, (BestRoute.dwg). A *resbuf* is populated with the name of the project which BestRoute.dwg is attached to using *ade_dwgsetof()*.

If the operation is successful, (the resbuf is not NULL) then the contents of that *resbuf* are displayed. The *resbuf* is then released as required.

```c
char* pszDwgPathName = "ADSRX_SAMPLE:\BestRoute.dwg";
ade_id dwgId = ade_dwggetid(pszDwgPathName);
struct resbuf* pDwgAttachedToRb = ade_dwgsetof(dwgId);
if (NULL != pDwgAttachedToRb)
{
    struct resbuf* rb = pDwgAttachedToRb;
    while(rb != NULL)
    {
        char* pszDwgPath = ade_dwgactualpath(rb->resval.rreal);
        acutPrintf("\
            The specified drawing "\n            is attached to: \n            , pszDwgPathName, pszDwgPath);
        rb = rb->rbnext;
```
} }
else {
    acutPrintf(
        "\nThe specified drawing is attached to no project."\n    );
}
acutRelRb(pDwgAttachedToRb);
ade_dwgsetsetting

Drawing Functions

Sets a drawing setting value.

int
ade_dwgsetsetting(
    ade_id dwg_id,
    struct resbuf* proplist);

Returns ADE_TRUE if the drawing is active or ADE_FALSE.

dwg_id  Drawing ID.
proplist List composed of a setting name and a value. See Setting Names below.

Setting Names

<table>
<thead>
<tr>
<th>Setting name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>dwgname</td>
<td>Drawing name; a full path name, such as c:\drawings\mydwg.dwg.</td>
</tr>
<tr>
<td>dwgdesc</td>
<td>Drawing description.</td>
</tr>
<tr>
<td>t_scale</td>
<td>Simple transform scale. For example, 1.2 = 120%</td>
</tr>
<tr>
<td>t_rotate</td>
<td>Simple transform rotation; rotation direction depends on the AutoCAD ANGDIR setting.</td>
</tr>
<tr>
<td>t_xoffset</td>
<td>Simple transform X offset.</td>
</tr>
<tr>
<td>t_yoffset</td>
<td>Simple transform Y offset.</td>
</tr>
<tr>
<td>t_apply</td>
<td>Flag value. Values can be: 1 = apply all simple transformations defined for the given drawing 0 = do not apply transformations</td>
</tr>
<tr>
<td>saveback</td>
<td>Save back coordinates, a sequence of corner points, in this order: lower left, lower right, upper right, upper left, separated by &quot;.&quot;</td>
</tr>
</tbody>
</table>
The following sample gets a drawing ID from an attached drawing using `ade_dwggetid()`. This ID and the `resbuf` containing the setting name/value pair are used by `ade_dwgsetsetting()` which returns a status code whose associated message is displayed. Then it releases the `resbuf`, as required.

```c
ade_id dwgId = ade_dwggetid(pszDwgPathName);
struct resbuf* pDwgSettingsRb = ads_buildlist
    (RTLB,
     RTSTR, "dwgdesc",
     RTSTR, "This is a drawing description",
     RTDOTE, 0);
returnCode = ade_dwgsetsetting(dwgId, pDwgSettingsRb);
if (RTNORM == returnCode) {
    acutPrintf("\nThe value has been successfully set for \%s",
               pszDwgPathName);
} else {
    acutPrintf("\nThe value could not be set.");
}
acutRelRb(pDwgSettingsRb);
```
ade_dwgunlock

**Drawing Functions**

Removes all object locks from a drawing.

```c
int ade_dwgunlock(
    ade_id dwg_id);
```

Returns RTNORM or an error code.

**Parameters**

- **dwg_id**  
  Drawing ID.

Using this function requires superuser privileges.
ade_dwgzoomextents

*Drawing Functions*

Zooms to the extents of the active drawings.

```c
int ade_dwgzoomextents();
```

Returns **RTNORM** or **RERROR**.
ade_editdefcen

Object Editing Functions

Defines a new label point for an object.

```c
int ade_editdefcen(
    ads_name ename,
    ads_point pt);
```

Returns RTNORM or an error code.

ename An AutoCAD entity name.

pt The label point, a list of values defined in 2D or 3D point.

Use this function with property alteration if the current label point is not suitable for the text object you are adding.

The following sample modifies the label point for every entity in the current drawing.

```c
ads_name ename;
ads_point newLabelPt;
double newLabelPtOffset = 1.25;
struct resbuf* pLabelPtValRb = NULL;
ads_name selectionSet;
acedSSGet("x", NULL, NULL, NULL, selectionSet);
long ssLength;
acedSSLen( selectionSet, &ssLength );
for(int i = 0; i < ssLength; ++i) {
    if(acedSSName(selectionSet, i, ename) == RTNORM ) {
        pLabelPtValRb = ade_expreval(
            ename,
            ".labelpt",
            "point");
        if (NULL != pLabelPtValRb) {
            newLabelPt[X] = (pLabelPtValRb->resval.rpoint[0] - newLabelPtOffset);
```
newLabelPt[Y] = (pLabelPtValRb->resval.rpoint[1] - newLabelPtOffset);
int resultCode = ade_editdefcen(
    ename,
    newLabelPt);
}
else {
    acutPrintf(
        "\nThe label point could not be determined.\n");
}
}
else {
    acutPrintf(
        "\nCould not get the entity.\n");
}
}
acutRelRb(pLabelPtValRb);
acedSSFree(selectionSet);
ade_editlockederased

Object Editing Functions

Gets the objects in the save set that have been erased.

```c
int ade_editlockederased(
        ads_name sel_set);
```

Returns RTNORM or an error code

```c
sel_set       The name of the selection set.
```
ade_editislocked

Object Editing Functions

Gets lock information about an object if it is locked.

```c
struct resbuf*
ade_editislocked(
    ads_name ename);
```

Returns a `resbuf` list of lock information about the specified locked object, or, if it is not locked, `NULL`.

ename AutoCAD entity name.

You must release the `resbuf`.

The list of lock information returned by this function contains the following, in order:

- Login name of the user who locked the object.
- Name and path of the drawing that contains the object.
- Date the object was locked.
- Time the object was locked.
- Name and path of the current drawing.

The following sample create a selection set of locked entities in a save back set using `ade_editlocked()`. A `resbuf` is then populated with edit lock information for each of the entities in the selection set using `ade_editislocked()`. That information is displayed and the `resbuf` is released as required.

```c
ads_name selectionSet;
int nResultCode = ade_editlocked(selectionSet);
long ssLength;
acedSSLen( selectionSet, &ssLength );
struct resbuf* pLockedEntitiesRb = NULL;
for( int i = 0; i < ssLength; ++i )
{
    ads_name ename;
    if( acedSSName( selectionSet, i, ename ) == RTNORM )
    {
```
pLockedEntitiesRb = ade_editislocked(ename);
if (NULL != pLockedEntitiesRb) {
    acutPrintf(
        "\nThe locked entity contains the following information:");
    struct resbuf* rb = pLockedEntitiesRb;
    while(rb != NULL) {
        if (rb->restype == RTSTR) {
            acutPrintf(
                "\n\t" , rb->resval.rstring);
        }
        rb = rb->rbnext;
    }
}
acutRelRb(pLockedEntitiesRb);

You can change the format of the date and time strings through options in the International dialog box in the Microsoft Windows Control Panel.
ade_editlocked

**Object Editing Functions**

Gets the objects in the save set that have been modified or are new.

```c
int ade_editlocked(
    ads_name sel_set);
```

Returns **RTNORM** or an error code.

- **sel_set**  The name of the selection set.
ade_editlockobjs

Object Editing Functions

Locks a set of objects and adds them to the save set.

```c
ads_real
ade_editlockobjs
    ads_name sel_set;
```

Returns the number of objects locked or ADE_REALFAIL.

```c
sel_set        The name of the selection set.
```

The function locks the objects contained in the designated selection set. Locking these objects adds them to the save set.

It is a good idea to compare the number of objects locked with the number of objects in the designated selection set. If the number locked is less than the number in the selection set, an error occurred in the locking process, and you should check the error stack.

The following sample creates a filtered selection set, adds its objects to a save set using ade_editlockobjs(), checks the result and displays information based on those results. The resbuf and selection set is then released as required.

```c
struct resbuf* pFilteredEntitySelectionRb = acutBuildList(
    RTDXF0, "LWPOLYLINE",
    8, "UtilityNetwork-Electric",
    0);

ads_name ssObjsForSaveSet;
acedSSGet("X", NULL, NULL, pFilteredEntitySelectionRb, ssObjsForSaveSet);
long filteredEntitySelectionLength = 0;
acedSSLLength(ssObjsForSaveSet, &filteredEntitySelectionLength);

ads_real objsAddedToSaveSet = ade_editlockobjs(ssObjsForSaveSet);
if (filteredEntitySelectionLength == objsAddedToSaveSet) {
    acutPrintf(
```
"The number of selected entities, (%d)"
"matches the number added to the save set."
, filteredEntitySelectionLength);
}
else {
    acutPrintf(
        "\nThe number of selected entities, (%d)"
        "does not match the number added to the save set, (%d)."
        , filteredEntitySelectionLength, objsAddedToSaveSet);
}
acutRelRb(pFilteredEntitySelectionRb);
int resultCode = acedSSFree(ssObjsForSaveSet);
ade_editnew

**Object Editing Functions**

Gets the objects in the saved set that are new.

```c
int ade_editnew(
    ads_name sel_set);
```

Returns **RTNORM** or an error code.

`sel_set`  The name of the selection set.
Unlocks a set of objects and removes them from the save set.

```c
ads_real
ade_editunlockobjs
   ads_name sel_set);
```

Returns the number of objects unlocked. If for some reason the selection set is freed prior to the call to `ade_editunlockobjs()`, the number of number of objects unlocked is returned as 0.0.

```c
sel_set The name of the selection set.
```

The function unlocks the objects in the specified selection set. If the selection set is `ADE_REALFAIL`, the function unlocks all erased objects. Unlocking objects removes them from the save set.

The following sample is a continuation of the `ade_editlockobjs` sample.

```c
ads_real objsRemovedFromSaveSet = ade_editunlockobjs(ssObjsForSaveSet);
if (filteredEntitySelectionLength == objsRemovedFromSaveSet) {
    acutPrintf( /* The number of selected entities, (%d) */
            " matches the number removed from the save set."
            , filteredEntitySelectionLength);
} else {
    acutPrintf( /* The number of selected entities, (%d) */
            " does not match the number removed from the save set, (%d)."
            , filteredEntitySelectionLength, objsRemovedFromSaveSet);
}
int resultCode = acedSSFree(ssObjsForSaveSet);
```
**ade_entsetlocation**

**Other Functions**

Sets a new text-label point for a drawing object.

```c
int ade_entsetlocation(
    ads_name ename,
    ads_point pt);
```

Returns **RTNORM** or an error code.

- `ename` — An AutoCAD entity name.
- `pt` — The new text-label point, a 2D or 3D point.

An drawing object's text-label point is the starting position for text added during a query property alteration. By default, the centroid of the object is the text-label point.
ade_errclear

Error Message Functions

Clears the error stack.

int
ade_errclear()

Returns RTNORM or an error code.
ade_errcode

Error Message Functions

Gets the error code for a given error in the stack.

```c
int ade_errcode(
    int err_index);
```

Returns an error code or NULL.

- **err_index** The position of the error in the stack, where 0 = first error

For a list of error codes, see Error Codes.
ade_errgetlevel

Error Message Functions

Gets the system error level.

```c
int ade_errgetlevel();
```

Returns an error level or an error code.
ade_errmsg

Error Message Functions

Gets the error message for a given error in the stack.

char*
ade_errmsg(
    int err_index);

Returns an error message or NULL.

err_index    Position of the error in the stack, where
            0 = first error.
ade_errpush

Error Message Functions

Pushes an error to the stack.

```c
int ade_errpush(
    int err_code,
    char* level,
    char* message);
```

Returns RTNORM or an error code.

- **err_code**: Error code (details below).
- **level**: Error level. Values can be: "warning", "error" (default), or the empty string. If the empty string, the error level is "error".
- **message**: Error message.

If your application will use custom error codes, define a range for them that does not conflict with any range used for AutoCAD Map error codes. To specify a general error, let the `err_code` argument be 1 (kAdeErr).

The valid `level` values, "warning", and "error", correspond respectively to the following error types:

<table>
<thead>
<tr>
<th>1</th>
<th>kAdeWarning</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>kAdeError</td>
</tr>
</tbody>
</table>

For a list of all error types, including the two that are valid `level` values, see Error Types.

The following example pushes an error to the stack.

```c
ade_errpush( 1, "error", "message text" );
```
ade_errpushstatement

Error Message Functions

Pushes a faulty SQL statement to the error stack.

```c
int ade_errpushstatement(
    char* statement,
    int position);
```

Returns RTNORM or an error code.

- **statement**: Faulty statement that caused the error.
- **position**: Starting position of the error in the faulty statement. Position 1 corresponds to the first character.

This function is designed to add diagnostic information to an error you have just pushed. It is associated with the latest error in the stack only. A call to `ade_errpushstatement` makes sense only if a call to `ade_errpush` immediately precedes it.
Returns the number of errors in the stack.

```c
int ade_errqty();
```

Returns an error count or `ADEERROR`. 
Sets the system error level.

```c
int ade_errsetlevel
  int level);
```

Returns RTNORM or an error code.

- **level**: Error level. Values can be: 0, 1, or 2

See [ade_errgetlevel](#) for details.
<table>
<thead>
<tr>
<th>Error Message Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>ade_errshowdlg</td>
</tr>
</tbody>
</table>

Displays the Map Messages dialog box.

```c
int ade_errshowdlg();
```

Returns RTNORM or an error code.
ade_errstatement
Error Message Functions

Gets the faulty SQL statement for a given error in the stack.

```c
struct resbuf*
ade_errstatement(
    int err_index);
```

Returns a statement (string) and a starting position (short), or NULL

- `err_index` Position of the error in the stack
  - `0` = first error

The SQL statement and error position are returned in the following format.

```
(faulty_statement err_pos)
```

The `faulty_statement` string quotes the faulty SQL statement that caused the error. The `err_pos` value identifies the starting position of the error in the faulty statement. Position 1 is the first character of the statement.

The expression `(ade_errstatement 2)`, which references the third error in the stack, could return

```
("xxx" 6)
```

where "xxx" is the faulty statement and 6 tells you that the trouble begins at the sixth character.

You must release the `resbuf`.

The following sample concludes by printing a faulty SQL statement to the screen.

The first step is to define a SQL query containing a faulty SQL statement which is in a `resbuf`. `Ade_qrydefine()` is then called with all required parameters. The error stack is cleared using `ade_errclear()` and the query is executed using `ade_qryexecute()`. The error stack is checked for any new errors using `ade_errqty()`, and if they exist a `resbuf` is populated with the SQL Select statement and the position that the syntactic error occurred. The contents of the `resbuf` are displayed and all `resbufs` are released as required.

```c
char* pszJoinOperator = ""; // none
char* pszBgnCondGrouping = ""; // none
```
char* pszNotOperator = "";  // none
char* pszCondType = "SQL";
char* pszEndCondGrouping = "";  // none

struct resbuf* pQueryConditionRb = acutBuildList(
    RTLB,
    RTSTR, "BlockGroup",
    RTSTR, "AMERI_ES == 0",
    RTLE,
    0);
ade_id queryId = ade_qrydefine(
    pszJoinOperator,
    pszBgnCondGrouping,
    pszNotOperator,
    pszCondType,
    pQueryConditionRb,
    pszEndCondGrouping);

int returnCode = ade_errclear();
ads_real queriedObjs = ade_qryexecute();
int errorStkQnty = ade_errqty();
if (errorStkQnty > 0)
{
    struct resbuf* pErrStatementRb = ade_errstatement(0);
    if (NULL != pErrStatementRb){
        struct resbuf* rb = pErrStatementRb;
        while(NULL != rb) {
            if (rb->restype == RTSTR) {
                if (_tcslen(rb->resval.rstring) != 0)
                {
                    acutPrintf("\n\nThe faulty SQL statement is:
\n\t" , rb->resval.rstring);
                    acutPrintf("\n\nThe syntax error has occurred at character: %d"
, rb->rbnext->resval.rint);
                }
            }
            rb = rb->rbnext;
        }
    }
    else {
        acutPrintf("\nNo error statements were returned.");
    }
}
acutRelRb(pErrStatementRb);

else
{
    acutPrintf("\nNo errors were encountered.");
}
acutRelRb(pQueryConditionRb);
ade_errtype

Error Message Functions

Gets the type of a given error in the stack.

```c
int ade_errtype(
    int err_index);
```

Returns an error type or ADEERROR.

- **err_index**  
  Position of the error in the stack  
  0 = first error

For a list of error types, see Error Types.
ade_expreval

Expression Evaluation Functions

Evaluates an expression.

```c
struct resbuf*
ade_expreval(
    ads_name ename,
    char* expr,
    char* type);
```

Returns the value of the expression or NULL.

ename          Optional drawing object name. Required if the expression uses object properties or data.
expr           Expression to evaluate.
return_type    Expected return type: "short", "long", "real", "string", or "point".

The following sample displays the .Length property of a pline entity which has been converted to Meters.

A selection set of entities from the current drawing is created using acedSSGet(). Each entity is passed to ade_expreval() along with any additional parameters. The resbuf returned by ade_expreval() contains the converted result which is then displayed. Then it releases the resbuf, as required.

```c
ads_name ename;
ads_name selectionSet;
char* pszExpr = "(* .LENGTH .3048)";
char* pszReturnType = "real";
struct resbuf* pEntLengthToM = NULL;
acedSSGet("x", NULL, NULL, NULL, &selectionSet);
long ssLength;
acedSSLenghth( selectionSet, &ssLength; );
for( int i = 0; i < ssLength; ++i )
{
    if( acedSSName(selectionSet, i, ename) == RTNORM )
    {
        pEntLengthToM = ade_expreval(enname, pszExpr, pszReturnType);
    }
}```
if (pEntLengthToM != NULL) {
    struct resbuf* rb = pEntLengthToM;
    while (rb != NULL) {
        acutPrintf(
            "%nThe current entities .LENGTH value converted to Meters is: %.4lf"
            , rb->resval.rreal);
        rb = rb->rbnext;
    }
}
acutRelRb(pEntLengthToM);
ade_keycolumnlist

SQL Environment Functions

Returns a list of the key column names for the specified link template.

```c
struct resbuf*
ade_keycolumnlist(
    char* linktemplate);
```

Returns a list of key column names or NULL.

`linktemplate` Link template.

For more information about link templates and using SQL, see the AutoCAD online documentation.

The following sample populates a `resbuf` with key column names using `ade_keycolumnlist()`. The contents of the `resbuf` are displayed and the `resbuf` is released as required.

```c
char* pszLinkTemplate = "BlockGroup";
struct resbuf* pKeyColumnListRb = ade_keycolumnlist(pszLinkTemplate);
if (NULL != pKeyColumnListRb) {
    struct resbuf* rb = pKeyColumnListRb;
    while(rb != NULL) {
        acutPrintf("The Key column name(s) are: %s", rb->resval.rstring);
        rb = rb->rbnext;
    }
} else {
    acutPrintf("There are no specified link templates in this project.");
}
acutRelRb(pKeyColumnListRb);
```
ade_odaddfield

Object Data Functions

Adds fields to a table.

```c
int ade_odaddfield(
    char* tabname,
    struct resbuf* fieldlist);
```

Returns RTNORM or an error code.

- **tabname** The table name, which can be up to 25 characters long. It must be unique, contain no spaces, and start with an alphanumeric character.
- **fieldlist** The list of fields to add; a sequence of field definitions.

A sequence of field definitions is introduced by the string "columns". Each field definition is a list of a-lists, and each a-list consists of a field property and a value, as follows:

<table>
<thead>
<tr>
<th>Field property</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>colname</td>
<td>Field name (RTSTR) can be up to 31 characters long. Must be unique, contain no spaces, and start with an alphanumeric character.</td>
</tr>
<tr>
<td>coldesc</td>
<td>Field description (RTSTR)</td>
</tr>
<tr>
<td>coltype</td>
<td>Field data type</td>
</tr>
<tr>
<td>defaultval</td>
<td>Default field value</td>
</tr>
</tbody>
</table>

The function adds fields to the table and to each of its records. In each record, the new fields are assigned default values in accord with their field definitions. The function has no effect on existing fields. In other words, the function adds fields to each set of object data defined by the table and attached to an object.

You must release the resbuf.

For an example, see Adding Fields to a Table.

The following sample creates a resbuf containing specifications for a new field, "field1" in the existing
object data table, "table1". This *resbuf* is passed to *ade_odaddfield()* which returns a result code indicating the success of the operation. Then it releases the *resbuf*, as required.

```c
struct resbuf* pAddFieldsRb = NULL;
char* pszOdTable = "table1";
pAddFieldsRb = acutBuildList(
    RTSTR, "columns",
    RTLB,
    RTSTR, "colname", RTSTR, "field1",
    RTDOTE,
    RTLB,
    RTSTR, "coldesc", RTSTR, "Field 1 Description",
    RTDOTE,
    RTLB,
    RTSTR, "coltype", RTSTR, "character",
    RTDOTE,
    RTLB,
    RTSTR, "defaultval", RTSTR, "Default Value",
    RTDOTE,
    RTLB,
    // Define more fields as needed
    0);
int nResultCode = ade_odaddfield(pszOdTable, pAddFieldsRb);
acutRelRb(pAddFieldsRb);
if (RTNORM == nResultCode) {
    acutPrintf(
        "\nThe specified field has been successfully added.");
} else {
    acutPrintf(
        "\nThe specified field has not been added.");
}
acutRelRb(pAddFieldsRb);
```
Attaches data to an object.

```
int ade_odaddrecord(
    ads_name ename,
    char* table);
```

Returns RTNORM or an error code.

**ename**  
An AutoCAD object name.

**table**  
The table name, which can be up to 25 characters long. It must be unique, contain no spaces, and start with an alphanumeric character.

Attaching data to an object is also called attaching a table to an object. This function attaches a new record in a specific table to a specific object. Typically, a record contains information about whatever it is that the object represents. For example, if a line in a drawing represents a section of pipe in a water system, an attached record could contain information about that section.

When a new record is attached, its fields contain default values that correspond to their field definitions. To get a field value, use `ade_odgetfield`; to change it, use `ade_odsetfield`. Field definitions are included in the table definition. See `ade_oddefinetab` for information about table definitions.

You can attach more than one record to the same object with additional calls to `ade_odaddrecord`. The additional records can be members of the same or different tables. If an object has only one record from a given table, the number of that record is 0. If you attach a second record from the same table, the number of that record is 1, and so on. Use `ade_odrecordqty` to find how many records of a given table are attached.

For example, if a section of water pipe is inspected at intervals, you could attach a number of records of the WATER INSPECTION table to the same line in the WATER drawing, and each record could contain the result of a different inspection.

The following sample creates a selection set of all entities in the current drawing using `acedSSGet()`. Each entity is then passed to `ade_odaddrecord()` with all required parameters. The result code returned by `ade_odaddrecord()` is displayed. Then it releases the `resbuf`, as required.

```
ads_name selectionSet;
```
ads_name ename;
char* pszOdTable = "table1";
acedSSGet("x", NULL, NULL, NULL, selectionSet);
long ssLength;
acedSSLengeth( selectionSet, &ssLength );
for( int i = 0; i < ssLength; ++i )
{
    if( acedSSName(selectionSet, i, ename) == RTNORM )
    {
        int nResultCode = ade_odaddrecord(ename, pszOdTable);

        if (RTNORM == nResultCode) {
            acutPrintf("\nThe specified record has been added.");
        }
        else {
            acutPrintf("\nThe specified record has not been added.");
        }
    }
}
Attaches a new record to an object.

```c
int ade_odattachrecord(
    ads_name ename,
    ade_id rec_id);
```

Returns RTNORM or an error code.

- **ename**  An AutoCAD object name.
- **rec_id** Record ID returned by `ade_odnewrecord`. 
ade_oddefinetab

Object Data Functions

Creates an object data table.

```c
int ade_oddefinetab(
    struct resbuf* tab_defn);
```

Returns RTNORM or an error code

- **tab_defn**: List of table elements: the table name, the table description, and a sequence field definitions.

The following sample creates a resbuf containing table definition parameters. ade_oddefinetab() uses this resbuf and returns a result code indicating the success of the operation. Then it releases the resbuf, as required.

```c
struct resbuf* pOdTableDefRb = acutBuildList(
    RTLB,
    RTSTR, "tablename", RTSTR, "NEWTABLE",
    RTDOTE,
    RTLB,
    RTSTR, "tabledesc", RTSTR, "New Sample Table",
    RTDOTE,
    RTLB,
    RTSTR, "columns",
    RTLB,
    RTLB,
    RTSTR, "colname", RTSTR, "FIELD1",
    RTDOTE,
    RTLB,
    RTSTR, "coldesc", RTSTR, "Field 1 Description",
    RTDOTE,
    RTLB,
    RTSTR, "coltype", RTSTR, "character",
    RTDOTE,
);```
RTLB,
    RTSTR, "defaultval", RTSTR, "Default Value",
    RTDOTE,
    RTLE,
    // Define more fields as needed
    RTLE,
    0);
int nResultCode = ade_oddefinetab(pOdTableDefRb);
if (RTNORM == nResultCode) {
    acutPrintf(
        "\nThe specified table has been successfully created.");
} else {
    acutPrintf(
        "\nThe specified table has not been created.");
}
acutRelRb(pOdTableDefRb);
ade_oddeletefield

Delete fields from a table.

int
ade_oddeletefield(
    char* tabname,
    struct resbuf* fieldList);

Returns RTNORM or an error code.

tabname
  The table name, which can be up to 25 characters long. It must be unique, contain no
  spaces, and start with an alphanumeric character.

fieldlist
  A list of field names.

The function deletes the fields from the table and from each of its records. The data contained in these
fields is also deleted. In other words, it deletes the fields and their data from each set of object data defined by the
table and attached to an object.

Note The fieldlist argument for ade_oddeletefield is a list of field names only. In the companion functions,
ade_odaddfield and ade_odmodifyfield, it is a list of field definitions.

The ade_oddeletefield function affects all active drawings in the drawing set. There should not be any
queried objects for this operation.

Note This function will not operate unless your end user has superuser privileges.

The following sample creates a resbuf containing the fields to delete. ade_oddeletefield() uses this resbuf
and returns a result code indicating a successful operation. Then it releases the resbuf, as required.

```
struct resbuf* pDeleteFieldsRb = NULL;
char* pszOdTable = "table1";
pDeleteFieldsRb = acutBuildList(
    RTSTR, "field1",
    RTSTR, "field2",
    RTSTR, "field3",
    0);
int nResultCode = ade_oddeletefield(pszOdTable, pDeleteFieldsRb);
```
if (RTNORM == nResultCode) {
    acutPrintf("\nThe fields specified have been successfully deleted.");
} else {
    acutPrintf("\nThe fields specified have not been deleted.");
}
acutRelRb(pDeleteFieldsRb);
ade_oddeletetab

Object Data Functions

Deletes a table.

```c
int ade_oddeletetab(
    char* tabname);
```

Returns RTNORM or an error code.

- **tabname** The table name, which can be up to 25 characters long. It must be unique, contain no spaces, and start with an alphanumeric character.

The function deletes a table and all of its records. It deletes every set of object data defined by the table and attached to an object, as well as the data contained in the records.

The `ade_oddeletetab` function affects all active drawings in the drawing set. There should not be any queried objects for this operation.

**Note** This function will not operate unless your end user has superuser privileges.
**ade_oddelrecord**

*Object Data Functions*

Deletes a record.

```c
int ade_oddelrecord(
    ads_name ename,
    char* table,
    int recnum);
```

Returns RTNORM or an error code.

- **ename** The AutoCAD entity name of the object to which the record is attached.
- **table** The name of the table to which the record belongs, up to 25 characters long. Must be unique, contain no spaces, and start with a character.
- **recnum** The record number. The number of the first record is 0.

The function deletes the record from the object. It deletes the set of object data defined by the table and attached to the object. This deletes the record from the table as well as the data contained in the record.

The record number is necessary because more than one record from the same table can be attached to an object. Use `ade_odrecordqty` to find how many records of a given table are attached.
**ade_odfreerec**

**Object Data Functions**

Frees the memory claimed in defining a new record.

```c
int ade_odfreerec(
    ade_id rec_id);
```

Returns RTNORM or an error code.

**rec_id**   The record ID returned by ade_odnewrecord.

**Warning** You must release a new record when you are finished with it.
ade_odgetfield

Object Data Functions

Gets a field value.

```c
struct resbuf*
ade_odgetfield(
    ads_name ename,
    char* table,
    char* field,
    int recnum);
```

Returns a field value or NULL.

- `ename`: AutoCAD object name.
- `table`: Table name can be up to 25 characters long. Must be unique, contain no spaces, and start with an alphanumeric character.
- `field`: Field name can be up to 31 characters long. Must be unique, contain no spaces, and start with an alphanumeric character.
- `recnum`: Record number. The number of the first record is 0.

To identify a unique record, you need to specify the table to which it belongs, the object to which it is attached, and its record number. The record number is necessary because more than one record from the same table can be attached to an object. For more information about records and record numbers, see `ade_odaddrecord`.

The field value returned can be one of four data types: integer, character, point, or real.

The following sample creates a selection set of all entities in the current drawing using `acedSSGet()`. Each entity in the selection set is passed to `ade_odgetfield()` with all required parameters, the results are stored in a `resbuf` which is displayed. Then it releases the `resbuf`, as required.

```c
struct resbuf* pGetFieldsRb = NULL;
ads_name selectionSet;
ads_name ename;
char* pszOdTable = "table1";
char* pszOdField = "field1";
```
int recnum = 0;
acedSSGet("x", NULL, NULL, NULL, selectionSet);
long ssLength;
acedSSLength(selectionSet, &ssLength);
for( int i = 0; i < ssLength; ++i )
{
    if( acedSSName(selectionSet, i, ename) == RTNORM )
    {
        pGetFieldsRb = ade_odgetfield(ename, pszOdTable, pszOdField, recnum);
        struct resbuf* rb = pGetFieldsRb;
        while(rb != NULL) {
            acutPrintf(
                "\nThe current entity has %s object data with a %s value of: %s",
                pszOdTable, pszOdField, rb->resval.rstring);
            rb = rb->rbnext;
        }
    }
}
acutRelRb(pGetFieldsRb);
ade_odgetrecfield

Object Data Functions

Gets a field value using a record ID.

```c
struct resbuf*
ade_odgetrecfield(
    ade_id recID,
    char* field);
```

Returns a field value (type varies) or NULL.

- `recID` Record ID returned by `ade_odgetrecord`.
- `field` Field name.

This function uses the record ID assigned by `ade_odgetrecord` to get the value of a particular field. This means of getting an object data field value is generally faster than any other.

You must release the `resbuf`.

The following sample creates a selection set of all entities in the current drawing using `acedSSGet()`. Each entity in the selection set is passed to `ade_odgetrecord()` which returns a record id. This record id along with other required parameters, are passed to `ade_odgetrecfield()` which stores its results in a `resbuf` which is displayed. Then it releases the `resbuf`, as required.

```c
struct resbuf* pOdRecFieldRb = NULL;
ads_name selectionSet;
ads_name ename;
char* pszOdTable = "table1";
char* pszOdField = "FIELD1";
int recnum = 0;
acedSSGet("x", NULL, NULL, NULL, selectionSet);
long ssLength;
acedSSLen
```
if ( acedSSName(selectionSet, i, ename) == RTNORM )
{
    recordId = ade_odgetrecord(ename, pszOdTable, recnum);
    pOdRecFieldRb = ade_odgetrecfield(recordId, pszOdField);
    struct resbuf* rb = pOdRecFieldRb;
    while (rb != NULL) {
        acutPrintf(
            "The current entity has %s object data with a %s value of: %s",
            pszOdTable, pszOdField, rb->resval.rstring);
        rb = rb->rbnext;
    }
}
acutRelRb(pOdRecFieldRb);
gets a record ID.

```
ade_id
ade_odgetrecord(
    ads_name ename,
    char* table,
    int recnum);
```

returns a record ID or ADE_NULLID.

- **ename**: AutoCAD object name.
- **table**: Table name.
- **recnum**: Record number; the first record number is 0.

The function assigns an ID to the record uniquely determined by the three arguments. Later you can use this record ID with `ade_odgetrecfield` to return the value of a particular field of this record. This means of getting an object data field value is generally faster than any other.

Three arguments are necessary because an AutoCAD object can be associated with more than one record in a table, in which case the records are distinguished by their record numbers. If there is only one record, its number is 0. For more information about records and record numbers, see `ade_odaddrecord`.

The following sample creates a selection set of all entities in the current drawing using `acedSSGet()`. Each entity in the selection set is passed to `ade_odgetrecord()` with all required parameters, the results are checked for value and displayed.

```
ads_name selectionSet;
ads_name ename;
char* pszOdTable = "Table1";
int recnum = 0;
acedSSGet("_x", NULL, NULL, NULL, selectionSet);
long ssLength;
acedSSLengeth( selectionSet, &ssLength );
ade_id recordId;
```
for( int i = 0; i < ssLength; ++i )
{
    if( acedSSName(selectionSet, i, ename) == RTNORM )
    {
        recordId = ade_odgetrecord(ename, pszOdTable, recnum);
        if (0 == recordId) {
            acutPrintf(
                "\nNo record id could be obtained.\n");  
        }  else {
            acutPrintf(
                "\nade_odgetrecord() returned: %.0lf",
                recordId);  
        }  
    }  
}  

ade_odgettables

**Object Data Functions**

Lists the tables attached to an object.

```c
struct resbuf*
ade_odgettables(
    ads_name ename);
```

Returns a list of table names (string) or NULL.

*ename* AutoCAD object name.

You must release the **resbuf**.

An object can have records of more than one table attached. This function lists all the tables that have records attached to the object. See ade_odaddrecord for information about records attached to objects.

An object can have more than one record from the same table attached. To find how many records of a given table are attached, use ade_odrecordqty.

The following sample creates a selection set of all entities in the current drawing using acedSSGet(). Each entity in the selection set is passed to ade_odgettables() with all required parameters, the results are stored in a **resbuf** and displayed. Then it releases the **resbuf**, as required.

```c
struct resbuf* pOdTablesRb = NULL;
ads_name selectionSet;
ads_name ename;
acedSSGet("x", NULL, NULL, NULL, selectionSet);
long ssLength;
acedSSLength( selectionSet, &ssLength );
for( int i = 0; i < ssLength; ++i )
{
    if( acedSSName(selectionSet, i, ename) == RTNORM )
    {
        pOdTablesRb = ade_odgettables(ename);
        struct resbuf* rb = pOdTablesRb;
        while(rb != NULL) {
            acutPrintf(
```
"The current entity has object data from %s attached: ", rb->resval.rstring);
    rb = rb->rbnxt;
  }
}
acutRelRb(pOdTablesRb);
ade_odmodifyfield

Object Data Functions

Modifies field properties in a table.

```c
int ade_odmodifyfield(
    char* tableName,
    struct resbuf* fieldList);
```

Returns RTNORM or an error code.

tableName Table name can be up to 25 characters long. Must be unique, contain no spaces, and start with an alphanumeric character.

fieldList Fields to modify. A sequence of field definitions. See Field Definitions below.

The ade_odmodifyfield function affects all active drawings in the drawing set. There should not be any queried objects for this operation.

**Note** This function will not operate unless your end user has superuser privileges.

**Field Definitions**

The field definitions are introduced by the "columns" string. Each field definition is a list of a-lists, and each a-list consists of a field property and a value, as follows:

<table>
<thead>
<tr>
<th>Field Property</th>
<th>Field Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>colname</td>
<td>Field name (RTSTR) can be up to 31 characters long. Must be unique, contain no spaces, and start with an alphanumeric character</td>
</tr>
<tr>
<td>coldesc</td>
<td>Field description (RTSTR)</td>
</tr>
<tr>
<td>coltype</td>
<td>Field data type</td>
</tr>
<tr>
<td>defaultval</td>
<td>Default field value</td>
</tr>
</tbody>
</table>

In the fieldlist argument, specify the fields to modify and their new field definitions. The function acts on
these fields only and has no effect on any others. For each field you specify, the function replaces the existing field definition with the new field definition in the table and in each of its records. In each record, the modified fields are assigned default values that correspond to their new field definitions. In other words, the function replaces field definitions in each set of object data defined by the table and attached to an object.

The fieldlist argument has the same format as the fieldlist argument in ade_odaddfield. The entry for this function has source code examples.

If a new field definition changes the field type, field values in existing records are converted to the new type if possible. This conversion may alter the values. For example, if you change the field type from real to integer, existing field values are converted by truncating their decimal parts.

The following sample creates a resbuf containing the field definitions to modify. ade_odmodifyfield() uses this resbuf and returns a result code indicating a successful operation. Then it releases the resbuf, as required.

```c
struct resbuf* pModifyFieldRb = NULL;
char* pszOdTable = "table1";
pModifyFieldRb = acutBuildList(
    RTSTR, "columns",
    RTLB,
    RTLB,
    RTSTR, "colname", RTSTR, "FIELD1",
    RTDOTE,
    RTLB,
    RTSTR, "coldesc", RTSTR, "Field 1 Description",
    RTDOTE,
    RTLB,
    RTSTR, "coltype", RTSTR, "integer",
    RTDOTE,
    RTLB,
    RTSTR, "defaultval", RTSTR, "0",
    RTDOTE,
    RTL,
    // Define more fields as needed
    0);
int nResultCode = ade_odmodifyfield(pszOdTable, pModifyFieldRb);
if (RTNORM == nResultCode) {
    acutPrintf("\nThe specified fields have been successfully modified.");
}
else {
    acutPrintf(
```
"The specified fields have not been modified.");
}
acutRelRb(pModifyFieldRb);
ade_odmodifytab

Object Data Functions

Redefines a table.

```c
int ade_odmodifytab(
    struct resbuf* tab_defn);
```

Returns RTNORM or an error code.

`tab_defn` List of table elements: the name of the table you will redefine, a new table description, and a sequence of new field definitions.

The `tab_defn` argument has the same format as the `tab_defn` argument in `ade_oddefinetab`. The entry for this function has source code examples.

For the table you specify in the `tab_defn` argument, the function replaces the existing table definition with the new one. For every object to which the table is attached, the corresponding fields of each record of the table are replaced. The old fields are deleted, and the new fields are assigned default values in accord with their field definitions.

The `ade_odmodifyfield` function affects all active drawings in the drawing set. There should not be any queried objects for this operation.

**Note** This function will not operate unless your end user has superuser privileges.

The following sample creates a `resbuf` containing the updated table definition. `ade_odmodifytab()` uses this `resbuf` and returns a result code indicating a successful operation. Then it releases the `resbuf`, as required.

```c
struct resbuf* pOdTableModRb = acutBuildList(
    RTLB,
    RTSTR, "tablename", RTSTR, "table1",
    RTDOTE,
    RTLB,
    RTSTR, "tabledesc", RTSTR, "New Sample Table",
    RTDOTE,
    RTLB,
    RTSTR, "columns",
```
int nResultCode = ade_odmodifytab (pOdTableModRb);
if (RTNORM == nResultCode) {
    acutPrintf("\nThe specified table has been successfully modified.\n");
} else {
    acutPrintf("\nThe specified table has not been modified.\n");
}
acutRelRb(pOdTableModRb);
ade_odnewrecord
Object Data Functions

Defines a new object data record.

ade_id
ade_odnewrecord(char* table);

Returns a new record ID or NULL.

| table  | An existing table name to which the new record will belong. |

The function creates a new record, populates its fields with default values according to the table definition, and returns the new record ID.
ade_odpresetfield

**Object Data Functions**

Assigns a value to a field in a new record.

```c
int ade_odpresetfield(
    ade_id rec_id,
    char* field,
    struct resbuf* value);
```

Returns **RTNORM** or an error code.

- **rec_id** Record ID returned by `ade_odnewrecord`.
- **field** Field name.
- **value** Field value.

Sets the value of a field in an Object Data record defined through `ade_odnewrecord`.

The following sample creates a new object data record for "table1" using `ade_odnewrecord()`. A resbuf is created which contains the new value to be applied to "field1" in the new object data record. `Ade_odpresetfield()` is called with all required parameters. A selection set containing all entities in the current drawing is created, then each entity has the new object data record attached using `ade_odattachrecord()`. `Ade_odfreerec()` is called to free the memory claimed in defining a new record and the resbuf is released as required.

```c
char* pszOdTable = "table1";
ade_id recId = ade_odnewrecord(pszOdTable);
struct resbuf* pOdFieldValRb = acutBuildList(RTSTR, "Newvalue",0);
char* pszOdField = "field1";
int returnCode = ade_odpresetfield(recId, pszOdField, pOdFieldValRb);
ads_name selectionSet;
acedSSGet("_x", NULL, NULL, NULL, selectionSet);
ads_name ename;
long ssLength;
acedSSLength(selectionSet, &ssLength);
```
for( int i = 0; i < ssLength; ++i )
{
    if( acedSSName(selectionSet, i, ename) == RTNORM )
    {
        int returnCode = ade_odattachrecord(ename, recordId);
    }
}
returnCode = ade_odfreerec(recordId);
acutRelRb(pOdFieldValRb);
ade_odrecordqty

Object Data Functions

Counts the records attached to an object.

```c
int ade_odrecordqty(
    ads_name ename,
    char* table);
```

Returns a record count or an error code.

- **ename**: AutoCAD object name.
- **table**: Table name can be up to 25 characters long. Must be unique, contain no spaces, and start with an alphanumeric character.

The function counts how many records of the same table are attached to the object. See [ade_odaddrecord](#) for more information about attaching records to objects.
ade_odsetfield

Object Data Functions

Sets a field value.

```c
int ade_odsetfield(
    ads_name ename,
    char* table,
    char* field,
    int recnum,
    struct resbuf* value);
```

Returns **RTNORM** or an error code.

**ename**  AutoCAD object name.

**table**  Table name can be up to 25 characters long. Must be unique, contain no spaces, and start with an alphanumeric character.

**field**  Field name can be up to 31 characters long. Must be unique, contain no spaces, and start with an alphanumeric character.

**recnum**  Record number; the first record number is 0.

**value**  New field value.

To identify a unique record, you need to specify the table to which it belongs, the object to which it is attached, and its record number. The record number is necessary because more than one record from the same table can be attached to an object. For more information about records and record numbers, see **ade_odaddrecord**.

The following sample creates a **resbuf** containing the updated field value. A selection set of all entities in the current drawing is created using **acedSSGet()**. Each entity in the selection set is passed to **ade_odsetfield()** with all required parameters, the returned result code is checked for **RTNORM** and displayed. Then it releases the **resbuf**, as required.

```c
char* pszOdTable = "table1";
ade_id recordId = ade_odnewrecord(pszOdTable);
```
struct resbuf* pOdSetFieldValRb = acutBuildList(RTSTR, "Newvalue", 0);
int recnum = 0;
char* pszOdField = "field1";
ads_name selectionSet;
ads_name ename;
acedSSGet("_x", NULL, NULL, NULL, selectionSet);
long ssLength;
acedSSLength( selectionSet, &ssLength; );
for( int i = 0; i < ssLength; ++i )
{
    if( acedSSName( selectionSet, i, ename ) == RTNORM )
    {
        int nResultCode = ade_odsetfield(
            ename,
            pszOdTable,
            pszOdField,
            recnum,
            pOdSetFieldValRb);
        if (RTNORM == nResultCode) {
            acutPrintf("
The field value for the specified record has been modified.");
        }
        else {
            acutPrintf("The field value for the specified record has not been modified.");
        }
    }
}
acutRelRb(pOdSetFieldValRb);
ade_odtabledefn

Object Data Functions

Gets a table definition.

```c
struct resbuf*
ade_odtabledefn(
    char* table);
```

Returns a table definition or NULL.

table     Table name can be up to 25 characters long. Must be unique, contain no spaces, and start with an alphanumeric character.

You must release the resbuf.

The table definition returned by ade_odtabledefn has the same format as the tab_defn argument in ade_oddefinetab. The entry for this function has source code examples.

The following sample populates a resbuf with object data table schema information using ade_odtabledefn(). The contents of the resbuf are displayed and the resbuf is released as required.

```c
char* pszOdTable = "rirds";
struct resbuf* pOdTableDefRb = ade_odtabledefn(pszOdTable);
if(NULL != pOdTableDefRb) {
    acutPrintf(
        "%nThe following information was obtained about the \"%s\" table:
        , pszOdTable);
    struct resbuf* rb = pOdTableDefRb;
    while(NULL != rb) {
        if (rb->restype == RTSTR) {
            acutPrintf(
                "%\n\t\"%s\"\n                , rb->resval.rstring);
        }
        rb = rb->rbnext;
    }
}
```
ade_odtablelist
Object Data Functions

Lists the tables in the drawing set.

struct resbuf*  
ade_odtablelist();

Returns a list of table names (string) or NULL.

You must release the resbuf.

The following sample parses the resbuf returned by ade_odtablelist() and prints a list of object data tables in the current drawing. Then it releases the resbuf, as required.

```c
struct resbuf* pOdTableListRb = NULL;
pOdTableListRb = ade_odtablelist();
if (NULL != pOdTableListRb) {
    acutPrintf("\nThe current drawing contains the following Object Data tables:");
    int nOdTables = 0;
    struct resbuf* rb = pOdTableListRb;
    while(rb != NULL) {
        ++nOdTables;
        acutPrintf("\nObject data table %d is named: %s"
            ,nOdTables, rb->resval.rstring);
        rb = rb->rbnext;
    }
} else {
    acutPrintf("\nThere are no object data tables in this project.");
}
acutRelRb(pOdTableListRb);
```
ade_osfexpand

Other Functions

Searches a directory and returns a list of file names.

```c
struct resbuf*
ade_osfexpand(
    char* path,
    char* extension,
    char* pattern);
```

Returns a list of file names or NULL.

- **path**: Directory in which to search or NULL. If NULL, the function searches the working directory.
- **extension**: File name extension or NULL. If NULL, the function uses "dwg".
- **pattern**: Wild card pattern or NULL. If NULL, the function uses "*" (search for all file names with the given extension and path).

You must release the `resbuf`.

For information about wild card patterns, look up "wild-card characters" on the Index tab of AutoCAD Map Help.

The following sample parses the `resbuf` returned by `ade_osfexpand()` and prints a list of matching files. Then it releases the `resbuf`, as required.

```c
char* pszFilePath = "C:\MyFiles";
char* pszFileExtension = "dwg";
char* pszFilePattern = NULL;
struct resbuf* pFileNamesRb = ade_osfexpand(pszFilePath, pszFileExtension, pszFilePattern);
while(pFileNamesRb != NULL) {
    acutPrintf(
        "%nThe following file of type "%s" were found in %s: %s"
        , pszFileExtension, pszFilePath, pFileNamesRb->resval.rstring);
    pFileNamesRb = pFileNamesRb->rbnext;
}
acutRelRb(pFileNamesRb);
ade_prefgetval

Option Functions

Gets an AutoCAD Map option setting.

struct resbuf*
ade_prefgetval(
    char* variable);

Returns an option value or NULL.

variable Option name. See the Options tables below.

You must release the resbuf.

The function return value depends on which option you specify. The tables below show option names and return values, organized by option type.

Work Session Options

<table>
<thead>
<tr>
<th>Option Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>RestoreLastActiveDwgsOnStartup</td>
<td>-1 or NULL.</td>
</tr>
<tr>
<td>ActivateDwgsOnAttach</td>
<td>-1 or NULL.</td>
</tr>
<tr>
<td>DontAddObjectsToSaveSet</td>
<td>-1 or NULL.</td>
</tr>
<tr>
<td>MarkObjectsForEditingWithoutPrompting</td>
<td>-1 or NULL.</td>
</tr>
<tr>
<td>LogFileActive</td>
<td>-1 or NULL.</td>
</tr>
<tr>
<td>LogFileName</td>
<td>File name (RTSTR). For example, &quot;ade.log&quot;.</td>
</tr>
<tr>
<td>LogMessageLevel</td>
<td>0, 1, or 2.</td>
</tr>
</tbody>
</table>

Query Options

<table>
<thead>
<tr>
<th>Option Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>QueryFileDirectory</td>
<td>Path (RTSTR). For example, &quot;c:\data\qry&quot;.</td>
</tr>
<tr>
<td>CaseSensitiveMatch</td>
<td>-1 or NULL.</td>
</tr>
<tr>
<td>SaveCurrQueryInSession</td>
<td>-1 or NULL.</td>
</tr>
<tr>
<td>Variable</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>---------------------------------------------------</td>
</tr>
<tr>
<td>MkSelSetWithQryObj</td>
<td>-1 or NULL.</td>
</tr>
<tr>
<td>DefaultJoinOperator</td>
<td>1 = OR, 2 = AND.</td>
</tr>
<tr>
<td>ColorForAdd</td>
<td>Color (RTSTR).</td>
</tr>
<tr>
<td>ColorForRemove</td>
<td>Color (RTSTR).</td>
</tr>
<tr>
<td>BlockLocnForQuery</td>
<td>1 = insertion point, 2 = bounding box.</td>
</tr>
<tr>
<td>TextLocnForQuery</td>
<td>1 = insertion point, 2 = bounding box.</td>
</tr>
<tr>
<td>ShowBlockAsInsPt</td>
<td>-1 or NULL.</td>
</tr>
<tr>
<td>ShowImageAsBoundary</td>
<td>-1 or NULL.</td>
</tr>
<tr>
<td>CreateAssociativeHatchObjects</td>
<td>-1 or NULL.</td>
</tr>
<tr>
<td>ReferenceBoundaryForAreaLocation</td>
<td>-1 or NULL.</td>
</tr>
<tr>
<td>Save Back Options</td>
<td></td>
</tr>
<tr>
<td>RedefineBlockDefinitions</td>
<td>-1 or NULL.</td>
</tr>
<tr>
<td>RedefineLayerDefinitions</td>
<td>-1 or NULL.</td>
</tr>
<tr>
<td>RedefineTextStyleDefinitions</td>
<td>-1 or NULL.</td>
</tr>
<tr>
<td>RemoveUnusedGroups</td>
<td>-1 or NULL.</td>
</tr>
<tr>
<td>EraseSavedBackObjects</td>
<td>-1 or NULL.</td>
</tr>
<tr>
<td>RemoveLockAfterSave</td>
<td>-1 or NULL.</td>
</tr>
<tr>
<td>CreateHistoryFileOfChanges</td>
<td>-1 or NULL.</td>
</tr>
<tr>
<td>CreateBackupFileOfSourceDwg</td>
<td>-1 or NULL.</td>
</tr>
<tr>
<td>External Database Options</td>
<td></td>
</tr>
<tr>
<td>NoOfSQLConditionsInHistory</td>
<td>RTSHORT.</td>
</tr>
<tr>
<td>DisplayTabsInSingleView</td>
<td>-1 or NULL.</td>
</tr>
<tr>
<td>OpenDataViewReadOnly</td>
<td>-1 or NULL.</td>
</tr>
<tr>
<td>SaveDataViewFmtChanges</td>
<td>-1 or NULL.</td>
</tr>
<tr>
<td>ReconnectDbOnWSOpen</td>
<td>-1 or NULL.</td>
</tr>
</tbody>
</table>
### ShowFullDBPath
-1 or NULL.

### KeepDataViewOnTop
-1 or NULL.

### dbfDatabases
RTSTR, one of the following: "Prompt", "DB3", "DB4", "DB5", "FOX2.0", "FOX2.5", or "FOX2.6".

### xlsDatabases
RTSTR, one of the following: "Prompt", "Excel3", "Excel4", "Excel5", or "Excel7".

### dbDatabases
RTSTR, one of the following: "Prompt", "Paradox3.0", "Paradox4.0", or "Paradox5.0".

### Coordinate Transformation Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>AdjustSizesAndScalesForChangesInUnits</td>
<td>-1 or NULL</td>
</tr>
<tr>
<td>AdjustRotationsForMapDistortions</td>
<td>-1 or NULL</td>
</tr>
<tr>
<td>AdjustSizesAndScalesForMapDistortions</td>
<td>-1 or NULL</td>
</tr>
<tr>
<td>AdjustElevations</td>
<td>-1 or NULL</td>
</tr>
<tr>
<td>AdjustZeroRotationObjects</td>
<td>-1 or NULL</td>
</tr>
</tbody>
</table>

### System Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>AccessWorkCenter</td>
<td>-1 or NULL</td>
</tr>
<tr>
<td>CheckoutDirectory</td>
<td>Path (RTSTR). For example, &quot;c:\data\dwg&quot; or &quot;&quot; if none.</td>
</tr>
<tr>
<td>PreserveAWCFiles</td>
<td>-1 or NULL</td>
</tr>
<tr>
<td>ForceUserLogin</td>
<td>-1 or NULL</td>
</tr>
<tr>
<td>EnableObjectLocking</td>
<td>-1 or NULL</td>
</tr>
<tr>
<td>ReadPrefFromINI</td>
<td>-1 or NULL</td>
</tr>
<tr>
<td>NumberofOpenDwgs</td>
<td>RTSHORT</td>
</tr>
<tr>
<td>DoublePrec</td>
<td>RTREAL, 0 or greater, but less than 1.</td>
</tr>
</tbody>
</table>

The "ForceUserLogin" and "DoublePrec" system options cannot be modified unless your end user has
superuser privileges.

If "DoublePrec" is set to 0, the behavior of data extension queries is the same as before introducing this option. The "DoublePrec" option has no user interface equivalent.

### Workspace Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>CheckClasses</td>
<td>-1 or NULL</td>
</tr>
<tr>
<td>CheckDrawings</td>
<td>-1 or NULL</td>
</tr>
<tr>
<td>CheckQueryLibrary</td>
<td>-1 or NULL</td>
</tr>
<tr>
<td>CheckDatabases</td>
<td>-1 or NULL</td>
</tr>
<tr>
<td>CheckTables</td>
<td>-1 or NULL</td>
</tr>
<tr>
<td>CheckQueries</td>
<td>-1 or NULL</td>
</tr>
<tr>
<td>CheckTopologies</td>
<td>-1 or NULL</td>
</tr>
<tr>
<td>CheckLPNs</td>
<td>-1 or NULL</td>
</tr>
<tr>
<td></td>
<td>Note that link path names (LPNs) have been replaced by link templates in AutoCAD Map.</td>
</tr>
<tr>
<td>ShowOPMOOnStartup</td>
<td>-1 or NULL</td>
</tr>
<tr>
<td>ShowWSpaceOnStartup</td>
<td>-1 or NULL</td>
</tr>
<tr>
<td>WSForwardView</td>
<td>-1 or NULL</td>
</tr>
<tr>
<td>WSWindowRect</td>
<td>A list of four values (RTSHORT) that define the left, top, right, and bottom of the window rectangle.</td>
</tr>
</tbody>
</table>

Database tables and database query categories are visible in the workspace only if "CheckTables" and "CheckQueries" are set to T and "CheckDatabases" is set to T also.

The following workspace options are read only. That is, they can be used only with ade_prefgetval to determine if a category is visible in the workspace.

### Read-Only Workspace Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>ClassesVisible</td>
<td>-1 or NULL</td>
</tr>
<tr>
<td>DrawingsVisible</td>
<td>-1 or NULL</td>
</tr>
<tr>
<td>QueryLibraryVisible</td>
<td>-1 or NULL</td>
</tr>
<tr>
<td>DatabasesVisible</td>
<td>-1 or NULL</td>
</tr>
</tbody>
</table>
The following sample parses the resbuf returned by `ade_prefgetval()` and prints the value for the specified option using a `switch case` statement. Then it releases the resbuf, as required.

```c
char* pszOptionVar = "LogFileName";
struct resbuf* pGetPrefValRb = ade_prefgetval(pszOptionVar);
if(pGetPrefValRb != NULL){
    struct resbuf* rb = pGetPrefValRb;
    while(rb != NULL) {
        switch(rb->restype)
        {
            case RTT:
                acutPrintf("The %s option variable contained the value: %d",
                           pszOptionVar, rb->resval.rint);
                break;
            case RTSHORT:
                acutPrintf("The %s option variable contained the value: %d",
                           pszOptionVar, rb->resval.rint);
                break;
            case RTSTR:
                acutPrintf("The %s option variable contained the value: %s",
                           pszOptionVar, rb->resval.rstring);
                break;
            default:
                acutPrintf("Could not determine the value");
                break;
        }
        rb = rb->rbnext;
    }
} else {
```
acutPrintf("\nNo value was assigned to %s.", pszOptionVar);
}
acutRelRb(pGetPrefValRb);
**ade_prefsetval**

**Option Functions**

Sets an AutoCAD Map option.

```c
int ade_prefsetval(
    char* variable,
    struct resbuf* value);
```

Returns RTNORM or an error code.

- **variable** Option name.
- **value** Value appropriate for the given option (type varies).

See [ade_prefgetval](#) for a list of option names and values.

The following example sets "ColorForAdd" to "red".

```c
ade_prefsetval("ColorForAdd", rb);
```

The argument `rb` points to a resbuf structure containing

```c
rb->resval.rstring = "red"
```

You can create a new resbuf structure with `ads_newrb()`.

The following sample creates a resbuf containing the updated option value. **ade_prefsetval()** is called with all required parameters, the returned result code is checked for RTNORM and displayed. Then it releases the resbuf, as required.

```c
char* pszOptionVar = "LogFileName";
struct resbuf* pSetPrefValRb = acutBuildList(RTSTR, "ade.log", 0);
int nResultCode = ade_prefsetval(pszOptionVar, pSetPrefValRb);
if (RTNORM == nResultCode) {
    acutPrintf("%nThe %s option variable has been successfully modified to: %s", pszOptionVar, pSetPrefValRb->resval.rstring);
```
} else {
    acutPrintf(
        \"nThe specified options variable has not been modified.\");
} acutRelRb(pSetPrefValRb);
ade_projgetctgname

Coordinate Transformation Functions

Identifies the category that a coordinate system belongs to.

char *
ade_projgetctgname(
    char* cscode);

Returns a coordinate system category or NULL.

cscode  Coordinate system code, eight characters.
ade_projgetinfo

Coordinate Transformation Functions

Gets information about a projection system.

```c
struct resbuf*
ade_projgetinfo(  
    char* cscode,  
    char* info_type);  
```

Returns a piece of coordinate system information or NULL.

- **cscode**: Coordinate system code, eight characters.
- **info_type**: Information type; see Information Types below.

You must release the `resbuf`.

### Information Types

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>description</td>
<td>Description. For example, &quot;World Geodetic System of 1984 Latitude/Longitude in Degrees&quot;.</td>
</tr>
<tr>
<td>projection</td>
<td>Projection. For example, &quot;Unity Conversion, produce/accept lat/longs&quot;.</td>
</tr>
<tr>
<td>datum</td>
<td>Datum. For example, &quot;North American Datum of 1927, Mean Values&quot;.</td>
</tr>
</tbody>
</table>

The following sample parses the `resbuf` returned by `ade_projgetinfo()` and prints the value for the specified option. Then it releases the `resbuf`, as required.

```c
char* pszCategoryCode = "MO-W";
char* pszInfoType = "description";
pCoordSysRb = ade_projgetinfo(pszCategoryCode, pszInfoType);
struct resbuf* rb = pCoordSysRb;
while(pCoordSysRb != NULL) {
    acutPrintf(  
        "\nThe following %s information is available for the %s coordinate system:\n\n%s"  
        , pszInfoType, pszCategoryCode, pCoordSysRb->resval.rstring);  
    
```
pCoordSysRb = pCoordSysRb->rbnext;
}
acutRelRb(pCoordSysRb);
ade_projgetwscode

*Coordinate Transformation Functions*

Gets the current drawing's coordinate system code.

```c
char*
ade_projgetwscode();
```

Returns a coordinate system code or the empty string.
ade_projlistcrdsysts

Coordinate Transformation Functions

Lists available coordinate systems in a given category.

```c
struct resbuf*
ade_projlistcrdsysts(
    char* categoryName);
```

Returns a list of available coordinate systems or NULL.

categoryName       Coordinate system code, eight characters.

The following sample parses the resbuf returned by `ade_projlistcrdsysts()` and prints the value for the specified option. Then it releases the resbuf, as required.

```c
char* pszCategoryName = "Lat Longs";
struct resbuf* pCoordSysRb = ade_projlistcrdsysts(pszCategoryName);
if(pCoordSysRb != NULL){
    acutPrintf(
        "%nThe following coordinate system(s) have been detected for the %s catagory:%n",
        pszCategoryName);
    struct resbuf* rb = pCoordSysRb;
    while(rb != NULL) {
        acutPrintf(
            "\n\n%\n%",          // Field mask
        , rb->resval.rstring);
        rb = rb->rbnext;
    }
}
else{
    acutPrintf(
        "%\nNo coordinate system(s) have been detected for the %s catagory: %s"
        , pszCategoryName);
}
acutRelRb(pCoordSysRb);
```
ade_projlistctgy
Coordinate Transformation Functions

Lists available coordinate system categories.

```c
struct resbuf* ade_projlistctgy();
```

Returns a list of coordinate system categories or `NULL`.

The following sample parses the `resbuf` returned by `ade_projlistctgy()` and prints the coordinate system category list. Then it releases the `resbuf`, as required.

```c
struct resbuf* pCoordSysCtgyRb = ade_projlistctgy();
if(pCoordSysCtgyRb != NULL) {
    acutPrintf("The following coordinate system categories have been detected:");
    struct resbuf* rb = pCoordSysCtgyRb;
    while(rb != NULL) {
        acutPrintf("%s", rb->resval.rstring);
        rb = rb->rbnext;
    }
} else {
    acutPrintf("No coordinate system categories have been detected");
}
acutRelRb(pCoordSysCtgyRb);
```
ade_projptbackward

Coordinate Transformation Functions

Converts point coordinates from destination coordinate system to source.

```c
int
ade_projptbackward(
    ads_point pt,
    ads_point result);
```

Returns RTNORM or an error code.

- **pt**  Destination point to convert, a set of 2D or 3D coordinate values. If 3D, the Z value is ignored.
- **result**  Corresponding source values.

Before you can use `ade_projptbackward` to convert points, you must first identify the coordinate systems that you are converting between. Use `ade_projsetsrc` to set the source system and `ade_projsetdest` to set the destination system. The `ade_projptbackward` function assumes that the coordinate values you pass to it belong to the destination system, and it returns corresponding source values. The `ade_projptforward` function does the inverse.

The following example prompts the user for a destination point and returns the corresponding source coordinates through the `result` parameter.

```c
ads_point pt1;
ads_point result;
ads_getpoint(NULL, "Pick a point", pt1);
ade_projptbackward (pt1, result);
```

For more information, see [Converting Coordinates](#).
ade_projptforward

Coordinate Transformation Functions

Converts point coordinates from source coordinate system to destination.

```c
int ade_projptforward(
    ads_point pt,
    ads_point result);
```

Returns **RTNORM** or an error code.

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>pt</code></td>
<td>Source point to convert, a set of 2D or 3D coordinate values. If 3D, the Z value is ignored.</td>
</tr>
<tr>
<td><code>result</code></td>
<td>Corresponding destination values.</td>
</tr>
</tbody>
</table>

Before you can use `ade_projptforward` to convert points, you must first identify the coordinate systems that you are converting between. Use `ade_projsetsrc` to set the source system and `ade_projsetdest` to set the destination system. The `ade_projptforward` function assumes that the coordinate values you pass to it belong to the source system, and it returns corresponding destination values. The `ade_projptbackward` function does the inverse.

The following example prompts the user for a source point and passes the corresponding destination coordinates through the `result` parameter.

```c
ads_point pt1;
ads_point result;
ads_getpoint(NULL, "Pick a point", pt1);
ade_projptforward(pt1, result);
```

For more information, see [Converting Coordinates](#).
Sets the destination coordinate system for converting points.

```c
int ade_projsetdest(
    char* cscode);
```

Returns RTNORM or an error code.

code Coordinate system code, eight characters.

Before you can use either ade_projptforward or ade_projptbackward to convert points, you must first identify the coordinate systems that you are converting between. Use ade_projsetdest to set the destination system and ade_projsetsrc to set the source system. The ade_projptforward function assumes that the coordinate values you pass to it belong to the source system, and it returns corresponding destination values. The ade_projptbackward function does the inverse.

For more information, see Converting Coordinates.
ade_projsetsrc

Coordinate Transformation Functions

Sets the source coordinate system for converting points.

int ade_projsetsrc(
    char* cscode);

Returns RTNORM or an error code.

cscode Coordinate system code, eight characters.

Before you can use either ade_projptforward or ade_projptbackward to convert points, you must first identify the coordinate systems that you are converting between. Use ade_projsetsrc to set the source system and ade_projsetdest to set the destination system. The ade_projptforward function assumes that the coordinate values you pass to it belong to the source system, and it returns corresponding destination values. The ade_projptbackward function does the inverse.

For more information, see Converting Coordinates.
Sets the coordinate system for the current drawing.

```c
int ade_projsetwscode(
    char* cscode);
```

Returns RTNORM or an error code.

code Coordinate system code, eight characters.
`ade_qldelctgy`  
*Query Library Functions*

Deletes a query library category.

```c
int ade_qldelctgy(
    ade_id ctgy_id);
```

Returns `RTNORM` or an error code.

- `ctgy_id` Category ID.
ade_qldelquery

Query Library Functions

Deletes a query from the query library.

int ade_qldelquery(ade_id qry_id);

Returns RTNORM or an error code.

qry_id Query ID.
ade_qlgetctgyinfo

Query Library Functions

Gets information about a query category.

```c
struct resbuf*
ade_qlgetctgyinfo(
    ade_id ctg_id,
    char* info);
```

Returns the requested information or NULL.

ctg_id  Category ID.
info  Type of category information to get: "name" to get the category name, or "qrylist" to get a list of query IDs of the queries in the category.

You must release the resbuf.

The information returned depends on the info argument you use, but it is always in list format. For example:

The following sample populates a resbuf with query category id's using ade_qllistctgy(). This resbuf is then used by ade_qlgetctgyinfo() to populate a resbuf containing category names or category id's depending on the value of info. The switch case statement prints the appropriate information, (category names or query Id's). Then it releases the resbuf, as required.

```c
struct resbuf* pQryCtgryRb = NULL;
char* pszCategoryInfo = "qrylist";
struct resbuf* pQryCtgryId = ade_qllistctgy();
if(pQryCtgryId != NULL) {
    struct resbuf* rb = pQryCtgryId;
    while(rb != NULL) {
        pQryCtgryRb = ade_qlgetctgyinfo(rb->resval.rreal, pszCategoryInfo);
        if(pQryCtgryRb != NULL) {
            switch(pQryCtgryRb->restype) {
                case RTREAL:
                    acutPrintf("The %s option you've specified produced the following query category id: %.0f\n", pszCategoryInfo, pQryCtgryRb->restval.rreal);
                    break;
            }
        }
    }
}
```

case RTSTR:
    acutPrintf(
        "\nThe %s option you've specified produced the following query category name: %s"
        , pszCategoryInfo, pQryCtgryRb->resval.real);
    break;
default:
    acutPrintf(  
        "\nCould not determine the value");
    break;
    }
    }
rb = rb->rbnext;
    }
}
else
    {
    acutPrintf(  
        "\nThe %s category you've requested produced no information"
        , pszCategoryInfo);
    }
acutRelRb(pQryCtgryId);
acutRelRb(pQryCtgryRb);
ade_qlgetqryinfo

Query Library Functions

Gets information about a query.

struct resbuf*
ade_qlgetqryinfo(
    ade_id qry_id,
    char* info);

Returns the requested information or NULL.

qry_id        Query ID.
info          Information type. See the Information Types table below.

You must release the resbuf.

Information Types

<table>
<thead>
<tr>
<th>name</th>
<th>Query name.</th>
</tr>
</thead>
<tbody>
<tr>
<td>description</td>
<td>Query description.</td>
</tr>
<tr>
<td>category</td>
<td>Category name.</td>
</tr>
<tr>
<td>qtype</td>
<td>Query type: 1 = internal, 2 = external.</td>
</tr>
<tr>
<td>filename</td>
<td>For an externally saved query, full path name (string).</td>
</tr>
</tbody>
</table>

A query gets a name and an ID when it is saved to a query category of the query library. A new query that you have not yet saved does not have a name or an ID.

The following sample obtains a query ID using ade_qlqrygetid(). Ade_qlgetqryinfo() is called with all required parameters and status information is displayed based on the resbuf returned by the function. The resbuf is then released as required.

```c
char* pszQueryName = "LocAll";
ade_id queryId = ade_qlqrygetid(pszQueryName);
```
char* pszQueryInfo = "category";
struct resbuf* pQueryInfoRb = ade qlget qry info(
    queryId,
    pszQueryInfo);
if (NULL != pQueryInfoRb) {
    acutPrintf("nThe "%s" query is contained within the "%s" query category."
        ,pszQueryName, pQueryInfoRb->resval.rstring);
} else {
    acutPrintf("
The requested information could not be found.");
}
acutRelRb(pQueryInfoRb);
ade_qllistctgy

Query Library Functions

Lists the query category IDs.

```
struct resbuf*
ade_qllistctgy();
```

Returns a list of category IDs, or, if there are no categories or an error occurs, `NULL`.

You must release the `resbuf`.

The following sample parses the `resbuf` returned by `ade_qllistctgy()` and prints a list of query category ID's. Then it releases the `resbuf`, as required.

```
struct resbuf* pQueryCategoryListRb = NULL;
pQueryCategoryListRb = ade_qllistctgy();
if (NULL != pQueryCategoryListRb) {
    int nQueryCategories = 0;
    struct resbuf* rb = pQueryCategoryListRb;
    while(rb != NULL) {
        ++nQueryCategories;
        acutPrintf(
            "\nQuery category %d has the Id: %.0lf",
            nQueryCategories, rb->resval.rreal);
        rb = rb->rbnext;
    }
} else {
    acutPrintf(
        "\nThere are no query categories in this project.\n");
}
acutRelRb(pQueryCategoryListRb);
```
ade_qlloadqry

Query Library Functions

Makes a saved query current.

int
ade_qlloadqry(
    ade_id qry_id);

Returns RTNORM or an error code.

qry_id Query ID.

Once loaded, the query becomes the current query. If there is already a current query, this query replaces it.
ade_qlqrygetid

Query Library Functions

Gets a query ID.

```c
ade_id
ade_qlqrygetid
    char* query_name);
```

Returns a query ID or ADE_NULLID.

query_name Query name.

A query gets a name and an ID when it is saved to a query category of the query library. A new query that you have not yet saved does not have a name or an ID.

The query name is enough to identify a query uniquely. The category name is not required. Within the current drawing and all attached source drawings, no two queries can have the same name, even if they are saved in different categories.
ade qlsetctgyname

Query Library Functions

Changes a query category name.

```c
int ade qlsetctgyname(
    ade_id ctgy_id,
    char* name);
```

Returns RTNORM or an error code.

- **ctgy_id**  Query category ID.
- **name**  Name of new category, up to 31 characters long. Must be unique, contain no spaces, and start with an alphanumeric character

The category name cannot contain spaces.
Changes a query name, description, or the category it belongs to.

```c
int ade qlsetquery(
    ade_id qry_id,
    char* info,
    struct resbuf* value);
```

Returns **RTNORM** or an error code.

- **qry_id** Query ID.
- **info** Type of information to modify. See the Query Properties table below.
- **value** New value (type varies).

### Query Properties

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Name of query (<strong>RTSTR</strong>), up to 31 characters long. Must be unique, contain no spaces, and start with an alphanumeric character.</td>
</tr>
<tr>
<td>description</td>
<td>Description of query (<strong>RTSTR</strong>), up to 132 characters long. Can contain spaces. Must be unique and start with an alphanumeric character.</td>
</tr>
<tr>
<td>category</td>
<td>Category ID, (<strong>RTREAL</strong>).</td>
</tr>
</tbody>
</table>

This function does not change file name or storage type.

A query gets a name and an ID when it is saved to a query category of the query library. A new query that you have not yet saved does not have a name or an ID.

The following sample creates a *resbuf* containing the updated option value. The id for the query to be modified is then obtained with `ade qlqrygetid()`. `ade qlsetquery()` is called with all required parameters, the returned result code is checked for **RTNORM** and displayed. Then it releases the *resbuf*, as required.

```c
char* pszQueryProperty = "name";
char* pszQueryName = "RtList";
```
struct resbuf* pQueryPropValRb = acutBuildList(RTSTR, "MyRenamedQuery", 0);
ade_id queryId = ade_qlqrygetid(pszQueryName);
if(queryId != ADE_NULLID) {
    int returnCode = ade_qlsetquery(queryId, pszQueryProperty, pQueryPropValRb);
    if (RTNORM == returnCode) {
        acutPrintf("The query %s property has been modified.", pszQueryProperty);
    }
    else {
        acutPrintf("The query %s property has not been modified.", pszQueryProperty);
    }
} else {
    acutRelRb(pQueryPropValRb);"
ade_qrclear

Query Functions

Clears the current query.

```c
int ade_qrclear();
```

Returns `RTNORM` or an error code.
ade_qrydefine
Query Functions

Defines a query.

ade_id
ade_qrydefine(
    char* joinop,
    char* bggroups,
    char* not_op,
    char* condtype,
    struct resbuf* qrycond,
    char* endgroups);

Returns the ID of the query condition it creates or ADE_NULLID.

joinop A joining operator: "and" or "or" or "" (none). If "" (none) is specified, the default joining
operator is used (see ade_prefgetval).

bggroups For grouping this condition with others in the query definition you are building. Use one
or more open parentheses as needed, or "" (none). For example, "((".

not_op The NOT operator, if needed: "not" or "" (none).

condtype A condition type: "Location", "Property", "Data", or "SQL".

qrycond A condition expression. Depends on the condition type. See Condition Expressions
below.

endgroups For grouping this condition with others in the query definition you are building. Use one
or more close parentheses as needed, or "" (none). For example, ")

A query definition is composed of one or more conditions, each defined by a separate ade_qrydefine call.
You can group conditions by supplying parentheses or empty strings to the bggroups or endgroups
parameters as needed.

You must specify all six ade_qrydefine arguments.

Condition Expressions

The qrycond parameter requires a condition expression. Condition expressions are lists. What you include
in the list depends on the condition type: Location, Property, Data, or SQL.

**Location Expressions**

**Property Expressions**

**Data Expressions**

**SQL Expressions**

You must release the `resbuf`.

The following sample creates a `resbuf` containing the query condition values, (objects classified as Ponds). `ade_qrydefine()` is called with all required parameters, the returned `queryId` is checked for `ADE_NULLID` and an appropriate message is displayed. Then it releases the `resbuf`, as required.

```c
char* pszJoinOperator = "";  // none
char* pszBgnCondGrouping = "";  // none
char* pszNotOperator = "";  // none
char* pszCondType = "Property";
char* pszEndCondGrouping = "";  // none
struct resbuf* pQueryConditionRb = acutBuildList(
    RTLB,
    RTSTR, "feature",
    RTSTR, "=",
    RTSTR, "Pond",
    RTLE,
    0);
ade_id queryId = ade_qrydefine(
    pszJoinOperator,
    pszBgnCondGrouping,
    pszNotOperator,
    pszCondType,
    pQueryConditionRb,
    pszEndCondGrouping);

if (queryId != ADE_NULLID) {
    acutPrintf(
        "%A %s query has been defined."
        , pszCondType);
} else {
    acutPrintf(
        "%A %s query was not defined."
        , pszCondType);
}
acutRelRb(pQueryConditionRb);
```
ade_qryexecute

Query Functions

Executes the current query.

ads_real
ade_qryexecute();

Returns the number of queried objects. If none, it returns 0.0.

Executing a query makes a new selection set of the queried objects if the "MkSelSetWithQryObj" option is turned on, as in the following:

```c
struct resbuf* pSetPrefValRb = acutBuildList(RTT, 0);
ade PREFsetval("MkSelSetWithQryObj", pSetPrefValRb);
acedRelRb(pSetPrefValRb);
```

The "P" argument identifies the "previous" selection set (the objects currently or most recently selected).

**Note** Whenever you create a selection set, you replace the previous selection set. Make sure you know which objects you are getting. The will query run slower in this case because of the extra work involved.

The following sample sets the MkSelSetWithQryObj option which will place the resulting queried objects in a selection set of your specification. Ade_qryexecute() is called and the number of objects placed into the selection set is displayed. The Resbuf is released as required and the selection set is freed.

```c
ads_name ssQueriedObjects;
char* pszOptionVar = "MkSelSetWithQryObj";
struct resbuf* pSetPrefValRb = acutBuildList(RTT, 0);
int nResultCode = ade PREFsetval(pszOptionVar, pSetPrefValRb);
if (RTNORM == nResultCode) {
    ads_real queriedObjects = ade_qryexecute();
    if (ADE_NULLID != queriedObjects) {
        acedSSGet("P", NULL, NULL, NULL, ssQueriedObjects);
        long ssQueriedObjectsLength;
        acedSSLen ssQueriedObjects, &ssQueriedObjectsLength;);
        acutPrintf("
            \%ld objects have been placed in the "QueriedObjects\" selection set."
                , ssQueriedObjectsLength);
```
} else {
    acutPrintf("\nNo objects were queried.");
}
else {
    acutPrintf("\nThe specified options variable has not been modified.");
}
acutRelRb(pSetPrefValRb);
int resultCode = acedSSFree(ssQueriedObjects);
ade_qrygetcond

Query Functions

Gets a condition of the current query.

```c
struct resbuf*
ade_qrygetcond(

    ade_id condition_id);
```

Returns a query condition or NULL.

`condition_id`  Query condition ID.

You must release the `resbuf`.

See [ade_qrydefine](#) for information about query conditions.

The following sample shows the use of `ade_qrygetcond()` only. A `resbuf` is created, the `resbuf` is filled based on the return of `ade_qrygetcond()`. Then it releases the `resbuf`, as required.

```c
struct resbuf* pNewQueryConditionRb = NULL;
pNewQueryConditionRb = ade_qrygetcond(queryId);
acutRelRb(pQueryConditionRb);
```

The following sample shows the use of `ade_qrygetcond()` in conjunction with `ade_qrydefine()`.

```c
char* pszJoinOperator = ""; // none
char* pszBgnCondGrouping = ""; // none
char* pszNotOperator = ""; // none
char* pszCondType = "Property";
char* pszEndCondGrouping = ""; // none
struct resbuf* pQueryConditionRb = acutBuildList(
    RTLB,
    RTSTR, "feature",
    RTSTR, "=",
    RTSTR, "Pond",
    RTLE,
    0);
```
After running the code, take notice of how the command line display matches the `ade_qrydefine()` parameter list.
ade_qrygetdwgandhandle

Query Functions

Gets the source drawing ID and original handle of a queried object.

```c
struct resbuf*
ade_qrygetdwgandhandle(
   ads_name ename);
```

Returns the drawing ID and handle for the queried object or NULL.

ename      AutoCAD entity name.

You must release the resbuf.

This function returns the ID of the source drawing from which the object was queried and the handle by which the object is known in that drawing.

The following sample creates a selection set of all entities in the current drawing using acedSSGet(). Each entity in the selection set is passed to ade_qrygetdwgandhandle() with all required parameters, the results of which populate a resbuf. The resbuf values are displayed and then released, as required.

```c
ads_name ename;
ads_name selectionSet;
struct resbuf* pQueriedObjInfoRb = NULL;
acedSSGet("_x", NULL, NULL, NULL, selectionSet);
long ssLength;
acedSSLLength( selectionSet, &ssLength);
for( int i = 0; i < ssLength; ++i )
{
   if( acedSSName(selectionSet, i, ename) == RTNORM )
   {
      pQueriedObjInfoRb = ade_qrygetdwgandhandle(ename);
      if(NULL != pQueriedObjInfoRb) {
         struct resbuf* rb = pQueriedObjInfoRb;
         while(NULL != rb) {
            switch(rb->restype)
            {
```
case RTREAL:
    acutPrintf("\n\nThe queried objects drawing Id is: %.0lf\n", rb->resval.real);
    break;

case RTSTR:
    acutPrintf("\nThe queried objects entity handle is: %s\n", rb->resval.rstring);
    break;

default:
    acutPrintf("\nCould not determine the value");
    break;

rb = rb->rbnext;
}
ade_qrygetentlist

Query Functions

Returns the list of entity handles for all objects that satisfy the current query in a specific drawing.

```c
struct resbuf*
ade_qrygetentlist(
    ade_id dwg_id);
```

Returns a list of the handles of selected objects or **NULL**.

**dwg_id**  Drawing ID of the drawing to query.

This function executes the current query and finds all objects that satisfy it in the drawing specified by the **dwg_id** argument. The entity handles of the objects are returned to the calling function.

Once you have the handle to an object, you can get the entity name with the **ade qryhandent** function and use it to perform other functions. For example, you could use **entget** (and **ads_entget**) to retrieve the entity and its definition data.

You must release the **resbuf**.

The following sample populates a **resbuf** with attached drawing id's using **ade_qllistctgy()**. This **resbuf** contains the input parameter used by **ade_qrygetentlist()** to populate a **resbuf** containing queried entity id's. Those id's are displayed, and the **resbufs**, are released as required.

```c
struct resbuf* pQueriedObjHandlesRb = NULL;
struct resbuf* pDsDwgIdRb = ade_dslist(ADE_NULLID, ADE_FALSE);
if (NULL != pDsDwgIdRb) {
    struct resbuf* rb = pDsDwgIdRb;
    while(NULL != rb) {
        struct resbuf* pQueriedObjHandlesRb = ade_qrygetentlist(rb->resval.rreal);
        if (NULL != pQueriedObjHandlesRb) {
            struct resbuf* rbHand = pQueriedObjHandlesRb;
            while(NULL != rbHand) {
                acutPrintf("This queried objects entity handle is: %s",
                    rbHand->resval.rstring);
                rbHand = rbHand->rbnext;
            }
        }
    }
}
```
rb = rb->rbnext;
}
}
acutPrintf("\nNo information could be retrieved.");
}
}
}
}
acutPrintf("\nNo queried objects were returned for this project.");
acutRelRb(pDsdwgIdRb);
acutRelRb(pQueriedObjHandlesRb);
ade_qrygetreptranform

Query Functions

Checks whether transformation is enabled for the current report query.

```c
int
ade_qrygetreptranform();
```

Returns **TRUE** or **FALSE**.

The following sample enables coordinate transformation for a report query using `ade_qrysetreptranform()`, then checks the status of the transformation flag using `ade_qrygetreptranform()`.

```c
ade_boolean bTxfrmForReportQry = ADE_TRUE;
int resultCode = ade_qrysetreptranform(bTxfrmForReportQry);
if (RTNORM == resultCode){
    acutPrintf(
        "\nReport query transformation has been set.");
}
else {
    acutPrintf(
        "\nNo transformation flag has been set.");
}
int nIsEnabled = ade_qrygetreptranform();
if (1 == nIsEnabled){
    acutPrintf(
        "\nReport query transformation is enabled.");
}
else {
    acutPrintf(
        "\nReport query transformation is not enabled.");
}
```
ade_qrygroup

Groups a sequence of two or more query conditions.

int
ade_qrygroup(
    ade_id condition_id1,
    ade_id condition_id2);

Returns RTNORM or an error code.

condition_id1      ID of first condition of the group.
condition_id2      ID of last condition of the group.

This function affects the current query.

A query definition consists of a sequence of query conditions. Within such a sequence, you can define subsequences of two or more conditions by grouping them (by enclosing them in parentheses). You can group conditions when you first define the query. See the bggroups and endgroups parameters of ade_qrydefine. Or you can do it later using ade_qrygroup.

When you call ade_qrygroup, the condition you specify as the first condition of the group (condition_id1) must be a predecessor to the one you specify as the last (condition_id2). The function groups the first and the last and any conditions in between.

The following sample defines a location condition which utilizes a buffer fence selection option. Three additional conditions are constructed based on SQL and the resulting query definition is displayed. Using ade_qrygroup() and the queryIds, (ownerQueryId and tele_UseQueryId) returned by ade_qrydefine(), conditions can be combined to produce a query which retrieves specific poles within a geographic area as displayed.

// Define the location condition.
char* pszJoinOperator      = ";"; // none
char* pszBgnCondGrouping   = ";"; // none
char* pszNotOperator       = ";"; // none
char* pszCondType          = "Location";
char* pszEndCondGrouping   = ";"; // none
ads_point bufferfencePt1;
bufferfencePt1[X] = 499375;
bufferfencePt1[Y] = 1451421;
ads_point bufferfencePt2;
bufferfencePt2[X] = 499383;
bufferfencePt2[Y] = 1451502;
ads_point bufferfencePt3;
bufferfencePt3[X] = 499422;
bufferfencePt3[Y] = 1451641;

struct resbuf* pLocQueryConditionRb = acutBuildList(
   RTLB,
    RTSTR, "Bufferfence",
    RTSTR, "Inside",
    RTREAL, 100.0,
    RTPOINT, bufferfencePt1,
    RTPOINT, bufferfencePt2,
    RTPOINT, bufferfencePt3,
    RTLE,
    0);

ade_id locQueryId = ade_qrydefine(
    pszJoinOperator,
    pszBgnCondGrouping,
    pszNotOperator,
    pszCondType,
    pLocQueryConditionRb,
    pszEndCondGrouping);

// Define the first SQL condition.
struct resbuf* pSQL_OwnerRb = acutBuildList(
    RTLB,
    RTSTR, "Pole",
    RTSTR, "OWNER = 'ELECTRIC'",
    RTLE,
    0);

ade_id OwnerQueryId = ade_qrydefine(
    "AND",
    "",
    "",
    "SQL",
    pSQL_OwnerRb,
    "");
// Define a second SQL condition.
struct resbuf* pSQLCable_UseRb = acutBuildList(
    RTLB,
    RTSTR, "Pole",
    RTSTR, "CABLE_USE = 'CABLECO'",
    RTLE,
    0);
ade_id Cable_UseQueryId = ade_qrydefine(
    "OR",
    "",
    "",
    "SQL",
    pSQLCable_UseRb,
    "")

// Define a third SQL condition.
struct resbuf* pSQLTele_UseRb = acutBuildList(
    RTLB,
    RTSTR, "Pole",
    RTSTR, "TELE_USE = 'TELCO'",
    RTLE,
    0);
ade_id Tele_UseQueryId = ade_qrydefine(
    "OR",
    "",
    "",
    "SQL",
    pSQLTele_UseRb,
    "")

The current query definition would appear as the following:

Location; INSIDE BUFFER FENCE

AND SQL:SELECT * FROM Pole WHERE OWNER = 'ELECTRIC'

OR SQL:SELECT * FROM Pole WHERE CABLE_USE = 'CABLECO'

OR SQL:SELECT * FROM Pole WHERE TELE_USE = 'TELCO'

int resultCode = ade_qrygroup(
    OwnerQueryId,
    Tele_UseQueryId);
After the call to ade_qrygroup(), the query definition would appear as the following:

Location; INSIDE BUFFER FENCE

AND (SQL:SELECT * FROM Pole WHERE OWNER = 'ELECTRIC')
OR SQL:SELECT * FROM Pole WHERE CABLE_USE = 'CABLECO'
OR SQL:SELECT * FROM Pole WHERE TELE_USE = 'TELCO')

To ungroup queries, use ade_qryungroup.
ade_qryhandent

Query Functions

Gets the entity name for the specified handle.

```c
int ade_qryhandent(  
    ade_id dwg_id,  
    char* handle,  
    ads_name result);
```

Returns RTNORM or an error code.

dwg_id    ID of the drawing in which the object resides.
handle    Original handle of the object in the specified drawing.
result    Output the entity name for the specified drawing ID and handle.

This function provides access to the entity name of an object in a source database.

You must use the retrieved entity name immediately before you call any other function (except ade_expreval) or return control to AutoCAD.

Once you have the entity name of an object, you can use it with other functions. For example, you could use entget (or ads_entget) to retrieve the entity and its definition data.

To get the original handle of the object in the source drawing, use the ade_qrygetentlist function.

To obtain a drawing ID, use ade_dslst.

To get the ID of a drawing given a drawing file path, use ade_dwggetid.

The following sample shows how you can combine ade_qrygetentlist and ade_qryhandent to count the number of objects in the source drawing(s) that are of type lwpolyline. First the current query defenition is cleared and a new location query is defined. The drawing ids for the attached drawings are obtained using ade_dslst(). Entity handles for objects which satisfy the current query are obtained from the active drawing using ade_qrygetentlist. The resbuf returned by ade_qrygetentlist is parsed and the entity name for each object is passed to acdbEntGet() which returns entity information that is checked for a type of LWPOLYLINE. A count of lwpolylines is displayed and all resbufs are released as required.
int resultCode = ade_qryclear();

char* pszJoinOperator = ""; // none
char* pszBgnCondGrouping = ""; // none
char* pszNotOperator = ""; // none
char* pszCondType = "Location";
char* pszEndCondGrouping = ""; // none

struct resbuf* pLocQueryConditionRb = acutBuildList(
    RTLB,
    RTSTR, "All",
    RTLE,
    0);
ade_id locQueryId = ade_qrydefine(
    pszJoinOperator,
    pszBgnCondGrouping,
    pszNotOperator,
    pszCondType,
    pLocQueryConditionRb,
    pszEndCondGrouping);

long lwPlineCount = 0;
struct resbuf* pSelectedObjHandlesRb = NULL;
struct resbuf* pEntdata = NULL;

struct resbuf* pAttachedDwgsRb = ade_dslist(ADE_NULLID, 1);
if (NULL != pAttachedDwgsRb) {
    struct resbuf* rb = pAttachedDwgsRb;
    while(NULL != rb) {
        if (ADE_TRUE == ade_dwgisactive(rb->resval.rreal)) {
            pSelectedObjHandlesRb = ade_qrygetentlist(rb->resval.rreal);
            if (NULL != pSelectedObjHandlesRb) {
                struct resbuf* rb1 = pSelectedObjHandlesRb;
                while(NULL != rb1) {
                    ads_name queriedEntity;
                    resultCode = ade_qryhandent(
                        rb->resval.rreal, // dwg_id
                        rb1->resval.rstring, // handle
                        queriedEntity);
                    pEntdata = acdbEntGet(queriedEntity);
                    while(NULL != pEntdata) {
                        if (0 == pEntdata->restype &&

(_tcscmp(pEntdata->resval.rstring, "LWPOLYLINE") == 0)) {

    lwPlineCount++;
    }
    pEntdata = pEntdata->rbnext;
    }
    rb1 = rb1->rbnext;
    }
}
acutPrintf(  
    "There are %d "LWPOLYLINES" in the active drawing set."
    , lwPlineCount);
} else{
    acutPrintf(  
    "The drawing was not active.");
    }
    rb = rb->rbnext;
    }
}
else {
    acutPrintf(  
    "There are no attached drawings in this project.");
}
acutRelRb(pAttachedDwgsRb);
acutRelRb(pSelectedObjHandlesRb);
acutRelRb(pEntdata);
ade_qrylist

Query Functions

Lists the IDs of the current query conditions.

```c
struct resbuf*
ade_qrylist();
```

Returns a list of the IDs of the current query conditions, or, if there is no current query, NULL.

You must release the `resbuf`.

The following sample populates a `resbuf` with query condition id's using `ade_qrylist()`. Those id's are displayed, and the `resbuf`, is released as required.

```c
struct resbuf* pQueryCondIdsRb = ade_qrylist();
if (NULL != pQueryCondIdsRb){
    struct resbuf* rb = pQueryCondIdsRb;
    while(NULL != rb) {
        acutPrintf("\nThe following query condition id's have been detected: %.0lf", rb->resval.rreal);
        rb = rb->rbnext;
    }
} else {
    acutPrintf("\nNo query conditions were returned for this project.");
}
acutRelRb(pQueryCondIdsRb);
```
ade_qrysave

Query Functions

Saves the current query.

ade_id
ade_qrysave(
    char* catname,
    struct resbuf* qryparams);

Returns the ID of the newly saved query or ADE_NULLID.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>catname</td>
<td>Category name. The category is created if it does not exist.</td>
</tr>
<tr>
<td>qryparams</td>
<td>A resbuf list composed of an information type and a value. See the Information Types table below.</td>
</tr>
</tbody>
</table>

Information Types

<table>
<thead>
<tr>
<th>name</th>
<th>Query name (RTSTR)</th>
</tr>
</thead>
<tbody>
<tr>
<td>description</td>
<td>Query description (RTSTR)</td>
</tr>
<tr>
<td>qtype</td>
<td>How the query is saved (RTSHORT): 1 = internal (default), 2 = external.</td>
</tr>
<tr>
<td>filename</td>
<td>For an external query, full path name (RTSTR).</td>
</tr>
<tr>
<td>saveoption</td>
<td>Bit code for the save options you are choosing (RTSHORT). See the Save Options table below.</td>
</tr>
</tbody>
</table>

The function saves the current query to the current drawing's query library or to a file.

- A query saved to the query library is called an internal query.
- A query saved to a file is called an external query.

You must specify a category name and a query name. In the current drawing and all attached drawings, no two queries can have the same name, even if they are saved in different categories. The default value for a description is the same as the query name. The default value for the storage type is internal. If you want to save the query externally, you must specify a file name for it.
**Save Options**

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Keep reference in query library.</td>
</tr>
<tr>
<td>2</td>
<td>Save list of active drawings.</td>
</tr>
<tr>
<td>4</td>
<td>Save location coordinates.</td>
</tr>
<tr>
<td>8</td>
<td>Save current property alteration definition.</td>
</tr>
<tr>
<td>16</td>
<td>Execute automatically.</td>
</tr>
</tbody>
</table>

A query gets a name and an ID only if it is referenced the query library. A new query that you have not yet saved does not have a name or an ID, and neither does an external query unless you keep a reference to it in the query library.

The following sample gets the location of the "QueryFileDirectory" using ade_prefgetval(). This path is combined with a query file name to create a parameter representing an external query. A resbuf is built which contains the name and description of an internal query plus the information required for the external query. Ade_qrysave() is called with with all required parameters, and the returned queryId is checked for value with an appropriate message displayed. All resbufs are then released, as required.

```c
char* pszOptionVar = "QueryFileDirectory";
struct resbuf* pGetPrefValRb = NULL;
pGetPrefValRb = ade_prefgetval(pszOptionVar);
CString sQueryFileName = "Hydro-210.qry";
CString sQueryFilePath = pGetPrefValRb->resval.rstring;
CString sQueryFileLoc = sQueryFilePath + sQueryFileName;
struct resbuf* pQuerySaveParamsRb = acutBuildList(
    RTLB,
    RTSTR, "name", RTSTR, "Hydro-210",
    RTDOTE,
    RTLB,
    RTSTR, "description", RTSTR, "Hydro-210 description",
    RTDOTE,
    RTLB,
    RTSTR, "qtype", RTSHORT, 2,
    RTDOTE,
    RTLB,
    RTSTR, "filename", RTSTR, sQueryFileLoc,
    RTDOTE,
    RTLB,
    RTSTR, "saveoption", RTSHORT, 3,
    RTDOTE,
    RTLE, 0 );
```
char* pszQueryCatgryName = "Hydrology";
ade_id queryId = ade_qrysavepszQueryCatgryName, pQuerySaveParamsRb);
if (queryId != ADE_NULLID) {
    acutPrintf(
        "\nA query has been saved to the internal library: \"%s\"\n",
        pszQueryCatgryName);
    acutPrintf(
        "\nAn external query has been saved as: %s\n",
        sQueryFileLoc);
} else {
    acutPrintf(
        "\nNo queries have been saved.\n";
}
acutRelRb(pGetPrefValRb);
acutRelRb(pQuerySaveParamsRb);
ade_qrysetaltprop

Query Functions

Turns property alteration on or off.

```c
int ade_qrysetaltprop(ade_boolean flag);
```

Returns RTNORM or an error code.

- **flag** Specifies whether property alteration is on or off: ADE_TRUE = on, ADE_FALSE = off.

This function affects the current query.

If there is no current property alteration definition, this function has no effect. To create a property alteration definition, use ade_altpdefine.
ade_qrysetcond

Query Functions

Replaces a query condition.

```c
int ade_qrysetcond(
    ade_id condition_id,
    struct resbuf* condition);
```

Returns RTNORM or an error code.

`condition_id` Query condition ID to replace.

`condition` New query condition (a list). See `ade_qrydefine`.

This function affects the current query.

You cannot alter grouping with this function. Any grouping you specify is ignored. To group or ungroup, use `ade_qrygroup` or `ade_qryungroup`.

The following sample defines a query using `ade_qrydefine()`. A `resbuf` is created which contains the replacement query condition. This `resbuf` and the query condition id returned by `ade_qrydefine()` are used as parameters to `ade_qrysetcond()`. The value returned by `ade_qrysetcond()` is checked against RTNORM and an appropriate message is displayed. The `resbuf` is then released as required.

```c
char* pszJoinOperator = ""; // none
char* pszBgnCondGrouping = ""; // none
char* pszNotOperator = ""; // none
char* pszCondType = "Property";
char* pszEndCondGrouping = ""; // none
struct resbuf* pQueryConditionRb = acutBuildList(
    RTLB,
    RTSTR, "layer",
    RTSTR, "=",
    RTSTR, "Pond",
    RTLE,
    0 );
```
ade_id queryCondId = ade_qrydefine(
    pszJoinOperator,
    pszBgnCondGrouping,
    pszNotOperator,
    pszCondType,
    pQueryConditionRb,
    pszEndCondGrouping);
if (queryCondId != ADE_NULLID) {
    struct resbuf* pSetQueryConditionRb = acutBuildList(
        RTLB,
        RTSTR, ",//joinop
        RTSTR, ",//bggroups
        RTSTR, ",//not_op
        RTSTR, "property",
        RTLB,
            RTSTR, "layer",
            RTSTR, ",=",
            RTSTR, "Water",
            RTLE,
        RTLE,
        0);
    int returnCode = ade_qrysetcond(queryCondId, pSetQueryConditionRb);
    if (RTNORM == returnCode) {
        acutPrintf(
            "\nThe query condition has been modified.\n");
    } else {
        acutPrintf(
            "\nThe query condition has not been modified.\n");
    }
    acutRelRb(pQueryConditionRb);
    acutRelRb(pSetQueryConditionRb);
}
Enables or disables transformation for the current report query.

```c
int ade_qrysetreptransform(
    ade_boolean flag);
```

Returns RTNORM or an error code.

**flag**  TRUE or FALSE, where TRUE = transformation enabled, and FALSE = transformation disabled.

The following sample enables coordinate transformation for a report query using `ade_qrysetreptransform()`.

```c
ade_boolean bTxfrmForReportQry = ADE_TRUE;
int resultCode = ade_qrysetreptransform(bTxfrmForReportQry);
if (RTNORM == resultCode){
    acutPrintf(
        "\nReport query transformation has been set.");
}
else {
    acutPrintf(
        "\nNo transformation flag has been set.");
}
```
ade_qrysettype

Query Functions

Sets the query mode: Preview, Draw, or Report.

```c
int ade_qrysettype(
    char* qrytype,
    ade_boolean multiline,
    char* templ,
    char* filename);
```

Returns RTNORM or an error code.

- **qrytype**: Query mode: "preview", "draw", or "report", where "preview" = Display queried objects without retrieving them, similar to Quick View, "draw" = Get queried objects from source drawings or external databases and copy to the current drawing, and "report" = Direct queried information to an output file.

- **multiline**: Whether to write report rows for sub-objects: TRUE or FALSE, where TRUE = Write report rows for sub-objects (objects such as vertices of polylines and attributes of blocks), and FALSE = Write lines for top-level objects only. Relevant only if qrytype is "report".

- **templ**: Ordered list of object properties to report. Each list element defines a report column. For example, ".type,.layer". Relevant only if qrytype is "report".

- **filename**: Path and file name of the output file. Relevant only if qrytype is "report".

This function affects the current query.

The three optional parameters, multiline, templ, and filename, are relevant only if the query mode is "report". If the query mode is "preview" or "draw", omit them.

The following example sets the query mode to Report using ade_qrysettype(). The templ, (pszReportTemplate) parameter represents output of a layer name, and two object data fields.

```c
char* pszQueryType = "Report";
char* pszReportTemplate = ".Layer,:AGE_30_49@ribgrp,:AMERI_ES@ribgrp";
char* pszReportFileName = "C:\\ADSRX\\QueryOutput\\Report_010404.txt";
ade_boolean bIsMultiline = ADE_FALSE;
```
int resultCode = ade_qrysettype(
    pszQueryType,
    bIsMultiline,
    pszReportTemplate,
    pszReportFileName);
if (RTNORM == resultCode){
    acutPrintf(
        "\nThe report query type has been set.");
}
else {
    acutPrintf(
        "\nUnable to set query type.");
}
Ungroups a sequence of two or more query conditions.

```c
int ade_qryungroup(
    ade_id condition_id1,
    ade_id condition_id2);
```

Returns RTNORM or an error code.

- **condition_id1**: Condition ID of the first grouped condition.
- **condition_id2**: Condition ID of the last grouped condition.

This function affects the current query.

A query definition consists of a sequence of query conditions. Within such a sequence, there can be subsequences that have been grouped by enclosing them in parentheses. Such groups may have been established when the query was first defined. See the `bggroups` and `endgroups` parameters of `ade_qrydefine`. Or they may have been established afterward by `ade_qrygroup`. However established, you can use `ade_qryungroup` to undo a group (remove its enclosing parentheses).

When you call `ade_qryungroup`, the condition you specify as the first of the group (`condition_id1`) must be a predecessor to the one you specify as the last (`condition_id2`). The function ungroups the first and the last and any conditions in between.

See the example used in `ade_qrygroup` for the before and after query definitions resulting from the use of `ade_qrygroup` and `ade_qryungroup`. 
ade_rtdefrange

Range Table Functions

Defines a range table.

ade_id
ade_rtdefrange(
    char* tabname,
    char* description,
    struct resbuf* range_defn);

Returns a range table ID or ADE_NULLID.

tabname           Range table name; can be up to 31 characters long. Must be unique, contain no spaces, and start with an alphanumeric character.
description      Range table description.
range_defn       Range table definition.

A range table allows you to alter properties of queried entities conditionally. It contains a set of property alteration values from which a single value is selected depending on conditions obtaining in the queried entity to be altered.

The range_defn argument is a range table definition, a list of range expressions. Each range expression includes (1) a condition and (2) a property alteration value to return if the condition is true. This information is expressed as a list of three elements: a range table operator and a comparison value (which together make up the condition), and the return value. You must state each value explicitly. You cannot substitute an expression.

See Using a Range Table for more information.

The following sample creates a resbuf containing the range expression values representing the table definition. ade_rtdefrange() is called with all required parameters, the returned rangeTableId is checked for ADE_NULLID and an appropriate message is displayed. Then it releases the resbuf, as required.

    char* pszrRangeTblName = "Range_ChgColor";
    char* pszrRangeTblDesc = "Change all except red to yellow";
    struct resbuf* pRangeTblDef = acutBuildList(RTLB,
ade_id rangeTableId = ade_rtdefrange(pszrRangeTblName, pszrRangeTblDesc, pRangeTblDef);
if (rangeTableId != ADE_NULLID) {
    acutPrintf("\nThe range table %s has been created.\n", pszrRangeTblName);
} else {
    acutPrintf("\nNo range table has been created.\n");
}
acutRelRb(pRangeTblDef);
ade_rtdeltable

*Range Table Functions*

Deletes a range table.

```c
int ade_rtdeltable(char* tablename);
```

Returns RTNORM or an error code.

**tablename** Range table name; can be up to 31 characters long. Must be unique, contain no spaces, and start with an alphanumeric character.
ade_rtgetid

Range Table Functions

Gets a range table ID.

```
ade_id
ade_rtgetid
       char* tablename);
```

Returns a range table ID or ADE_NULLID.

**tablename**

Range table name; can be up to 31 characters long. Must be unique, contain no spaces, and start with an alphanumeric character.
ade_rtgetprop

Range Table Functions

Gets the value of a range table property.

```
struct resbuf*
ade_rtgetprop(
    ade_id rt_id,
    char* property);
```

Returns a property value, or list, or NULL.

- **rt_id**: Range table ID.
- **property**: Property to get the value of. See the Range Table Properties table below.

You must release the `resbuf`.

### Range Table Properties

<table>
<thead>
<tr>
<th>name</th>
<th>Range table name.</th>
</tr>
</thead>
<tbody>
<tr>
<td>description</td>
<td>Range table description.</td>
</tr>
<tr>
<td>expr</td>
<td>Range table definition (list of range expressions).</td>
</tr>
</tbody>
</table>

See [ade_rtdefrange](#) for information about setting range table properties.

The following sample obtains a range table id using the sample code associated with [ade_rtdefrange](#).

[Aden_getprop](#) uses this range table id along with the specified property parameter and fills the `resbuf` with the corresponding value. If the operation is successful, the `resbuf` is parsed and the requested information is displayed. Then it releases the `resbufs`, as required.

```c
char* pszRangeTblName = "Range_ChgColor";
char* pszRangeTblDesc = "Change all except red to yellow";
struct resbuf* pRangeTblDef = acutBuildList(
    RTLB,
    RTLB,
    RTSTR, ",",
```
ade_id rangeTableId = ade_rtdereftable(ade_defrange(ade_defrange(pszrTableName, pszrTableDesc, pTableDef));

// Range Table has been defined.

struct resbuf* pRangeTblPropRb = NULL;
char* pszRangeTblPropVal = "description";
if (rangeTableId != ADE_NULLID) {
pRangeTblPropRb = ade_rangeprop(rangeTableId, pszRangeTblPropVal);
if (pRangeTblPropRb != NULL) {
    acutPrintf(
        "The range table \"%s\" property you requested is: \"%s\".
    , pszRangeTblPropVal, pRangeTblPropRb->resval.rstring);
}
else {
    acutPrintf(
        "No range table has been created.");
}
}

} else {
    acutPrintf(
"No range table has been created.");
}
acutRelRb(pRangeTblDef);
acutRelRb(pRangeTblPropRb);
ade_rtlist

Range Table Functions

Lists the IDs of all range tables defined in the project.

struct resbuf* 
ade_rtlist();

Returns a list of range table IDs or NULL.

You must release the resbuf.

The following sample parses the resbuf returned by ade_rtlist() and prints a list of range tables in the current project. Then it releases the resbuf, as required.

```c
struct resbuf* pRtListRb = NULL;
pRtListRb = ade_rtlist();
if (NULL != pRtListRb) {
    int nRangeTables = 0;
    struct resbuf* rb = pRtListRb;
    while(rb != NULL) {
        ++nRangeTables;
        acutPrintf("\nRange table %d has the Id: %.0lf", nRangeTables, rb->resval.rreal);
        rb = rb->rbnext;
    }
} else {
    acutPrintf("\nThere are no range tables in this project.");
}
acutRelRb(pRtListRb);
```
ade_saveobjs

Object Saving Functions

Saves objects queued for saving back to the source drawings.

```c
int ade_saveobjs(
    struct resbuf* priorities);
```

Returns RTNORM or an error code.

`priorities` List containing one to four save back operation codes, depending on the number of save back operations you are specifying. List the codes in order of their relative priority. See the Save-Back Operation Codes table below.

### Save-Back Operation Codes

<table>
<thead>
<tr>
<th>Code</th>
<th>Operation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Came From</td>
<td>Saves objects to their source drawings.</td>
</tr>
<tr>
<td>2</td>
<td>Selective</td>
<td>Saves objects you select to the drawings you specify.</td>
</tr>
<tr>
<td>3</td>
<td>Layer</td>
<td>Saves objects to layers in the source drawings that use the same names as those in which the objects lie.</td>
</tr>
<tr>
<td>4</td>
<td>Area</td>
<td>Saves objects to the source drawing within whose extents they lie, even if only partially within.</td>
</tr>
</tbody>
</table>

During the save back operation, the options are executed in the order specified in the `priorities` list. The list must contain at least one option. For example:

The following example creates a `resbuf` containing save-back operation code(s). `Ade_saveobjs()` uses this `resbuf` and returns RTNORM if successful. Status messages are displayed based on the returned code. Then it releases the `resbuf`, as required.

```c
struct resbuf* pSaveBackPriorityRb = acutBuildList(RTSHORT, 1, 0);
int returnCode = ade_saveobjs(pSaveBackPriorityRb);
if (RTNORM == returnCode) {
```
To save a selection set to a specific drawing, use `ade_savetodwg`. 
ade_savetodwg

Object Saving Functions

Saves a selection set to a specific drawing.

```c
int ade_savetodwg(
    ads_name sel_set,
    ade_id dwg_id);
```

Returns RTNORM or an error code.

- **sel_set**: Selection set name.
- **dwg_id**: Drawing ID of the destination drawing.

To save objects queued for saving back to the source drawings, use `ade_saveobjs`. 
ade_sqlgetenvstring

SQL Environment Functions

Gets a string describing the SQL environment.

char*
ade_sqlgetenvstring(
    char* linktemplate);

Returns a string describing the SQL environment or NULL.

linktemplate Link template.

For more information about link templates using SQL, see the AutoCAD online documentation.

The following sample gets the SQL environment string using ade_sqlgetenvstring() and displays that information based on the returned value.

```
char* pszLinkTemplateName = "BlockGroup";
char* pszSQLEnvironmentString = ade_sqlgetenvstring(pszLinkTemplateName);
if (NULL == pszSQLEnvironmentString ||
    _tcscmp(pszSQLEnvironmentString, "") == 0) {
    acutPrintf("
No environment string was obtained.");
} else {
    acutPrintf("\nThe current environment string is: \"%s\".",
                 pszSQLEnvironmentString);
}
ade_userget

User Security Functions

Gets the login name or entity lock name of the local user.

```c
int ade_userget(
    ade_boolean for_entity_locks,
    char* username);
```

Returns RTNORM or an error code.

- **for_entity_locks**: What to do if the local user is not logged into the application (optional): TRUE or FALSE, where TRUE = Get the user name used to identify the owner of object locks set locally, and FALSE = Return FALSE. Omitting this argument is the same as supplying FALSE.

- **username**: The user name or the empty string.

A user name can have as many as 32 characters.

If the local user is logged into the application, the user's application login name is used to identify the owner of object locks set locally. If the local user is not logged into the application, the user's operating system login name is used.

This function helps you determine if the owner of a particular object lock is the local user.
ade_usergetrights
User Security Functions

Gets the access rights of a user.

```c
int ade_usergetrights(
    char* userName,
    int* pUserRights);
```

Returns RTNORM or an error code.

- **username**        Login name, at most 32 characters.
- **pUserRights**     An integer equal to the sum of the rights allowed.

If the `userName` argument is **NULL**, the function returns the rights of the current user. The return value is a bit code for the rights allowed, as shown in the following table.

**User Rights Codes**

<table>
<thead>
<tr>
<th>Code</th>
<th>User Rights</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Superuser (in which case the other bits don't matter).</td>
</tr>
<tr>
<td>2</td>
<td>Permission to alter the drawing set.</td>
</tr>
<tr>
<td>4</td>
<td>Permission to edit objects.</td>
</tr>
<tr>
<td>8</td>
<td>Permission to execute a draw query.</td>
</tr>
<tr>
<td>16</td>
<td>Permission to edit Feature Class definition.</td>
</tr>
</tbody>
</table>

If the `username` argument is omitted or **NULL**, and there is no current user, the function returns a bit code with all bits set, because the no-current-user condition is possible only if the system option "ForceUserLogin" is set to **NULL**, in which case all users have all rights except those reserved for a superuser.

Only a superuser can specify a login name other than their own. If the `username` argument is not the login
name of the current user, and the current user does not have superuser rights, the function returns `NULL`, and the message "Access is denied" is added to the error stack.
ade_userlist

User Security Functions

Lists the current users.

\texttt{struct resbuf*}
\texttt{ade_userlist();}

Returns a list of user login names or \texttt{NULL}.

The following sample parses the \texttt{resbuf} returned by \texttt{ade_userlist()} and prints a list of user names. Then it releases the \texttt{resbuf}, as required.

\begin{verbatim}
struct resbuf* pUserListRb = NULL;
pUserListRb = ade_userlist();
if (NULL != pUserListRb) {
    while (pUserListRb != NULL) {
        pUserListRb = pUserListRb->rbnext;
        acutPrintf("\nUser\t-\t%s", pUserListRb->resval);
    }
} else {
    acutPrintf("\nUser list could not be created");
}
acutRelRb(pUserListRb);
\end{verbatim}
ade_userset

User Security Functions

Logs in a user.

int
ade_userset(
    char* username,
    char* password);

Returns RTNORM or an error code

username Login name or NULL.
password Password or NULL

If either argument is omitted, the User Login dialog box displays. If a login name was specified, it appears in the dialog's Login Name field.

If both arguments are specified, but the user cannot be logged in, one of the following messages is added to the error stack:

    Invalid user name.
    Invalid password.

If the drawing set includes active drawings containing locked entities, the current user cannot be changed. If you attempt to log in a different user under those conditions, the following error message is added to the error message stack:

    Cannot login again when drawings are locked/active.
ade_userSetRights

User Security Functions

Sets the access rights for a user.

```c
int ade_userSetRights(  
    char* userName,  
    int userRights);  
```

Returns RTNORM or an error code.

- **userName**: Login name.
- **userRights**: An integer equal to the sum of the rights to allow. See the User Rights Codes table below.

### User Rights Codes

<table>
<thead>
<tr>
<th>Code</th>
<th>User Rights</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Superuser (in which case the other bits don't matter).</td>
</tr>
<tr>
<td>2</td>
<td>Permission to alter the drawing set.</td>
</tr>
<tr>
<td>4</td>
<td>Permission to edit objects.</td>
</tr>
<tr>
<td>8</td>
<td>Permission to execute a draw query.</td>
</tr>
<tr>
<td>16</td>
<td>Permission to edit Feature Class definition.</td>
</tr>
</tbody>
</table>

This function cannot execute unless the current user has superuser rights, and it cannot change the rights of the current user in any case. If an ordinary user is logged in when this function is called, or a superuser is logged in and the function call would change the rights of the current user, the function returns an error code of -1002, and the following message is added to the error message stack:

  Access denied
ade_version

Other Functions

Gets the version number of the Data Extension programming interface.

```c
struct resbuf*ade_version();
```

Returns a version number (string) or NULL.

You must release the `resbuf`.

The following sample parses the `resbuf` returned by `ade_version()` and prints the version number. Then it releases the `resbuf`, as required.

```c
struct resbuf* pVersionRb = NULL;
pVersionRb = ade_version();
if (NULL != pVersionRb) {
    acutPrintf("\nThe current version is: %s", pVersionRb->resval);
} else {
    acutPrintf("\nThe version could not be determined");
}
acutRelRb(pVersionRb);
```
The functions for drawing management begin with `ade_dwg`.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ade_dwgactivate</code></td>
<td>Activates a drawing.</td>
</tr>
<tr>
<td><code>ade_dwgactualpath</code></td>
<td>Returns the full path of a drawing.</td>
</tr>
<tr>
<td><code>ade_dwgaliaspath</code></td>
<td>Returns the alias path of a drawing.</td>
</tr>
<tr>
<td><code>ade_dwgattriblist</code></td>
<td>Returns a list of attribute tags for the specified block name.</td>
</tr>
<tr>
<td><code>ade_dwgdeactivate</code></td>
<td>Deactivates a drawing.</td>
</tr>
<tr>
<td><code>ade_dwggetid</code></td>
<td>Gets the drawing ID of a drawing.</td>
</tr>
<tr>
<td><code>ade_dwggetsetting</code></td>
<td>Gets a drawing setting value.</td>
</tr>
<tr>
<td><code>ade_dwghaslocks</code></td>
<td>Checks if a drawing has locked objects.</td>
</tr>
<tr>
<td><code>ade_dwgindex</code></td>
<td>Applies specified index operations to a drawing.</td>
</tr>
<tr>
<td><code>ade_dwgindexdef</code></td>
<td>Specifies which indexes are to be created or removed.</td>
</tr>
<tr>
<td><code>ade_dwgisactive</code></td>
<td>Checks if a drawing is active.</td>
</tr>
<tr>
<td><code>ade_dwgistoplevel</code></td>
<td>Checks if a drawing is directly attached to the project drawing.</td>
</tr>
<tr>
<td><code>ade_dwgproplist</code></td>
<td>Lists all values found in a drawing for a given drawing property.</td>
</tr>
<tr>
<td><code>ade_dwgquickview</code></td>
<td>Displays a quick view of a drawing.</td>
</tr>
<tr>
<td><code>ade_dwgselectdlg</code></td>
<td>Displays the Select Drawings dialog box.</td>
</tr>
<tr>
<td><code>ade_dwgsetof</code></td>
<td>Identifies the drawings to which a given drawing is attached.</td>
</tr>
<tr>
<td><code>ade_dwgsetsetting</code></td>
<td>Sets a drawing setting value.</td>
</tr>
<tr>
<td><code>ade_dwgunlock</code></td>
<td>Removes all object locks from a drawing.</td>
</tr>
<tr>
<td><code>ade_dwgzoomextents</code></td>
<td>Zooms to the extents of the active drawings.</td>
</tr>
</tbody>
</table>
The functions for drawing set management begin with ade_ds.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ade_dsattach</td>
<td>Attaches a drawing to the project.</td>
</tr>
<tr>
<td>ade_dsdetach</td>
<td>Detaches a drawing from the project.</td>
</tr>
<tr>
<td>ade_dsisnested</td>
<td>Checks if a drawing has nested drawings.</td>
</tr>
<tr>
<td>ade_dlist</td>
<td>Lists the drawings attached to a given drawing.</td>
</tr>
<tr>
<td>ade_dsproplist</td>
<td>Lists all values found in the drawing set for a given drawing property.</td>
</tr>
</tbody>
</table>
Drive Alias Functions

Data Extension Function Synopsis

The drive alias functions begin with ade_alias.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ade_aliasadd</td>
<td>Creates a drive alias.</td>
</tr>
<tr>
<td>ade_aliasdelete</td>
<td>Deletes a drive alias.</td>
</tr>
<tr>
<td>ade_aliasgetlist</td>
<td>Lists all drive aliases in the project.</td>
</tr>
<tr>
<td>ade_aliasupdate</td>
<td>Assigns a new drive and path to a drive alias.</td>
</tr>
</tbody>
</table>
The functions for handling error messages begin with ade_err.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ade_errclear</td>
<td>Clears the error stack.</td>
</tr>
<tr>
<td>ade_errcode</td>
<td>Gets the error code for a specific error on the error stack.</td>
</tr>
<tr>
<td>ade_errgetlevel</td>
<td>Gets the system error level.</td>
</tr>
<tr>
<td>ade_errmsg</td>
<td>Gets the error message for a specific error on the error stack.</td>
</tr>
<tr>
<td>ade_errpush</td>
<td>Pushes an error message to the stack.</td>
</tr>
<tr>
<td>ade_errpushstatement</td>
<td>Pushes a statement to the stack.</td>
</tr>
<tr>
<td>ade_errqty</td>
<td>Returns the number of error messages on the stack.</td>
</tr>
<tr>
<td>ade_errsetlevel</td>
<td>Sets the system error level.</td>
</tr>
<tr>
<td>ade_errshowdlg</td>
<td>Displays the Map Messages dialog box, which shows a list of error messages on the stack.</td>
</tr>
<tr>
<td>ade_errstatement</td>
<td>Gets the erroneous statement for a specific error on the stack.</td>
</tr>
<tr>
<td>ade_errtype</td>
<td>Gets the type of a specific error in the stack.</td>
</tr>
</tbody>
</table>
Expression Evaluation Function

Data Extension Function Synopsis

The expression evaluation function begins with ade_exp.

| ade_expreval | Evaluates an AutoCAD Map expression. |
The functions for object data management begin with `ade_od`.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ade_odaddfield</code></td>
<td>Adds fields to a table.</td>
</tr>
<tr>
<td><code>ade_odaddrecord</code></td>
<td>Attaches data to an object.</td>
</tr>
<tr>
<td><code>ade_odattachrecord</code></td>
<td>Attaches a new record to an object.</td>
</tr>
<tr>
<td><code>ade_oddefinetab</code></td>
<td>Creates an object data table.</td>
</tr>
<tr>
<td><code>ade_oddeletefield</code></td>
<td>Deletes fields from a table.</td>
</tr>
<tr>
<td><code>ade_oddeletetab</code></td>
<td>Deletes a table.</td>
</tr>
<tr>
<td><code>ade_oddelrecord</code></td>
<td>Deletes a record.</td>
</tr>
<tr>
<td><code>ade_odfreerec</code></td>
<td>Frees the memory claimed in defining a new record.</td>
</tr>
<tr>
<td><code>ade_odgetfield</code></td>
<td>Gets a field value.</td>
</tr>
<tr>
<td><code>ade_odgetrecfield</code></td>
<td>Gets a field value using a record ID.</td>
</tr>
<tr>
<td><code>ade_odgetrecord</code></td>
<td>Gets a record ID.</td>
</tr>
<tr>
<td><code>ade_odgettables</code></td>
<td>Lists the tables attached to an object.</td>
</tr>
<tr>
<td><code>ade_odmodifyfield</code></td>
<td>Modifies field properties in a table.</td>
</tr>
<tr>
<td><code>ade_odmodifytab</code></td>
<td>Redefines a table.</td>
</tr>
<tr>
<td><code>ade_odnewrecord</code></td>
<td>Defines a new object data record.</td>
</tr>
<tr>
<td><code>ade_odpresetfield</code></td>
<td>Assigns a value to a field in a new record.</td>
</tr>
<tr>
<td><code>ade_odrecordqty</code></td>
<td>Counts the records attached to an object.</td>
</tr>
<tr>
<td><code>ade_odsetfield</code></td>
<td>Sets a field value.</td>
</tr>
<tr>
<td><code>ade_odtabledefn</code></td>
<td>Gets a table definition.</td>
</tr>
<tr>
<td>ade_odtablelist</td>
<td>Lists the tables in the project.</td>
</tr>
</tbody>
</table>
The functions for object editing begin with `ade_edit`.

<table>
<thead>
<tr>
<th>Function</th>
<th>Synopsis</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ade_editdefcen</code></td>
<td>Defines a new label point for an object.</td>
</tr>
<tr>
<td><code>ade_editlockederased</code></td>
<td>Gets the objects in the save set that have been erased.</td>
</tr>
<tr>
<td><code>ade_editislocked</code></td>
<td>Gets lock information about an object if it is locked.</td>
</tr>
<tr>
<td><code>ade_editlocked</code></td>
<td>Gets the objects in the save set that have been modified.</td>
</tr>
<tr>
<td><code>ade_editlockobjs</code></td>
<td>Locks a set of objects and adds them to the save set.</td>
</tr>
<tr>
<td><code>ade_editnew</code></td>
<td>Gets the objects in the save set that are new.</td>
</tr>
<tr>
<td><code>editunlockobjs</code></td>
<td>Unlocks a set of objects and removes them from the save set.</td>
</tr>
</tbody>
</table>
## Object Saving Functions

### Data Extension Function Synopsis

The object saving functions begin with `ade_save`.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ade_saveobjs</code></td>
<td>Saves objects queued for saving back to the source drawings.</td>
</tr>
<tr>
<td><code>ade_savetodwg</code></td>
<td>Saves a selection set to a specific drawing.</td>
</tr>
</tbody>
</table>
The option functions begin with ade_pref.

<table>
<thead>
<tr>
<th>Function</th>
<th>Synopsis</th>
</tr>
</thead>
<tbody>
<tr>
<td>ade_prefgetval</td>
<td>Gets an option setting.</td>
</tr>
<tr>
<td>ade_prefsetval</td>
<td>Sets an option.</td>
</tr>
</tbody>
</table>
The functions for specifying how objects retrieved in a query should be altered begin with `ade_altp`.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ade_altpclear</code></td>
<td>Clears the current property alteration definition.</td>
</tr>
<tr>
<td><code>ade_altpdefine</code></td>
<td>Creates a property alteration expression.</td>
</tr>
<tr>
<td><code>ade_altpdelprop</code></td>
<td>Deletes a property alteration expression.</td>
</tr>
<tr>
<td><code>ade_altpgetprop</code></td>
<td>Gets a property alteration expression.</td>
</tr>
<tr>
<td><code>ade_altplist</code></td>
<td>Lists the IDs of the current property alteration expressions.</td>
</tr>
<tr>
<td><code>ade_altpsetprop</code></td>
<td>Modifies a property alteration expression.</td>
</tr>
</tbody>
</table>
Query Functions

**Data Extension Function Synopsis**

The functions for query management begin with `ade_qry`.

<table>
<thead>
<tr>
<th>Function</th>
<th>Synopsis</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ade_qryclear</code></td>
<td>Clears the current query.</td>
</tr>
<tr>
<td><code>ade_qrydefine</code></td>
<td>Defines a query.</td>
</tr>
<tr>
<td><code>ade_qryexecute</code></td>
<td>Executes the current query.</td>
</tr>
<tr>
<td><code>ade_qrygetcond</code></td>
<td>Gets a query condition.</td>
</tr>
<tr>
<td><code>ade_qrygetdwgandhandle</code></td>
<td>Gets the source drawing ID and original handle of a queried object.</td>
</tr>
<tr>
<td><code>ade_qrygetentlist</code></td>
<td>Returns entity handles for objects that satisfy the current query.</td>
</tr>
<tr>
<td><code>ade_qrygetrepretransform</code></td>
<td>Checks whether transformation is enabled for the current report query.</td>
</tr>
<tr>
<td><code>ade_qrygroup</code></td>
<td>Groups query conditions.</td>
</tr>
<tr>
<td><code>ade_qryhandent</code></td>
<td>Gets the entity name for the specified handle.</td>
</tr>
<tr>
<td><code>ade_qrylist</code></td>
<td>Lists the IDs of the current query conditions.</td>
</tr>
<tr>
<td><code>ade_qrysave</code></td>
<td>Saves the current query.</td>
</tr>
<tr>
<td><code>ade_qrysetaltprop</code></td>
<td>Turns property alteration on or off.</td>
</tr>
<tr>
<td><code>ade_qrysetcond</code></td>
<td>Modifies a query condition.</td>
</tr>
<tr>
<td><code>ade_qrysetrepretransform</code></td>
<td>Enables or disables transformation for the current report query.</td>
</tr>
<tr>
<td><code>ade_qrysettype</code></td>
<td>Sets the query mode: Preview, Draw, or Report.</td>
</tr>
<tr>
<td><code>ade_qryungroup</code></td>
<td>Ungroups query conditions.</td>
</tr>
</tbody>
</table>
The functions for query library management begin with `ade_ql`.

<table>
<thead>
<tr>
<th>Function</th>
<th>Synopsis</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ade_qldelctgy</code></td>
<td>Deletes a query library category.</td>
</tr>
<tr>
<td><code>ade_qldelquery</code></td>
<td>Deletes a query from the query library.</td>
</tr>
<tr>
<td><code>ade_qlgctgyinfo</code></td>
<td>Gets information about a query category.</td>
</tr>
<tr>
<td><code>ade_qlgryinfo</code></td>
<td>Gets information about a query.</td>
</tr>
<tr>
<td><code>ade_qllistctgy</code></td>
<td>Lists the query category IDs.</td>
</tr>
<tr>
<td><code>ade_qlloadqry</code></td>
<td>Makes a saved query current.</td>
</tr>
<tr>
<td><code>ade_qlqrgygid</code></td>
<td>Gets a query ID.</td>
</tr>
<tr>
<td><code>ade_qlsetctgyname</code></td>
<td>Changes a query category name.</td>
</tr>
<tr>
<td><code>ade_qlsetquery</code></td>
<td>Changes a query name, description, or the category it belongs to.</td>
</tr>
</tbody>
</table>
The functions for range table management begin with `ade_rt`.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ade_rtdfrange</code></td>
<td>Defines a range table.</td>
</tr>
<tr>
<td><code>ade_rtdetable</code></td>
<td>Deletes a range table.</td>
</tr>
<tr>
<td><code>ade_rtgetid</code></td>
<td>Gets a range table ID.</td>
</tr>
<tr>
<td><code>ade_rtgetprop</code></td>
<td>Gets the value of a range table property.</td>
</tr>
<tr>
<td><code>ade_rlist</code></td>
<td>Lists the IDs of all range tables defined in the project.</td>
</tr>
</tbody>
</table>
The SQL environment functions begin with `ade_sql` or `ade_key`.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ade_keycolumnlist</code></td>
<td>Returns a list of the key column names for the specified link path name.</td>
</tr>
<tr>
<td><code>ade_sqlgetenvstring</code></td>
<td>Gets a string describing the SQL environment.</td>
</tr>
</tbody>
</table>
User Security Functions

Data Extension Function Synopsis

The user security functions begin with `ade_user`.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ade_userget</code></td>
<td>Gets the login name or entity lock name of the local user.</td>
</tr>
<tr>
<td><code>ade_usergetrights</code></td>
<td>Gets the access rights of the specified user.</td>
</tr>
<tr>
<td><code>ade_userlist</code></td>
<td>Lists the current users.</td>
</tr>
<tr>
<td><code>ade_userset</code></td>
<td>Logs in a user.</td>
</tr>
<tr>
<td><code>ade_usersetrights</code></td>
<td>Sets the access rights for the specified user.</td>
</tr>
</tbody>
</table>
Other Functions

Data Extension Function Synopsis

Miscellaneous data extension functions.

<table>
<thead>
<tr>
<th>Function</th>
<th>Synopsis</th>
</tr>
</thead>
<tbody>
<tr>
<td>ade_entsetlocation</td>
<td>Sets a new text-label point for a drawing object.</td>
</tr>
<tr>
<td>ade_osfexpand</td>
<td>Searches a directory and returns a list of file names.</td>
</tr>
<tr>
<td>ade_version</td>
<td>Gets the version number of the Data Extension API.</td>
</tr>
</tbody>
</table>
Enumerates annotation expression fields.

```cpp
enum eAnnotationExpressionFields {
    kUndefinedField = 0x00,
    kAttDefAnnotationString = 0x01,
    kAttDefTextStyle = 0x02,
    kAttDefTextHeight = 0x03,
    kAttDefTextLayer = 0x04,
    kAttDefTextColor = 0x05,
    kAttDefTextRotation = 0x06,
    kAttDefTextJustification = 0x07,
    kAttDefTextLineWidth = 0x08,
    kAttDefTextTrueColor = 0x09,
    kBlockPosition = 0x10,
    kBlockRotation = 0x20,
    kBlockScaleFactor = 0x30,
    kBlockColor = 0x40,
    kBlockLayer = 0x50,
    kBlockLinetype = 0x60,
    kBlockLineWidth = 0x70,
    kBlockAssocEnt = 0x80,
    kBlockTrueColor = 0x90
};
```

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Enumerates annotation insert precedence modes.

```cpp
def isInsertPrecedenceMode {
    kExpressionPriority = 0,
    kOverridePriority
}
```

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns the object ID of the entity that an annotation reference references.

\[
\text{AcDbObjectId AnnotationBlockReferenceAssociatedObjectId(}
\text{\quad const AcDbBlockReference* pBlkRef)}
\]

Returns the object ID of the associated entity, or \text{AcDbObjectId::kNull} if an error occurs.

Created with a commercial version of \text{Doc-O-Matic}. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at \text{support@toolsfactory.com}. 
Creates a new annotation text object in a template.

```c
Acad::ErrorStatus CreateAnnotationText(
    AcDbObjectId & newAnnotationTextId,
    AcDbObjectId templateId
);
```

### Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>newAnnotationTextId</td>
<td>Output object ID of the newly created annotation text object.</td>
</tr>
<tr>
<td>templateId</td>
<td>Input object ID of the annotation template to add the annotation text object to.</td>
</tr>
</tbody>
</table>

### Returns

Returns Acad::eOk if successful; otherwise, returns an error code.

### Remarks

The new annotation text object will have default property values (that is, no expression overrides). At minimum, you should set the new object's annotation string (kAttDefAnnotationString expression) immediately after creation, typically with SetExpressionString().

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Attaches an annotation reference to an associated entity.

```cpp
Acad::ErrorStatus InsertAnnotationReference(
    AcDbObjectId& newBlockReferenceId,
    AcDbObjectId templateOrBlockId,  
    const AcDbObjectId assocEnt,     
    const AcDbBlockReference * pMatchThisBlockReference = NULL
);
```

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>newBlockReferenceId</td>
<td>Input/Output object ID of the newly created annotation reference. If newBlockReferenceId is AcDbObjectId::kNull, then a new annotation reference is created. If not, then the AcDbBlockReference whose Object ID is newBlockReferenceID is reused.</td>
</tr>
<tr>
<td>templateOrBlockId</td>
<td>Input object ID of an annotation template's block table record or a regular AutoCAD block table record to attach.</td>
</tr>
<tr>
<td>assocEnt</td>
<td>Input object ID of the entity to which the reference will be attached.</td>
</tr>
<tr>
<td>pMatchThisBlockReference</td>
<td>Input optional pointer to an existing annotation reference, whose static-property and expression-string overrides are used to create the new annotation reference.</td>
</tr>
</tbody>
</table>

### Returns

Returns Acad::eOk if successful; otherwise, returns an error code.
This function takes a template or block table record ID and an associated entity object ID. The new block reference object ID is an output parameter. If the caller provides an optional pointer to an existing annotation reference, that reference's static-property and expression-string overrides are used to create the new annotation reference.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Attaches an annotation reference to an associated entity.

```cpp
Acad::ErrorStatus InsertAnnotationReference(
    AcDbObjectId& newBlockReferenceId,
    AcDbObjectId templateOrBlockId,
    const AcDbObjectId assocEnt,
    const AnnotationOverrides & overrides
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>newBlockReferenceId</td>
<td>Input/Output object ID of the newly created annotation reference. If newBlockReferenceId is AcDbObjectId::kNull, then a new annotation reference is created. If not, then the AcDbBlockReference whose Object ID is newBlockReferenceID is reused.</td>
</tr>
<tr>
<td>templateOrBlockId</td>
<td>Input object ID of an annotation template's block table record or a regular AutoCAD block table record to attach.</td>
</tr>
<tr>
<td>assocEnt</td>
<td>Input object ID of the entity to which the reference will be attached.</td>
</tr>
<tr>
<td>overrides</td>
<td>Input AnnotationOverrides struct containing the static-property and expression-string overrides to be used when creating the new annotation reference. In the AnnotationOverrides structure, pass a NULL for each individual property that you do not want to override; instead, these property values will be computed from the template's corresponding expression or static property.</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns an error code.
Remarks

This function takes a template or block table record ID and an associated entity object ID. The new block reference object ID is an output parameter. The static-property and expression-string overrides of the input structure are used to create the new annotation reference (unless a NULL prevents an override).

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Attaches an annotation reference to one or more associated entities.

```
Acad::ErrorStatus InsertAnnotationReferences(
   AcDbObjectIdArray& newBlockReferenceIds,
   AcDbObjectId templateOrBlockId,
   const AcDbObjectIdArray& assocEntArray,
   const AcDbBlockReference * pMatchThisBlockReference = NULL
);
```

File
AcMapAnnotationManager.h

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>newBlockReferenceIds</td>
<td>Output array of object IDs of the newly created annotation references.</td>
</tr>
<tr>
<td>templateOrBlockId</td>
<td>Input object ID of an annotation template's block table record or a regular AutoCAD block table record to attach.</td>
</tr>
<tr>
<td>assocEntArray</td>
<td>Input array of object IDs of the entities to which the references will be attached.</td>
</tr>
<tr>
<td>pMatchThisBlockReference</td>
<td>Input optional pointer to an existing annotation reference, whose static-property and expression-string overrides are used to create the new annotation reference.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns an error code.

Remarks

This function takes a template or block table record ID and an array of associated entity object IDs. An array of new block reference object IDs is an output parameter. If the caller provides an optional pointer to an existing annotation reference, that reference's static-property and expression-string
overrides are used to create the new annotation references.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determines whether a block reference is an annotation reference.

```cpp
bool IsAnnotationBlockReference(
    const AcDbBlockReference * pBlkRef
);
```

**Parameters**

- **pBlkRef**
  - Input block reference to examine.

**Returns**

- Returns true if the block reference is an annotation reference; otherwise, returns false.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determined whether an attribute definition is an annotation text entity.

```cpp
bool IsAnnotationText(
    const AcDbAttributeDefinition * pAttDef
);
```

File

`AcMapAnnotationManager.h`

**Parameters**

- `pAttDef`: Input attribute definition to examine.

**Returns**

Returns true if the attribute definition is an annotation text entity; otherwise, returns false.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links

AcMapAnnotationManager Namespace
AcMapAnnotationManager:: SetAnnotationInsertPrecedenceMode Function
AcMapAnnotationManager Namespace

Sets the annotation insert precedence mode.

AcMapAnnotationManager::eInsertPrecedenceMode SetAnnotationInsertPrecedenceMode
AcMapAnnotationManager::eInsertPrecedenceMode newPrecedence
);

File

AcMapAnnotationManager.h

Parameters

Description

newPrecendence
New annotation insert precendence mode to set.

Returns

Returns the previous precendence mode.

Remarks

Annotation references may give expressions priority (legacy behavior) or may give overrides priority. The previous insert mode is returned.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets an expression string to store with an annotation template, annotation reference, or annotation text object.

\begin{verbatim}
Acad::ErrorStatus SetExpressionString(
    AcMapExpression & pExpression,
    AcDbObject* pObj,
    AcMapAnnotationManager::eAnnotationExpressionFields field
);
\end{verbatim}

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pExpression</td>
<td>Input expression to store with the object.</td>
</tr>
<tr>
<td>pObj</td>
<td>Input pointer to the AcDbBlockTableRecord (template), AcDbBlockReference (reference), or AcDbAttributeDefinition (text) to store the expression string with.</td>
</tr>
<tr>
<td>field</td>
<td>Input <code>eAnnotationExpressionFields</code> value of the expression to store. For annotation templates and annotation references, use only kBlock* codes; for annotation text entities, use only kAttDef* codes.</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns an error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Returns the prefix that is prepended to all annotation block table records internally.

```cpp
const ACHAR* TemplateNameBTRPrefix();
```

FILE

`AcMapAnnotationManager.h`

Returns the internal prefix prepended to all annotation block table records.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Structure that holds all static-property and expression-string overrides used for creating or updating annotations.

```c
struct AnnotationOverrides {
    const double * pdRotationOverride;
    const double * pdScaleFactorOverride;
    const AcCmColor * pColorOverride;
    const ACHAR * pszLayerOverride;
    const ACHAR * pszLinetypeOverride;
    const AcDb::LineWeight * pLineweightOverride;
    const ACHAR * pszRotationExpressionOverride;
    const ACHAR * pszScaleFactorExpressionOverride;
    const ACHAR * pszColorExpressionOverride;
    const ACHAR * pszLayerExpressionOverride;
    const ACHAR * pszLinetypeExpressionOverride;
    const ACHAR * pszLineweightExpressionOverride;
    const ACHAR * pszPositionExpressionOverride;
};
```

File

AcMapAnnotationManager.h

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pdRotationOverride</td>
<td>An override block rotation value, or NULL to override.</td>
</tr>
<tr>
<td>pdScaleFactorOverride</td>
<td>An override block scale factor value, or NULL not to override.</td>
</tr>
<tr>
<td>pColorOverride</td>
<td>An override block color value, or NULL not to override.</td>
</tr>
<tr>
<td>pszLayerOverride</td>
<td>An override block layer value, or NULL not to override.</td>
</tr>
<tr>
<td>pszLinetypeOverride</td>
<td>An override block linetype value, or NULL not to override.</td>
</tr>
<tr>
<td>pLineweightOverride</td>
<td>An override block lineweight value, or NULL not to override.</td>
</tr>
</tbody>
</table>
pszRotationExpressionOverride  An override block rotation expression, or NULL not to override.
pszScaleFactorExpressionOverride  An override block scale factor expression, or NULL not to override.
pszColorExpressionOverride  An override block color expression, or NULL not to override.
pszLayerExpressionOverride  An override block layer expression, or NULL not to override.
pszLinetypeExpressionOverride  An override block linetype expression, or NULL not to override.
pszLineweightExpressionOverride  An override block lineweight expression, or NULL not to override.
pszPositionExpressionOverride  An override block position expression, or NULL not to override.

Remarks

Set a parameter to NULL if you do not want to override the default or existing value of that property.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

virtual ~AcMapClassificationManager();

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Constructs an instance of this class by using the specified AutoCAD database.

```
AcMapClassificationManager(
    AcDbDatabase * pDb
);
```

Parameters
- **pDb**: Input AutoCAD database.

Returns
- Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Examines an entity for out-of-range or missing classified-property values, fixing these values if desired.

```
AcMapObjClass::EErrCode Audit(
    AcDbObjectId& entId,
    bool bFixOutOfRange,
    bool bFixMissingProperties
);
```

Parameters

<table>
<thead>
<tr>
<th>Description</th>
<th>Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input ID of the entity to examine.</td>
<td>entId</td>
</tr>
<tr>
<td>Input true to reset each out-of-range property to its default value, or false to not reset these properties.</td>
<td>bFixOutOfRange</td>
</tr>
<tr>
<td>Input true to add each missing property to the entity, or false to not add these properties.</td>
<td>bFixMissingProperties</td>
</tr>
</tbody>
</table>

Returns

Returns `AcMapObjClass::EErrCode` eOk if successful. Returns `AcMapObjClass::EErrCode` eNoSchemaFileAttached if no feature-definition file is attached to the current drawing. Returns `AcMapObjClass::EErrCode` eEntityNotClassified if the entity is unclassified. Returns `AcMapObjClass::EErrCode` eOutOfRange if at least one property value is out of range. This error code is returned only if bFixOutOfRange is true and an out-of-range property could not be fixed. If the range of the Layer property specifies a layer that does not exist in the current drawing, for example, attempting to fix this nonexistent layer will fail. Returns `AcMapObjClass::EErrCode` eMissingProperty if the entity is missing at least one property. This error code is returned only if bFixMissingProperties is true and a missing property could not be added to the entity. If a missing object-data property refers to a table and column with the same name but different type as the one in the current drawing, for example, the type mismatch prevents adding the property and setting its value. Returns `AcMapObjClass::EErrCode` eFailed if the process failed for some other reason.

Created with a commercial version of Doc-O-Matic. In order to make this
message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Examines multiple entities for out-of-range or missing classified-property values, listing and fixing these values if desired.

```c++
AcMapObjClass::EErrCode Audit(
    AcDbObjectIdArray* paEntIdsOutOfRange,
    AcDbObjectIdArray* paEntIdsMissingProperties,
    AcDbObjectIdArray* paEntIdsNotClassified,
    AcDbObjectIdArray& aEntIds,
    bool bFixOutOfRange,
    bool bFixMissingProperties
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>paEntIdsOutOfRange</td>
<td>Output entity IDs of out-of-range properties, or NULL if not needed by the caller.</td>
</tr>
<tr>
<td>paEntIdsMissingProperties</td>
<td>Output entity IDs of missing properties, or NULL if not needed by the caller.</td>
</tr>
<tr>
<td>paEntIdsNotClassified</td>
<td>Output IDs of unclassified entities. or NULL if not needed by the caller.</td>
</tr>
<tr>
<td>aEntIds</td>
<td>Input IDs of the entities to examine.</td>
</tr>
<tr>
<td>bFixOutOfRange</td>
<td>Input true to reset each out-of-range property to its default value, or false to not reset these properties.</td>
</tr>
<tr>
<td>bFixMissingProperties</td>
<td>Input true to add each missing property to the entity, or false to not add these properties.</td>
</tr>
</tbody>
</table>

Returns

- `AcMapObjClass::EErrCode eOk` if successful.
- `AcMapObjClass::EErrCode eNoSchemaFileAttached` if no feature-definition file is attached to the current drawing.
- `AcMapObjClass::EErrCode eFailed` if the process failed for at least one entity.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Determines whether the current user has sufficient privileges to change the classification information in the feature-definition file.

```cpp
bool CanCurrentUserAlterSchema() const;
```

Returns true if the user has adequate privileges; otherwise, returns false.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Unclassifies an entity entirely. Unclassifying entities does not trigger internal transactions. This function differs from Unclassify() because it ignores the feature-definition file that was used to classify the entity.

```cpp
AcMapObjClass::EErrCode ClearAllTags(AcDbObjectId& entId);
```

**Parameters**

entId Input ID of the entity to unclassify entirely.

**Returns**

Returns `AcMapObjClass::EErrCode` eOk if successful. Returns `AcMapObjClass::EErrCode` eAlreadyUnclassified if the entity is already unclassified. Returns `AcMapObjClass::EErrCode` eFailed if the process failed for some other reason.

Created with a commercial version of [Doc-O-Matic](https://toolsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Unclassifies multiple entities entirely. Unclassifying entities does not trigger internal transactions. This function differs from Unclassify() because it ignores the feature-definition file that was used to classify the entities.

```cpp
AcMapObjClass::EErrCode ClearAllTags(
    AcDbObjectIdArray* paAlreadyClearedIds,
    AcDbObjectIdArray& aEntIds
);
```

**Parameters**

- `paAlreadyClearedIds` Output IDs of the entities that had no classification information at all, or NULL if not needed by the caller.
- `aEntIds` Input IDs of the entities to unclassify entirely.

**Returns**

- Returns `AcMapObjClass::EErrCode eOk` if all entities are cleared successfully.
- Returns `AcMapObjClass::EErrCode eAlreadyUnclassified` if at least one entity is already unclassified. Returns `AcMapObjClass::EErrCode eFailed` if the process failed for at least one entity.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Detaches the current feature-definition file from the current drawing.

AcMapObjClass::EErrCode  DetachCurrentFeatureDefinitionFile();

Returns

Returns AcMapObjClass::EErrCode eOk if successful. Returns AcMapObjClass::EErrCode eFailed if the process failed.

Remarks

This function succeeds even if no feature-definition file is attached to the current drawing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Lists all the classification tags of an entity.

**AcMapObjClass::EErrCode** GetAllTags(
    AcMapStringArray & aTagNames,
    AcMapStringArray & aSchemaNames,
    const AcDbObjectId & entId
) const;

**Parameters**
- **aTagNames**
  - Output array of feature class names.
- **aSchemaNames**
  - Output array of feature-definition file names. The array index of each feature-definition file name matches the index of each corresponding feature class name in aTagNames.
- **entId**
  - Input ID of the entity to examine.

**Returns**
- Returns **AcMapObjClass::EErrCode** eOk if successful. Returns **AcMapObjClass::EErrCode** eEntityNotClassified if the entity is unclassified. Returns **AcMapObjClass::EErrCode** eFailed if the process failed for some other reason.

**Remarks**

If an entity was classified multiple times by using different feature-definition files, this function retrieves all the entity's feature class names.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Lists all the classified entities in the current drawing.

```c++
bool GetClassifiedEntities(
    AcDbObjectIdArray& aEntIds
) const;
```

Parameters  

- **aEntIds**: Output IDs of classified entities.

Returns

Returns true if at least one entity ID is returned; otherwise, returns false if no entity ID is returned or the process failed.

Remarks

An entity is considered to be classified even if the corresponding feature class definition is not in the attached feature-definition file.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Counts the number of feature class definitions in the feature-definition file attached to the current drawing.

```cpp
AcMapObjClass::EErrCode GetFeatureClassDefinitionCount(
  long* plCount
) const;
```

Parameters

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>plCount</td>
</tr>
</tbody>
</table>

Returns

Returns `AcMapObjClass::EErrCode eOk` if successful. Returns `AcMapObjClass::EErrCode eNoSchemaFileAttached` if no feature-definition file is attached to the current drawing. Returns `AcMapObjClass::EErrCode eFailed` if the process failed for some other reason.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Lists all the feature class definition names in the feature-definition file attached to the current drawing.

```cpp
AcMapObjClass::EErrCode GetFeatureClassNames(
    AcMapStringArray& aClassNames
) const;
```

**Parameters**

- `aClassNames`: Output array of class definition names.

**Returns**

- Returns `AcMapObjClass::EErrCode` `eOk` if successful.
- Returns `AcMapObjClass::EErrCode` `eNoSchemaFileAttached` if no feature-definition file is attached to the current drawing.
- Returns `AcMapObjClass::EErrCode` `eFailed` if the process failed for some other reason.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at `support@toolsfactory.com`.

---

Links

- AcMapClassificationManager Class, AcMapClassificationManager Class
- AcMapClassificationManager:: GetFeatureClassNames Method
- AcMapClassificationManager Class | AcMapClassificationManager Class
Lists all the properties and their values of an entity.

```
AcMapObjClass::EErrCode GetProperties(
    AcArray<AcMapObjClassProperty>* aProperties,
    AcArray<VARIANT>* paValues,
    const AcDbEntity * pEntity
) const;
```

### Parameters

**aProperties**
Output array of `AcMapObjClassProperty` properties. The caller must free this object.

**paValues**
Output array of property values, or NULL if not needed by the caller. The array index of each property value matches the index of each corresponding property in `aProperties`. The name, category, and type properties echo the entity's current state. The default value is set to the property's current value. The read-only property is set to false. The visibility property is set to true. The range property is not set. The caller must free this object.

**pEntity**
Input entity to examine.

### Returns

Returns `AcMapObjClass::EErrCode` `eOk` if successful. Returns `AcMapObjClass::EErrCode` `eFailed` if the process failed for some other reason.

### Remarks

Use this function to find all properties of a specific entity and decide which one to classify in the corresponding feature class definition. If you are not using the properties to update a feature class definition, this function can run without classification requirements (such as a feature-definition file or class name).

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems
registering this software please contact us at support@toolsfactory.com.
Lists all the properties and their values of an entity.

```cpp
AcMapObjClass::EErrCode GetProperties(
    AcArray<AcMapObjClassProperty>* aProperties,
    AcArray<VARIANT>* paValues,
    const AcDbObjectId& entId
) const;
```

Parameters

- **aProperties**: Output array of `AcMapObjClassProperty` properties. The caller must free this object.
- **paValues**: Output array of property values, or NULL if not needed by the caller. The array index of each property value matches the index of each corresponding property in `aProperties`. The name, category, and type properties echo the entity's current state. The default value is set to the property's current value. The read-only property is set to false. The visibility property is set to true. The range property is not set. The caller must free this object.
- **entId**: Input ID of the entity to examine.

Returns

Returns `AcMapObjClass::EErrCode` `eOk` if successful. Returns `AcMapObjClass::EErrCode` `eFailed` if the process failed for some other reason.

Remarks

Use this function to find all properties of a specific entity and decide which one to classify in the corresponding feature class definition. If you are not using the properties to update a feature class definition, this function can run without classification requirements (such as a feature-definition file or class name).

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems
registering this software please contact us at support@toolsfactory.com.
Lists all the unclassified entities in the current drawing.

```cpp
bool GetUnclassifiedEntities(
    AcDbObjectIdArray& aEntIds
) const;
```

**Parameters**
- `aEntIds` Output IDs of all unclassified entities.

**Returns**

Returns true if at least one entity ID is returned; otherwise, returns false if no entity ID is returned or the process failed.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Lists all the undefined entities in the current drawing.

```cpp
bool GetUndefinedEntities(
    AcDbObjectIdArray& aEntIds
) const;
```

Parameters
- **aEntIds**

Description
Output IDs of all undefined entities.

Returns
- Returns true if at least one entity ID is returned; otherwise, returns false if no entity ID is returned or the process failed.

Remarks

A classified entity is undefined if it lacks a corresponding feature class definition in the attached feature-definition file, or if it was classified with a different feature-definition file.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at **support@toolsfactory.com**.
Links
AcMapClassificationManager Class, AcMapClassificationManager Class
AcMapClassificationManager::ReloadCurrentFeatureDefinitionFile Method
AcMapClassificationManager Class | AcMapClassificationManager Class

Reloads the feature-definition file attached to the current drawing.

AcMapObjClass::EErrCode ReloadCurrentFeatureDefinitionFile();

Returns

Returns AcMapObjClass::EErrCode eOk if successful. Returns AcMapObjClass::EErrCode eNoSchemaFileAttached if no feature-definition file is attached to the current drawing. Returns AcMapObjClass::EErrCode eFailed if the process failed for some other reason.

Remarks

Use this function with caution. The feature-definition file's data and file name are cached in memory, so if a user is modifying the file while this function executes, it may reload outdated data while the file and memory are unsynchronized (prior to the user saving the file). In particular, this function will fail if the user renames or deletes the feature-definition file.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapClassificationManager Class, AcMapClassificationManager Class
AcMapClassificationManager:: SaveCurrentFeatureDefinitionFile Method
AcMapClassificationManager Class | AcMapClassificationManager Class

Saves the feature-definition file attached to the current drawing.

AcMapObjClass::EErrCode SaveCurrentFeatureDefinitionFile() const;

Returns
AcMapObjClass::EErrCode eOk if successful. Returns
AcMapObjClass::EErrCode eNoSchemaFileAttached if no feature-definition file is attached to the current drawing. Returns AcMapObjClass::EErrCode eFailedSavingSchema if the feature-definition file could not be saved. Returns AcMapObjClass::EErrCode eNoUserPrivilegeToAlterSchema if the current user lacks the privileges to change the feature-definition file. Returns AcMapObjClass::EErrCode eFailed if the process failed for some other reason.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Unclassifies an entity. Unclassifying entities does not trigger internal transactions. Unclassify only entities classified with the currently attached feature-definition file. If you unclassify an entity that was classified with a different feature-definition file, the corresponding tags will not be changed. To remove all existing tags, regardless of which feature-definition file was used, use ClearAllTags().

`AcMapObjClass::EErrCode Unclassify(
   AcDbObjectId& entId
);`

Parameters
entId Input ID of the entity to unclassify.

Returns

Returns `AcMapObjClass::EErrCode` eOk if successful. Returns `AcMapObjClass::EErrCode` eNoSchemaFileAttached if no feature-definition file is attached to the current drawing. Returns `AcMapObjClass::EErrCode` eAlreadyUnclassified if the entity is already-unclassified. Returns `AcMapObjClass::EErrCode` eFailed if the process failed for some other reason.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Unclassifies multiple entities. Unclassifying entities does not trigger internal transactions. Unclassify only entities classified with the currently attached feature-definition file. If you unclassify an entity that was classified with a different feature-definition file, the corresponding tags will not be changed. To remove all existing tags, regardless of which feature-definition file was used, use ClearAllTags().

```
AcMapObjClass::EErrCode Unclassify(
    AcDbObjectIdArray* paAlreadyUnclassifiedIds,
    AcDbObjectIdArray& aEntIds
);
```

- **Parameters**
  - `paAlreadyUnclassifiedIds` Output IDs of already-unclassified entities, or NULL if not needed by the caller.
  - `aEntIds` Input IDs of the entities to unclassify.

**Returns**

Returns `AcMapObjClass::EErrCode` `eOk` if all entities are declassified successfully. Returns `AcMapObjClass::EErrCode` `eNoSchemaFileAttached` if no feature-definition file is attached to the current drawing. Returns `AcMapObjClass::EErrCode` `eFailed` if the process failed for at least one entity.

Created with a commercial version of *Doc-O-Matic*. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

```cpp
virtual ~AcMapObjClassDefinition( void );
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td>Void.</td>
</tr>
</tbody>
</table>

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Add a new property to this feature class definition.

\[
\text{AcMapObjClass}:\text{EErrCode} \text{ AddProperty(}
\text{   const AcMapObjClassProperty} \& \text{ property})
\]

**Parameters**

- **property**: Input property to add.

**Returns**

Returns \text{AcMapObjClass}:\text{EErrCode} \text{ eOk} if successful. Returns \text{AcMapObjClass}:\text{EErrCode} \text{ eClassNotFromCurrentSchema} if the class is not from the current feature-definition file. Returns \text{AcMapObjClass}:\text{EErrCode} \text{ ePropertyAlreadyExists} if the property already exists. Returns \text{AcMapObjClass}:\text{EErrCode} \text{ eFailed} if the process failed for some other reason.

Created with a commercial version of \text{Doc-O-Matic}. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the full pathname of the feature-definition file that this feature class definition belongs to.

```cpp
const ACHAR* GetFeatureDefinitionFile() const;
```

Returns the feature-definition file's full pathname.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Retrieves the name of this feature class definition.

```cpp
const ACHAR* GetName() const;
```

Returns

Returns the class name.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links

AcMapObjClassDefinition Class, AcMapObjClassDefinition Class
AcMapObjClassDefinition:: GetProperties Method
AcMapObjClassDefinition Class | AcMapObjClassDefinition Class

Lists all the classified properties of this feature class definition.

AcMapObjClass::EErrCode GetProperties( 
        AcArray&lt;AcMapObjClassProperty*&gt; & aProperties 
    ) const;

Parameters

aProperties            Description
Output list of AcMapObjClassProperty classified properties. The caller must free this object.

Returns

Returns AcMapObjClass::EErrCode eOk if successful. Returns
AcMapObjClass::EErrCode eClassNotFromCurrentSchema if the class is not from the current feature-definition file. Returns AcMapObjClass::EErrCode ePropertyNotFound if none classified. Returns AcMapObjClass::EErrCode eFailed if the process failed for some other reason.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Lists the AutoCAD entity types that this feature class definition supports.

**AcMapObjClass::EErrCode** GetSupportedEntityTypes(
    AcArray<AcRxClass*>& aEntityTypes,
    AcMapStringArray& aBlockNames
) const;

Parameters | Description
--- | ---
aEntityTypes | Output array of supported entity types.
aBlockNames | Output array of supported block names.

Returns

Returns **AcMapObjClass::EErrCode** eOk if the class supports at least one entity type. Returns **AcMapObjClass::EErrCode** eClassNotFromCurrentSchema if the class is not from the current feature-definition file. Returns **AcMapObjClass::EErrCode** eRxClassNotFound if an entity type was found but the corresponding AcRxClass was not found in the class dictionary. Returns **AcMapObjClass::EErrCode** eFailed if no entity type is supported or if the process failed for some other reason.

Remarks

The list typically matches the value that AcRxObject::desc() returns. A feature class definition can support any block reference, or specific block reference names. For any block reference, the corresponding entity type AcDbBlockReference::desc() is output in aEntityTypes. For specific block reference names, the names are output in aBlockNames.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapObjClassDefinition Class, AcMapObjClassDefinition Class
AcMapObjClassDefinition:: GetSupportedEntityTypes Method
AcMapObjClassDefinition Class | AcMapObjClassDefinition Class

Lists the AutoCAD entity types that this feature class definition supports.

**AcMapObjClass::EErrCode** GetSupportedEntityTypes(
    AcMapStringArray& aEntityTypes,
    AcMapStringArray& aBlockNames
) const;

Parameters

<table>
<thead>
<tr>
<th>Description</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>aEntityTypes</td>
<td>Output array of supported entity types.</td>
</tr>
<tr>
<td>aBlockNames</td>
<td>Output array of supported block names.</td>
</tr>
</tbody>
</table>

Returns

Returns **AcMapObjClass::EErrCode** eOk if the class supports at least one entity type. Returns **AcMapObjClass::EErrCode** eClassNotFromCurrentSchema if the class is not from the current feature-definition file. Returns **AcMapObjClass::EErrCode** eFailed if no entity type is supported or if the process failed for some other reason.

Remarks

The list typically matches the value that AcRxClass::name() returns. A feature class definition can support any block reference, or specific block reference names. For any block reference, the corresponding entity type AcDbBlockReference::desc()->name() is output in aEntityTypes. For specific block reference names, the names are output in aBlockNames.

Created with a commercial version of [Doc-O-Matic](http://docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determines whether this feature class definition is a strict base class.

```cpp
AcMapObjClass::EErrCode IsBaseClassOnly(
    bool* pbIsBaseOnly
) const;
```

**Parameters**
- `pbIsBaseOnly`: Output true if the feature class definition is a strict base class; otherwise, false.

**Returns**
- Returns `AcMapObjClass::EErrCode` `eOk` if successful. Returns `AcMapObjClass::EErrCode` `eClassNotFromCurrentSchema` if the class is not from the current feature-definition file. Returns `AcMapObjClass::EErrCode` `eFailed` if the process failed for some other reason.

**Remarks**

You can derive other classes from a strict base class but not classify objects with it.

Created with a commercial version of *Doc-O-Matic*. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determines whether this feature class definition is visible in the AutoCAD Map project workspace.

```cpp
AcMapObjClass::EErrCode IsVisibleInWorkspace(
    bool* pbIsVisible
) const;
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pbIsVisible</td>
<td>Output true if the feature class definition is visible, or false if it is invisible.</td>
</tr>
</tbody>
</table>

Returns

Returns `AcMapObjClass::EErrCode` `eOk` if successful. Returns `AcMapObjClass::EErrCode` `eClassNotFromCurrentSchema` if the class is not from the current feature-definition file. Returns `AcMapObjClass::EErrCode` `eFailed` if the process failed for some other reason.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
This function allows a client program to find out whether a link to the external data is going to be preserved to the FDO or the whole record is conveyed to the FDO.

\texttt{AcMapObjClass::EErrCode \texttt{LinkedDataMovedToFdo}\left(\texttt{bool \& bDataMove}\right)\texttt{const};}

Parameters

- \texttt{bDataMove}: Output value is true if the only link to external data preserved to FDO. Output is false if the actual data is preserved to the FDO.

Returns

Returns \texttt{AcMapObjClass::eOk} if successful. Returns \texttt{AcMapObjClass::eClassNotFromCurrentSchema} if class not from current schema. Returns \texttt{AcMapObjClass::eFailed} if fails for any reason.

Created with a commercial version of \texttt{Doc-O-Matic}. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at \texttt{support@toolsfactory.com}. 
This function allows a client program to define the behavior of the linked data in process of querying and saveing from/to FDO. A link to the external data only will be preserved to the FDO if the boolean parameter is set to true. Otherwise, records will be moved to the FDO. conveyed to the FDO.

```cpp
AcMapObjClass::EErrCode SetLinkedDataMovedToFdo(
    bool bDataMove
);
```

**Parameters**

bDataMove

**Description**

Input value. True if only link to external data preserved to FDO. False if the actual data is preserved to the FDO.

**Returns**

Returns AcMapObjClass::eOk if successful. Returns AcMapObjClass::eClassNotFromCurrentSchema if class not from current schema. Returns AcMapObjClass::eFailed if fails for any reason.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets this feature class definition to visible or invisible in the AutoCAD Map project workspace.

```cpp
AcMapObjClass::EErrCode SetVisibleInWorkspace(
    bool bVisible
);
```

Parameters  
---

**bVisible**  
Input true for visible or false for invisible.

Returns  

- Returns `AcMapObjClass::EErrCode eOk` if successful.  
- Returns `AcMapObjClass::EErrCode eClassNotFromCurrentSchema` if the class is not from the current feature-definition file.  
- Returns `AcMapObjClass::EErrCode eFailed` if the process failed for some other reason.

Remarks  

A visible class will not appear in the project workspace if one of its parent classes is invisible.

Created with a commercial version of [Doc-O-Matic](https://toolsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

```
virtual ~AcMapObjClassProperty();
```

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the direct category of this property.

```c++
void GetCategory(
    AcMapStringArray& aCategories
) const;
```

**Parameters**
- `aCategories`: Output category.

**Returns**
- Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://doc-o-matic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the default value of this property.

```cpp
bool GetDefaultValue(
    VARIANT * pvarDefault
) const;
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pvarDefault</td>
<td>Output default value. The caller must free this object, typically with VariantClear().</td>
</tr>
</tbody>
</table>

Returns

Returns true if successful; otherwise, returns false.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the name of this property.

```cpp
const ACHAR* GetName() const;
```

Returns the property's name.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links

AcMapObjClassProperty Class, AcMapObjClassProperty Class

AcMapObjClassProperty:: GetRange Method

AcMapObjClassProperty Class | AcMapObjClassProperty Class

Retrieves the range of valid values for this property.

`const ACHAR* GetRange() const;`

Returns

Returns the property's range of valid values, or "--" if the range has not been set with SetRange().

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links

AcMapObjClassProperty Class, AcMapObjClassProperty Class
AcMapObjClassProperty:: GetType Method
AcMapObjClassProperty Class | AcMapObjClassProperty Class

Retrieves the type of this property.

VARTYPE GetType() const;
Returns

Returns the property's type.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the value of this property from the specified entity.

```cpp
AcMapObjClass::EErrCode GetValue(
    VARIANT * pvarValue,
    const AcDbObjectId& entId
) const;
```

**Parameters**

<table>
<thead>
<tr>
<th>Description</th>
<th>pvarValue</th>
<th>entId</th>
</tr>
</thead>
<tbody>
<tr>
<td>Output value of the property.</td>
<td>Input object ID of the entity whose property value to get.</td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

Returns `AcMapObjClass::EErrCode` eOk if successful. Returns `AcMapObjClass::EErrCode` ePropertyNotFound if the property is not present on the entity. Returns `AcMapObjClass::EErrCode` eFailed if the process failed for some other reason.

**Remarks**

The entity and property can be classified or unclassified.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determine whether a value falls in this property's range of valid values.

```cpp
bool IsInRange(
    VARIANT * pVar
);
```

**Parameters**

- `pVar`: Input value to examine.

**Returns**

Returns true if the value is in the range, or false if it is out of range or the range has not been set with SetRange().

Created with a commercial version of [Doc-O-Matic](#). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapObjClassProperty Class, AcMapObjClassProperty Class
AcMapObjClassProperty:: IsReadOnly Method
AcMapObjClassProperty Class | AcMapObjClassProperty Class

Determines whether this property is read-only or read-write.

```cpp
bool IsReadOnly() const;
```

Returns

Returns true if the property is read-only, or false if it is read-write.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determines whether this property is visible or invisible.

```cpp
bool IsVisible() const;
```

Returns true if the property is visible, or false if it is invisible.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the default value of this property.

```c
AcMapObjClass::EErrCode SetDefaultValue(
    VARIANT * pvarDefault
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pvarDefault</td>
<td>Input default value.</td>
</tr>
</tbody>
</table>

Returns `AcMapObjClass::EErrCode` `eOk` if successful. Returns `AcMapObjClass::EErrCode` `eInvalidType` if the type is invalid. Returns `AcMapObjClass::EErrCode` `eClassNotFromCurrentSchema` if the property's class is not in the current feature-definition file. Returns `AcMapObjClass::EErrCode` `eOutOfRange` if the value falls outside the valid range. Returns `AcMapObjClass::EErrCode` `eFailed` if the process failed for some other reason.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets this property to read-only or read-write.

\[\text{AcMapObjClass::EErrCode SetReadOnly(} \text{bool bReadOnly)};\]

Parameters Description
bReadOnly Input true to set the property to read-only, or false to set it to read-write.

Returns

Returns \text{AcMapObjClass::EErrCode eOk} if successful. Returns \text{AcMapObjClass::EErrCode eClassNotFromCurrentSchema} if the property's class is not in the current feature-definition file. Returns \text{AcMapObjClass::EErrCode eFailed} if the process failed for some other reason.

Created with a commercial version of \text{Doc-O-Matic}. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at \text{support@toolsfactory.com}.
Sets the value of this property for the specified entity. The entity and property can be classified or unclassified. If this property is classified, this function checks the property's range of valid values and, if the value is invalid, sets the value to its default value if bFixOutOfRangeIfClassified is true. Properties retrieved with \texttt{AcMapClassificationManager::GetClassifiedProperties()} (twoforms) are classified properties. Properties retrieved with \texttt{AcMapClassificationManager::GetProperties()} (twoforms) are unclassified properties, and SetValue() sets property values regardless of the value of bFixOutOfRangeIfClassified.

\begin{verbatim}
AcMapObjClass::EErrCode SetValue(
    const AcDbObjectId& entId,
    const VARIANT * pvarValue,
    bool bFixOutOfRangeIfClassified
);
\end{verbatim}

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>entId</td>
<td>Input object ID of the entity whose property value to set.</td>
</tr>
<tr>
<td>pvarValue</td>
<td>Input value to set the property to.</td>
</tr>
<tr>
<td>bFixOutOfRangeIfClassified</td>
<td>Input true to reset an out-of-range classified-property value to its default value, or false not to reset it.</td>
</tr>
</tbody>
</table>

Returns \texttt{AcMapObjClass::EErrCode} eOk if successful. Returns \texttt{AcMapObjClass::EErrCode} ePropertyNotFound if the entity does not have the specified property. Returns \texttt{AcMapObjClass::EErrCode} ePropertyReadOnly if the property is a read-only property, or if the property is classified and \texttt{AcMapObjClassProperty::IsReadOnly()} is true. Returns \texttt{AcMapObjClass::EErrCode} eOutOfRange if the property value is still out of range. This error code is returned only if bFixOutOfRangeIfClassified is true, the property is classified, and the out-of-range property value could not be reset.

For example, if the range of the Layer property specifies a layer that does not
exist in the current drawing, an attempt to fix this nonexistent layer will fail. Returns AcMapObjClass::EErrCode eFailed if the process failed for some other reason.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets this property to visible or invisible.

\[ \text{AcMapObjClass::EErrCode} \text{ SetVisible}( \text{bool} \ bVisible); \]

Parameters

\begin{tabular}{|l|l|}
\hline
bVisible & Description \\
\hline
\end{tabular}

Input true to set the property to visible, or false to set it to invisible.

Returns

Returns \text{AcMapObjClass::EErrCode} eOk if successful. Returns \text{AcMapObjClass::EErrCode} eClassNotFromCurrentSchema if the property's class is not in the current feature-definition file. Returns \text{AcMapObjClass::EErrCode} eFailed if the process failed for some other reason.

Created with a commercial version of \text{Doc-O-Matic}. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns the string representation of a value.

```
ACHAR* ToString(
    const VARIANT* pvarValue
) const;
```

Parameters

- `pvarValue` Input value.

Returns

Returns the string value, or NULL if the type is invalid. The caller must free this object, typically with acutDelString().

Remarks

The value's type must be the same as that of the property.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

```cpp
virtual ~AcMapObjClassReactor(
    void
);
```

Parameters

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td>Void.</td>
</tr>
</tbody>
</table>

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Constructs an instance of this class.

```
AcMapObjClassReactor(
    void
);
```

Parameters Description
void Void.

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapObjClassSystem Class, AcMapObjClassSystem Class
AcMapObjClassSystem::: RemoveObjClassReactor Method
AcMapObjClassSystem Class | AcMapObjClassSystem Class

Removes a reactor from the list of the classification reactors.

```cpp
bool RemoveObjClassReactor(
    AcMapObjClassReactor* pReactor
);
```

Parameters

**pReactor**

Description

Input *AcMapObjClassReactor* reactor.

Returns

Returns true if successful. Returns false if the reactor was not found in the list of reactors.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Destroys an instance of this class.

```cpp
virtual ~AcMapObjClassSystem(
    void
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td>Void.</td>
</tr>
</tbody>
</table>

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Constructs an instance of this class.

```
AcMapObjClassSystem(
    void
);
```

Parameters Description
void Void.

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapObjClassSystem Class, AcMapObjClassSystem Class
AcMapObjClassSystem:: AddObjClassReactor Method
AcMapObjClassSystem Class | AcMapObjClassSystem Class

Adds a reactor to the list of the classification reactors.

```cpp
bool AddObjClassReactor(
    AcMapObjClassReactor* pReactor
);
```

Parameters | Description
--- | ---
pReactor | Input AcMapObjClassReactor reactor.

Returns

Returns true if successful. Returns false if the reactor is already active.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapGeometryPE Class
AcMapGeometryPE:: EGeometryType Enumeration
AcMapGeometryPE Class

Types for simple geometry elements.

```cpp
enum EGeometryType {
    kUnsupported = 0,
    kPoint = 1,
    kPolyline = 2,
    kPolygon = 3
};
```

File

AcMapGeometryPE.h

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapGeometryPE Class, AcMapGeometryPE Class
AcMapGeometryPE:: beginRead Method
AcMapGeometryPE Class | AcMapGeometryPE Class

Executes before reading an entity begins.

**virtual void** beginRead(
    AcDbEntity * pEnt
);

Parameters Description
pEnt The entity.

Returns

Returns nothing.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Executes after reading an entity ends.

```cpp
virtual void endRead(
    AcDbEntity * pEnt
);
```

Parameters Description

pEnt The entity.

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Gets the type of a simple element.

```cpp
virtual AcMapGeometryPE::EGeometryType getGeometryType(
    AcDbEntity * pEnt,
    unsigned int geomIndex
);
```

**Parameters**

- `pEnt`: The entity containing the element.
- `geomIndex`: The element index.

**Returns**

Returns the type.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Counts simple elements in an entity.

```cpp
virtual unsigned int getNumGeometries(
    AcDbEntity * pEnt
);
```

Parameters | Description
--- | ---
pEnt | The entity.

Returns

Returns the count.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Counts vertices of a polyline or polygon.

```cpp
virtual unsigned intgetNumVertices(
    AcDbEntity * pEnt,
    unsigned int geomIndex
);
```

Parameters

- **pEnt**: The entity containing the element.
- **geomIndex**: The element index.

Returns

Returns the count.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapGeometryPE Class, AcMapGeometryPE Class
AcMapGeometryPE::getPoint Method
AcMapGeometryPE Class | AcMapGeometryPE Class

Gets a point element.

virtual Acad::ErrorStatus getPoint(
    AcGePoint3d & p,
    AcDbEntity * pEnt,
    unsigned int geomIndex
);

Parameters                  Description
p                           Output point in WCS coordinates.
pEnt                        The entity containing the element.
geomIndex                   The element index.

Returns
Returns Acad::eOk on success.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapGeometryPE Class, AcMapGeometryPE Class
AcMapGeometryPE:: getVertex Method
AcMapGeometryPE Class | AcMapGeometryPE Class

Gets a vertex of a polyline or polygon.

```cpp
virtual Acad::ErrorStatus getVertex(
    AcGePoint3d & pt,
    double & bulge,
    AcGeVector3d & normal,
    AcDbEntity * pEnt,
    unsigned int geomIndex,
    unsigned int vertexIndex
);
```

Parameters Description
---
pt Output point in WCS coordinates.
bulge Output segment bulge.
normal Output segment normal.
pEnt Input the entity containing the element.
geomIndex Input the element index.
vertexIndex Input the vertex index.

Returns

Returns Acad::eOk on success.

Remarks

A segment bulge greater than zero indicates that the next segment is an arc. Where the next segment is an arc, segment normal identifies the plane in which the arc lies. Note that segment bulge at the last vertex is not defined, and neither is segment normal, because there is no next segment.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapQueryPE Class
AcMapQueryPE:: ELocationType Enumeration
AcMapQueryPE Class

Location-query process types.

```c
enum ELocationType {
    kBoundingBox = 1,
    kIntersectWith = 2,
    kAbstractGeometry = 3
};
```

File

AcMapQueryPE.h

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Property types for AutoCAD Map entities.

```cpp
enum EPropType {
    kElevation = 1,
    kThickness = 2,
    kBlockId = 3,
    kTextStyleId = 4,
    kTextStyle = 5,
    kArea = 6,
    kLength = 7,
    kRotation = 8,
    kHeight = 9,
    kScale = 10,
    kWidth = 11,
    kBulge = 12,
    kStartAngle = 13,
    kEndAngle = 14,
    kStartWidth = 15,
    kEndWidth = 16,
    kRadius = 17,
    kShapeName = 18,
    kShapeSize = 19,
    kImageName = 20,
    kAttributeTag = 21,
    kPoint1 = 22,
    kPoint2 = 23,
    kPoint3 = 24,
    kPoint4 = 25
};
```

File

AcMapQueryPE.h

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Can the entity be saved back?.

```cpp
virtual bool canSaveBack(
    AcDbEntity * pEnt = NULL,
    AcDbDatabase * toDb = NULL
);
```

**Parameters**
- `pEnt`: The entity.
- `toDb`: Database to save back to.

**Returns**

Returns true if the entity can be saved back.

**Remarks**

If the entity can be NULL, meaning you are being asked about all entities of this type, the database will be also NULL in this case.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Get any physical point on the entity to check whether it is inside location boundary or outside.

```c++
virtual Acad::ErrorStatus getPointsOnEntity(
    AcGePoint3dArray & pts,
    AcDbEntity * pEnt
);
```

**Parameters**
- `pts` Output a set of points.
- `pEnt` The entity.

**Returns**
Returns Acad::eOk on success.

**Remarks**
In case of complex geometry (like multi-point or multi-line) the function should return multiple points, one point per subgeometry. Default getStretchPoint() rarely gives the right result.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
virtual Acad::ErrorStatus get PropertyValue(
    AcMapValue & val,
    AcDbEntity * pEnt,
    AcMapQueryPE::EPropType type
);

Parameters
val
pEnt
type

Description
Output property value.
The entity.
Property type.

Returns

Returns Acad::eOk on success.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Does the entity have a given property?.

```
virtual bool hasProperty(
    AcDbEntity * pEnt,
    AcMapQueryPE::EPropType type
);
```

Parameters Description
pEnt The entity.
type Property type.

Returns

Returns true if it has the property.

Remarks

These are non-AcDbEntity properties used in property queries, property alteration, and AutoCAD Map expressions.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
One more property alteration functions telling whether we need to apply hatch alteration to the entity or not.

```cpp
virtual Acad::ErrorStatus isClosed(
    bool & bClosed,
    AcDbEntity * pEnt
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bClosed</td>
<td>Return flag.</td>
</tr>
<tr>
<td>pEnt</td>
<td>The entity.</td>
</tr>
</tbody>
</table>

Returns Acad::eOk on success.

Remarks

If not implemented, the default behaviour is to hatch MPolygons and closed AcDbCurves.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Can an entity be changed by property alteration?

```cpp
virtual bool isPropertyReadOnly(
    AcDbEntity * pEnt,
    AcMapQueryPE::EPropType type
);
```

**Parameters**

- **pEnt**
  - The entity.
- **type**
  - Property type to alter.

**Returns**

Returns true if the property can be altered.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Specifies how the entity will be processed in a location query.

```cpp
virtual AcMapQueryPE::ELocationType locationQueryType(AcDbEntity * pEnt);
```

Parameters

- `pEnt` - The entity.

Returns

Returns a location-query process type.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapQueryPE Class, AcMapQueryPE Class
AcMapQueryPE:: onClosedSet Method
AcMapQueryPE Class | AcMapQueryPE Class

Sends a notification to the entity at the time of closed set calculation (query and save back).

```cpp
virtual void onClosedSet(
    AcDbObjectIdArray & ids,
    AcDbEntity * pEnt
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ids</td>
<td>Default closed set, which can be changed inside this function.</td>
</tr>
<tr>
<td>pEnt</td>
<td>The entity.</td>
</tr>
</tbody>
</table>

Returns

Returns nothing.

Remarks

Default implementation is AcDbObject::dwgOut() with AcDb::kIdFiler.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapQueryPE Class, AcMapQueryPE Class
AcMapQueryPE:: setPropertyValue Method
AcMapQueryPE Class | AcMapQueryPE Class

Performs a property alteration on an entity.

virtual Acad::ErrorStatus setPropertyValue(
    AcDbEntity * pEnt,
    AcMapQueryPE::EPropType type,
    const AcMapValue & val
);

Parameters Description
pEnt The entity.
type Property type.
val Value to set.

Returns

Returns Acad::eOk on success.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Swap ID with the prototype entity during saving back.

```cpp
virtual Acad::ErrorStatus swapIdWith(
    AcDbEntity * pEnt,
    AcDbObjectId otherId,
    Adesk::Boolean swapXdata,
    Adesk::Boolean swapExtDict
);
```

---

**Parameters**

- **pEnt**: The entity.
- **otherId**: ID of the prototype entity.
- **swapXdata**: Input true to swap extended entity data.
- **swapExtDictionary**: Input true to swap extension dictionary.

**Returns**

Returns Acad::eOk on success.

**Remarks**

Saving back involves the following steps:

- Clone objects to the original database using `wblockCloneObjects()`.
- Swap IDs for all old and new copies using `AcDbObject::swapIdWith()`.
- Swap all references using `AcDbObject::swapReferences()`.
- Erase old copies.

There are two virtual `AcDbObject` methods which could be used by custom objects to participate in the process: `subSwapIdWith()` and `swapReferences()`. The `swapIdWith` method lets you control swapping Xdata and Extension dictionaries. By default, they are not swapped.

Created with a commercial version of [Doc-O-Matic](#). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Transforms the entity (projection transformation during query and save back, and rubber-sheeting).

```cpp
virtual Acad::ErrorStatus transformBy(
    AcDbEntity * pEnt,
    bool bMirror,
    bool bSimple,
    Acad::ErrorStatus (*getMatrixInPoint) (AcGeMatrix3d &mat, bool b);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pEnt</td>
<td>The entity.</td>
</tr>
<tr>
<td>bSimple</td>
<td>Output true if transformation matrix is constant, and you can call regular transformBy().</td>
</tr>
<tr>
<td>bMirror</td>
<td>Output true if you are rubber-sheeting and you need to swap all angles and bulges.</td>
</tr>
<tr>
<td>getMatrixInPoint</td>
<td>Callback function to calculate transformation matrix at the specified point.</td>
</tr>
<tr>
<td>mat</td>
<td>Output the result matrix.</td>
</tr>
<tr>
<td>bUniScaleOrtho</td>
<td>Output true if you need the matrix to be isUniScaledOrtho() because rubber-sheeting will create a non-orthogonal matrix.</td>
</tr>
<tr>
<td>p</td>
<td>The point to calculate matrix at.</td>
</tr>
</tbody>
</table>

Returns Acad::eOk on success.

Remarks

This is non-linear transformation, where matrix is different in the different points. Default implementation calculates a constant matrix at some middle point on the entity, and calls AcDbEntity::transformBy() with that constant matrix.
Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

```cpp
~AcMapDataSources(
    void
);
```

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Constructs an instance of this class.

```c
AcMapDataSources(
    void
);
Returns

Returns nothing.
```

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Detaches all data sources from the current AutoCAD Map work session.

```cpp
void DetachAllDataSources(
    void
);  
```

Returns

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Disconnects all data sources from the current AutoCAD Map work session.

```c++
void DisconnectAllDataSources(
    void
);
```

Returns

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Retrieves the names of the attached data sources in the current AutoCAD Map work session.

```c++
void GetAttachedDataSources(
    AcMapStringArray& paDataSourceNames
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>paDataSourceNames</td>
<td>Output array of the names of the attached data sources.</td>
</tr>
</tbody>
</table>

Returns

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Counts the number of attached data sources in the current AutoCAD Map work session.

```cpp
long GetAttachedDataSourcesCount(
    void
);

Returns

Returns the number of attached data sources.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the names of the connected data sources in the current AutoCAD Map work session.

```c
void GetConnectedDataSources(
    AcMapStringArray& paDataSourceNames
);
```

**Parameters**

| paDataSourceNames | Output array of the names of the connected data sources. |

**Returns**

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Counts the number of connected data sources in the current AutoCAD Map work session.

```c++
long GetConnectedDataSourcesCount(
    void
);
```

Returns the number of connected data sources.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the names of the disconnected data sources in the current AutoCAD Map work session.

```c
void GetDisconnectedDataSources(
    AcMapStringArray& paDataSourceNames
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>paDataSourceNames</td>
<td>Output array of the names of the disconnected data sources.</td>
</tr>
</tbody>
</table>

Returns

Returns nothing.

Created with a commercial version of [Doc-O-Matic](#). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Adds a scale-factor reactor to monitor scale-factor changes.

```c
void DMAddScaleFactorReactor(
    DMScaleFactorReactor * pReactor
);
```

Parameters

- **pReactor**: Input scale-factor reactor.

Returns

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Determine the display measurement units.

```cpp
bool DMDisplayIsMetric();
```

Returns

Returns true for centimeters, or false for inches.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determines the dwg measurement units.

```cpp
bool DMDwgUnitsAreMetric(
    double& dDwgConversionFactor,
    AcDbDatabase* pDb
);
```

File

DmScaleFactor.h

Parameters

- `dDwgConversionFactor`: Output double for converting from dwg units to inches or centimeters
- `pDb`: Input AcDbDatabase to test for metric dwg units.

Returns

Returns true for metric, or false for imperial.

Remarks

Intended for internal use only.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the cached current scale factor.

```
double DMGetCurrentScaleFactor();
```

File

DmScaleFactor.h

Returns

Returns the cached current scale factor.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Forces the scale factor to update.

```c
void DMRefreshScale();
```

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links

AcMapDMScaleFactor Namespace
AcMapDMScaleFactor:: DMRemoveScaleFactorReactor Function

AcMapDMScaleFactor Namespace

Removes a scale-factor reactor.

```c
void DMRemoveScaleFactorReactor(
    DMScaleFactorReactor * pReactor
);
```

File

DmScaleFactor.h

Parameters | Description
---|---
pReactor | Input scale-factor reactor.

Returns

Returns nothing.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMScaleFactor Namespace
DMScaleFactorReactor Class
AcMapDMScaleFactor Namespace

Reactor class for scale-factor changes.

class DMScaleFactorReactor;

File

DmScaleFactor.h

Methods

~DMScaleFactorReactor  Destroys an instance of this class.
DMScaleFactorReactor  Constructs an instance of this class.
setScaleFactorChanged  Invoked when the scale factor changes.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Enumerates the types of display-management active status.

```cpp
enum EDMStatus {
    kProcessing = 0x0001
};
```

File

DmDisplayManagement.h

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>kPassive</td>
<td>Passive.</td>
</tr>
<tr>
<td>kActive</td>
<td>Active.</td>
</tr>
</tbody>
</table>

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links

AcMapDMDisplayManagement Namespace
AcMapDMDisplayManagement:: AddProjectReactor Function

AcMapDMDisplayManagement Namespace

Adds a display-management reactor to monitor project-level activities such as creation and deletion.

```cpp
bool AddProjectReactor(
    AcMapProject * pProject,
    AcMapDMProjectReactor* pDMReactor
);
```

File

DmDisplayManagement.h

Parameters | Description
--- | ---
pProject | Input AutoCAD Map project.
pDMReactor | Input AcMapDMProjectReactor.

Returns

Returns true if successful.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Used to commit uncommitted edits to objects the Element queried in from external data sources.

```c++
Acad::ErrorStatus CommitEdits(
    AcMapDMElement* pElement
);
```

File

DmDisplayManagement.h

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pElement</td>
<td>Input Element for which this call applies.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful. Returns Acad::eNotApplicable if pElement->ClonesObjectsFromExternalSource() is false.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Used to commit uncommitted edits to objects the Elements within the specified Map queried in from external data sources.

```
Acad::ErrorStatus CommitEdits(
    AcMapDMMap* pMap
);
```

Parameters

- **pMap**: Input Map for which this call applies.

Returns

Returns Acad::eOk if successful. Returns Acad::eNotApplicable if all of the Map's Elements'

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMDisplayManagement Namespace
AcMapDMDisplayManagement:: DiscardEdits Function
AcMapDMDisplayManagement Namespace

Used to discard uncommitted edits to objects the Element queried in from external data sources.

```cpp
Acad::ErrorStatus DiscardEdits(
    AcMapDMElement* pElement
);
```

File

DmDisplayManagement.h

Parameters Description
pElement Input Element for which this call applies.

Returns

Returns Acad::eOk if successful. Returns Acad::eNotApplicable if pElement->ClonesObjectsFromExternalSource() is false.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Used to discard uncommitted edits to objects the Elements within the specified Map queried in from external data sources.

```cpp
Acad::ErrorStatus DiscardEdits(
    AcMapDMMMap* pMap
);
```

File

DmDisplayManagement.h

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pMap</td>
<td>Input Map for which this call applies.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful. Returns Acad::eNotApplicable if all of the Map's Elements'

Created with a commercial version of [Doc-O-Matic](https://toolsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapDMDisplayManagement Namespace
AcMapDMDisplayManagement:: DMMapManagerId Function
AcMapDMDisplayManagement Namespace

Retrieves the ID of the display-management map manager object for a database.

exist

AcDbObjectId DMMapManagerId(
    AcDbDatabase * pDatabase,
    bool bCreate = true
);
File

DmDisplayManagement.h

Parameters Description
pDatabase Input database.
bCreate Input true to create a map manager if one does not already exist. The default value is true.

Returns

Returns the ID of the AcMapDMMMapManager.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the ID of the display-management map manager object for a project.

```cpp
AcDbObjectId DMMapManagerId(
    AcMapProject * pProject,
    bool bCreate = true
);
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pProject</td>
<td>Input AutoCAD Map project.</td>
</tr>
<tr>
<td>bCreate</td>
<td>Input true to create a map manager if one does not already exist. The default value is true.</td>
</tr>
</tbody>
</table>

Returns

Returns the ID of the AcMapDMMapManager.

Created with a commercial version of [Doc-O-Matic](http://www.doc-o-matic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the ID of display-management style library object for a database.

```
AcDbObjectId DMStyleLibraryId(
    AcDbDatabase * pDb
);
```

Parameters | Description
--- | ---
pDb | Input database.

Returns

Returns the ID of the AcMapDMStyleLibrary.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
	AcMapDMDisplayManagement Namespace
	AcMapDMDisplayManagement:: DMStyleLibraryId Function
	AcMapDMDisplayManagement Namespace

Retrieves the ID of display-management style library object for a project.

AcDbObjectId DMStyleLibraryId(
    AcMapProject * pProject
);

File

DmDisplayManagement.h

Parameters Description
    pProject Input AutoCAD Map project.

Returns

Returns the ID of the AcMapDMStyleLibrary.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMDisplayManagement Namespace
AcMapDMDisplayManagement::GetCurrentMapBaseElementInvisibleEntities Function
AcMapDMDisplayManagement Namespace

Retrieves the visibility status of the map base element and the IDs of entities made invisible when it was unchecked.

```cpp
bool GetCurrentMapBaseElementInvisibleEntities(
    bool& bMapBaseElemInvisible,
    AcDbObjectIdArray& objIds
);
```

File
DmDisplayManagement.h

Parameters Description
bMapBaseElemInvisible Output flag for map base visibility (true indicates invisible).
objIds Output array of IDs of entities that were made invisible. If the map base element is visible, this array will have zero length.

Returns

Returns true if successful.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the display-management map manager object for the specified database.

```cpp
Acad::ErrorStatus GetDMMapManager(
    AcMapDMMapManager*& pMapMan,
    AcDbDatabase* pDatabase,
    AcDb::OpenMode mode
);
```

File

DmDisplayManagement.h

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pMapMan</td>
<td>Output AcMapDMMapManager.</td>
</tr>
<tr>
<td>pDatabase</td>
<td>Input database.</td>
</tr>
<tr>
<td>mode</td>
<td>Input mode to open the map manager in.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the display-management map manager object for a specified AutoCAD Map project.

```cpp
Acad::ErrorStatus GetDMMapManager(
    AcMapDMMMapManager*& pMapMan,
    AcMapProject* pProject,
    AcDb::OpenMode mode
);
```

**File**

DmDisplayManagement.h

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pMapMan</td>
<td>Output AcMapDMMMapManager.</td>
</tr>
<tr>
<td>pProject</td>
<td>Input AutoCAD Map project.</td>
</tr>
<tr>
<td>mode</td>
<td>Input mode to open the map manager in.</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

**Remarks**

If mode is AcDb::kForWrite and the objects do not exist, an empty map manager is created.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieve the display-management style library object for a database.

```cpp
Acad::ErrorStatus GetDMStyleLibrary(
    AcMapDMStyleLibrary*& pStyleLib,
    AcDbDatabase * pDb,
    AcDb::OpenMode mode
);
```

**File**

DmDisplayManagement.h

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pStyleLib</td>
<td>Output AcMapDMStyleLibrary.</td>
</tr>
<tr>
<td>pDb</td>
<td>Input database.</td>
</tr>
<tr>
<td>mode</td>
<td>Input mode to open the library in.</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Retrieves the display-management style library object for a project.

```cpp
Acad::ErrorStatus GetDMStyleLibrary(
    AcMapDMStyleLibrary*& pStyleLib,
    AcMapProject * pProject,
    AcDb::OpenMode mode
);
```

Parameters

- **pStyleLib**: Output AcMapDMStyleLibrary.
- **pProject**: Input AutoCAD Map project.
- **mode**: Input mode to open the library in.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determine whether a project has display-management data.

```cpp
bool HasDMData(
    AcMapProject * pProject
);
```

Parameters

- **pProject**: Input AutoCAD Map project.

Returns

Returns true if the project has display-management data.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMDisplayManagement Namespace
AcMapDMDisplayManagement:: HasUncommittedEdits Function
AcMapDMDisplayManagement Namespace

Used to determine if an Element has uncommitted edits to objects it queried in from external data sources.

```cpp
bool HasUncommittedEdits(
    AcMapDMElement* pElement
);
```

File
DmDisplayManagement.h

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pElement</td>
<td>Input Element for which this call applies.</td>
</tr>
</tbody>
</table>

Returns

Returns true if the Element has a dirty save set.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Used to determine if a Map has uncommitted edits to objects queried in from external data sources by any of the Map's Elements.

```cpp
bool HasUncommittedEdits(
    AcMapDMMap* pMap
);
```

File

DmDisplayManagement.h

Parameters

- **pMap**: Input Map for which this call applies.

Returns

Returns true if any Element within the Map has a dirty save set.

Remarks

All Elements within the Map are checked.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Notifies the display manager that an application is about to write a drawing database to an external data store.

```c
void OnBeginWritingToDataStore(
    AcDbDatabase * pDb
);
```

**File**

DmDisplayManagement.h

**Parameters**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pDb</td>
</tr>
<tr>
<td>Input drawing database to be written to an external data store.</td>
</tr>
</tbody>
</table>

**Returns**

Returns nothing.

**Remarks**

Display Manager removes all effects, if any, of stylization applied to entities.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Notifies the display manager that an application is finished writing a drawing database to an external data store.

```cpp
void OnEndWritingToDataStore(
    AcDbDatabase * pDb
);
```

**File**

DmDisplayManagement.h

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pDb</td>
<td>Input drawing database that was written to an external data store.</td>
</tr>
</tbody>
</table>

**Returns**

Returns nothing.

**Remarks**

Display Manager restores the effects, if any, of stylization.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Removes a display-management reactor from a project.

```cpp
bool RemoveProjectReactor(
    AcMapProject * pProject,
    AcMapDMProjectReactor* pDMReactor
);
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pProject</td>
<td>Input AutoCAD Map project.</td>
</tr>
<tr>
<td>pDMReactor</td>
<td>Input <code>AcMapDMProjectReactor</code>.</td>
</tr>
</tbody>
</table>

Returns

Returns true if successful.

Created with a commercial version of [Doc-O-Matic](https://www.doc-o-matic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the display-management current active status.

```c
unsigned int Status();
```

File

DmDisplayManagement.h

Returns

Returns the bitwise combination of status bits defined by EDMStatus.

Remarks

Display management is active when it perform stylization operations.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Builds range tables.

```cpp
virtual AcMapDMThematicBuildRangesErrorCode BuildRangeTables();
```

Returns `AcMapDMThematicBuildRangesErrorCode::keOk` if successful.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Adds a reactor to this map.

```cpp
bool AddReactor(
    AcMapDMMapReactor* pReactor
);
```

Parameters

<table>
<thead>
<tr>
<th>pReactor</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input AcMapDMMapReactor to add.</td>
<td></td>
</tr>
</tbody>
</table>

Returns

Returns true if successful.

Remarks

A reactor monitors map notifications.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Removes a reactor from this map.

```cpp
bool RemoveReactor(
    AcMapDMMapReactor* pReactor
);
```

**Parameters**
- **pReactor**: Input `AcMapDMMapReactor` to remove.

**Returns**
- Returns `true` if successful.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

```cpp
virtual ~AcMapDMAAllDrawOrderItemsIterator();
```

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determines whether the iterator has reached the end of the collection.

```cpp
virtual bool Done() const = 0;
```

Returns

Returns true if the iterator has reached the end of the collection; otherwise, returns false.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Advances to the next element in the iteration.

**virtual bool** Next() = 0;

Returns

Returns true if successful; otherwise, returns false.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Retrieves the ID of the current object in the iteration.

```cpp
virtual AcDbObjectId ObjectId() const = 0;
```

Returns a valid ID if successful.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMAAllItemsIterator Class, AcMapDMAAllItemsIterator Class
AcMapDMAAllItemsIterator:: ~AcMapDMAAllItemsIterator Destructor
AcMapDMAAllItemsIterator Class | AcMapDMAAllItemsIterator Class

Destroys an instance of this class.


virtual ~AcMapDMAAllItemsIterator();

Returns


Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determines whether the iterator has reached the end of the collection.

```cpp
virtual bool Done() const = 0;
```

Returns true if the iterator has reached the end of the collection; otherwise, returns false.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Advances to the next element in the iteration.

\texttt{virtual bool Next() = 0;}

Returns true if successful; otherwise, returns false.

Created with a commercial version of \texttt{Doc-O-Matic}. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at \texttt{support@toolsfactory.com}.
Retrieves the ID of the current object in the iteration.

```cpp
virtual AcDbObjectId ObjectId() const = 0;
```

Returns a valid ID if successful.

Created with a commercial version of [Doc-O-Matic](https://www.toolsfactory.com/docomatic). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Destroys an instance of this class.

```cpp
virtual ~AcMapDMAllStyleReferencesIterator();
```

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determines whether the iterator has reached the end of the collection.

```cpp
virtual bool Done() const = 0;
```

Returns

Returns true if the iterator has reached the end of the collection; otherwise, returns false.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Advances to the next element in the iteration.

```cpp
virtual bool Next() = 0;
```

Returns

Returns true if successful; otherwise, returns false.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMAI1StyleReferencesIterator Class,
AcMapDMAI1StyleReferencesIterator Class
AcMapDMAI1StyleReferencesIterator:: ObjectId Method
AcMapDMAI1StyleReferencesIterator Class |
AcMapDMAI1StyleReferencesIterator Class

Retrieves the ID of the current object in the iteration.

`virtual AcDbObjectId ObjectId() const = 0;`

Returns

Returns a valid ID if successful.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Moves to the first element in the iteration.

```cpp
virtual void Rewind() = 0;
```

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the ID of the style that the current object points to.

```cpp
virtual AcDbObjectId StyleId() const = 0;
```

Returns a valid ID if successful.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the scale threshold of the current object.

\texttt{virtual double ThresholdScale() const = 0;}

Returns the scale threshold.

Created with a commercial version of \texttt{Doc-O-Matic}. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at \texttt{support@toolsfactory.com}.
Destroys an instance of this class.

```cpp
virtual ~AcMapDMAAttachedDwgsQueryDataSourceDescriptor();
```

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Constructs an instance of this class.

AcMapDMAttachedDwgsQueryDataSourceDescriptor();

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Lists the drawings used as the scope of the query.

```cpp
virtual Acad::ErrorStatus GetDrawingList(
    AcMapStringArray* pDwgList
) const;
```

**Parameters**

- `pDwgList`  
  Output array of strings. Each array entry is a file path, given in the same sequence as in the project.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the query definition.

```cpp
virtual Acad::ErrorStatus GetQuery(
    struct resbuf*& pRb
) const;
```

Parameters

- **pRb**: Output resbuf object that contains the query.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the list of drawings to use as the scope of the query.

```cpp
virtual Acad::ErrorStatus SetDrawingList(
    const AcMapStringArray* pDwgList
);
```

**Parameters**

- **pDwgList**
  - Input array of strings. Each array entry is a file path, given in the same sequence as in the project.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the query definition.

```cpp
virtual Acad::ErrorStatus SetQuery(
    const struct resbuf* pRb
);```

**Parameters**

- **pRb**: Input resbuf object that contains the query.

**Returns**

- Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Destroys an instance of this class.

```cpp
virtual ~AcMapDMAttachedDwgsQueryElement();
```

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Constructs an instance of this class.

AcMapDMAttachedDwgsQueryElement();

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMAttachedDwgsQueryElement Class,
AcMapDMAttachedDwgsQueryElement Class
AcMapDMAttachedDwgsQueryElement:: AcquireEntities Method
AcMapDMAttachedDwgsQueryElement Class |
AcMapDMAttachedDwgsQueryElement Class

Runs an ADE query against the drawing scope defined in the data-source descriptor.

**virtual** Acad::ErrorStatus AcquireEntities();

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Remarks

See also
AcMapDMAttachedDwgsQueryDataSourceDescriptor::SetDrawingList().

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Clones objects from the source drawings.

```cpp
virtual bool ClonesObjectsFromExternalSource() const;
```

Returns true if successful; otherwise, returns false.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Erases entities that are part of this query element.

```cpp
virtual Acad::ErrorStatus DismissEntities();
```

Returns Acad::eOk if successful; otherwise, returns a different error code.

Remarks

See also AcquireEntities().

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
virtual Acad::ErrorStatus dwgInFields(
    AcDbDwgFiler* pFiler
);

Parameters
pFiler Input filer to use to read the object's data.

Returns
Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
AcMapDMAttachedDwgsQueryElement Class

AcMapDMAttachedDwgsQueryElement::dwgOutFields Method

Let this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dwgOutFields(
    AcDbDwgFiler* pFiler
) const;
```

### Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pFiler</td>
<td>Input filer to use to write the object's data.</td>
</tr>
</tbody>
</table>

### Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](http://www.dmtools.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Let this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dxfInFields(
   AcDbDxfFiler* pFiler
);
```

**Parameters**

- **pFiler**
  - Description: Input filer to use to read the object's data.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapDMAttachedDwgsQueryElement Class,
AcMapDMAttachedDwgsQueryElement Class
AcMapDMAttachedDwgsQueryElement:: dxfOutFields Method
AcMapDMAttachedDwgsQueryElement Class |
AcMapDMAttachedDwgsQueryElement Class

Lets this object write its data. See also dxfOutFields() in the AutoCAD
ObjectARX Developer's Guide. This overloaded function is called by the system
as needed; it is unlikely that you will need to call it directly.

virtual Acad::ErrorStatus dxfOutFields(
    AcDbDxfFiler* pFiler
) const;

Parameters Description
pFiler Input filer to use to write in the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this
message disappear you need to register this software. If you have problems
registering this software please contact us at support@toolsfactory.com.
Retrieves the query definition's data-source descriptor.

```cpp
virtual Acad::ErrorStatus GetAcquisitionCriteria(
    AcMapDMDataSourceDescriptor* pDataSourceDsc
) const;
```

Parameters

- `pDataSourceDsc`: Output pointer to the data-source descriptor.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMAttachedDwgsQueryElement Class,
AcMapDMAttachedDwgsQueryElement Class
AcMapDMAttachedDwgsQueryElement:: SetAcquisitionCriteria Method
AcMapDMAttachedDwgsQueryElement Class |
AcMapDMAttachedDwgsQueryElement Class

Sets the query definition's data-source descriptor.

```cpp
virtual Acad::ErrorStatus SetAcquisitionCriteria(
    const AcMapDMDataSourceDescriptor* pDataSourceDsc);
```

Parameters Description
pDataSourceDsc Input pointer to a data-source descriptor.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Remarks

This function determines which entities become part of this element.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMAAttchedDwgsQueryElement Class,
AcMapDMAAttchedDwgsQueryElement Class
AcMapDMAAttchedDwgsQueryElement:: SetVisible Method
AcMapDMAAttchedDwgsQueryElement Class |
AcMapDMAAttchedDwgsQueryElement Class

Makes the entities that are part of this element visible or invisible.

```cpp
virtual Acad::ErrorStatus SetVisible(
    bool bNewVal,
    double dScale
);
```

Parameters

- **bNewVal**: Input true to make the entities visible, or false to make them invisible.
- **dScale**: Input scale at which to make entities visible.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroyed an instance of this class.

\textbf{virtual} \texttt{\textasciitilde AcMapDMCurrentDwgQueryElement}();

Returns nothing.

Created with a commercial version of \texttt{Doc-O-Matic}. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at \texttt{support@toolsfactory.com}.
Constructs an instance of this class.

AcMapDMCurrentDwgQueryElement();
Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Runs the query against the current drawing.

```cpp
virtual Acad::ErrorStatus AcquireEntities();
```

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Clones objects from the source drawings.

```cpp
virtual bool ClonesObjectsFromExternalSource() const;
```

Returns false (because objects from only the current drawing are selected).

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapDMCurrentDwgQueryElement Class,
AcMapDMCurrentDwgQueryElement Class
AcMapDMCurrentDwgQueryElement:: dwgInFields Method
AcMapDMCurrentDwgQueryElement Class |
AcMapDMCurrentDwgQueryElement Class

Lets this object read its data. See also dwgInFields() in the AutoCAD
ObjectARX Developer's Guide. This overloaded function is called by the system
as needed; it is unlikely that you will need to call it directly.

virtual Acad::ErrorStatus dwgInFields(
    Acad::DwgFiler* pFiler
);

Parameters Description
pFiler Input filer to use to read the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this
message disappear you need to register this software. If you have problems
registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMCurrentDwgQueryElement Class,
AcMapDMCurrentDwgQueryElement Class
AcMapDMCurrentDwgQueryElement:: dwgOutFields Method
AcMapDMCurrentDwgQueryElement Class |
AcMapDMCurrentDwgQueryElement Class

Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dwgOutFields(
    AcDbDwgFiler* pFiler
) const;
```

**Parameters**

* pFiler

**Description**

Input filer to use to write the object's data.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](#). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapDMCurrentDwgQueryElement Class,
AcMapDMCurrentDwgQueryElement Class
AcMapDMCurrentDwgQueryElement:: dxfInFields Method
AcMapDMCurrentDwgQueryElement Class |
AcMapDMCurrentDwgQueryElement Class

Lets this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

virtual Acad::ErrorStatus dxfInFields(
    AcDbDxfFiler* pFiler
);
Parameters  Description
pFiler  Input filer to use to read the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
virtual Acad::ErrorStatus dxfOutFields(
   AcDbDxfFiler* pFiler
) const;

Parameters Description
pFiler Input filer to use to write in the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the query definition's data-source descriptor.

```cpp
virtual Acad::ErrorStatus GetAcquisitionCriteria(
    AcMapDMDataSourceDescriptor* pDataSourceDsc
) const;
```

Parameters

- **pDataSourceDsc**: Output pointer to the data-source descriptor.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Invoked when an AutoCAD Map project is initialized.

```cpp
virtual bool OnMapProjectInitialized(
    AcMapProject* pMapProject
);
```

**Parameters**
- `pMapProject` Input AutoCAD Map project.

**Returns**
Returns true if successful.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the query definition's data-source descriptor.

```cpp
virtual Acad::ErrorStatus SetAcquisitionCriteria(
    const AcMapDMDDataSourceDescriptor* pDataSourceDsc);
```

**Parameters**

- **Description**
  - `pDataSourceDsc`: Input pointer to a data-source descriptor.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

**Remarks**

This function determines which entities become part of this element.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Enumerates the types of default styles.

```cpp
enum DefaultStyleType {
    kEntityFade = 0
};
```

**Parameters**
- **kEntityFade**: Fade style.

Created with a commercial version of [Doc-O-Matic](https://toolsfactory.com/docomatic). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapDMDefaultStyle Class, AcMapDMDefaultStyle Class
AcMapDMDefaultStyle:: ~AcMapDMDefaultStyle Destructor
AcMapDMDefaultStyle Class | AcMapDMDefaultStyle Class

Destroys an instance of this class.

virtual ~AcMapDMDefaultStyle();
Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Constructs an instance of this class.

AcMapDMDefaultStyle();

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Applies the style to an entity.

```cpp
virtual Acad::ErrorStatus Apply(
    void*& pCookie,
    AcDbEntity* pEnt,
    Adesk::UInt32 flag = 0
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pCookie</td>
<td>Input memory value used by styles to transfer cached data across time from one stylization method to another. Cookie memory is considered to be opaque for existing styles, and should not be modified by the caller.</td>
</tr>
<tr>
<td>pEnt</td>
<td>Input entity to which the style is applied.</td>
</tr>
<tr>
<td>flag</td>
<td>Reserved. For internal use only.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Clears the fade. See also IsFadeCleared().

`Acad::ErrorStatus ClearFade();`

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Clones the object. See also clone() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual AcRxObject* clone() const;
```

Returns a pointer to the clone if successful, or NULL otherwise.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Copies the contents of an object into the messaged object, if feasible. See also copyFrom() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus copyFrom(
    const AcRxObject* other
);
```

**Parameters**

- `other` Input pointer to the object to copy from.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMDefaultStyle Class, AcMapDMDefaultStyle Class
AcMapDMDefaultStyle::dwgInFields Method
AcMapDMDefaultStyle Class | AcMapDMDefaultStyle Class

Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dwgInFields(
    AcDbDwgFiler* pFiler
);
```

**Parameters**
- **Description**
  - `pFiler`: Input filer to use to read the object's data.

**Returns**
Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMDefaultStyle Class, AcMapDMDefaultStyle Class
AcMapDMDefaultStyle:: dwgOutFields Method
AcMapDMDefaultStyle Class | AcMapDMDefaultStyle Class

Let this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dwgOutFields(
    AcDbDwgFiler* pFiler
) const;
```

**Parameters**
- **pFiler**
  - Input filer to use to write the object's data.

**Returns**
- Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](mailto:info@toolsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapDMDefaultStyle Class, AcMapDMDefaultStyle Class
AcMapDMDefaultStyle:: dxfInFields Method
AcMapDMDefaultStyle Class | AcMapDMDefaultStyle Class

Lets this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dxfInFields(
    AcDbDxfFiler* pFiler
);
```

Parameters                  Description
pFiler                     Input filer to use to read the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMDefaultStyle Class, AcMapDMDefaultStyle Class
AcMapDMDefaultStyle:: dxfOutFields Method
AcMapDMDefaultStyle Class | AcMapDMDefaultStyle Class

Lets this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dxfOutFields(
    AcDbDxfFiler* pFiler
) const;
```

Parameters Description
pFiler Input filer to use to write the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the fade property. See also SetFade().

```cpp
Acad::ErrorStatus GetFade(
    Adesk::UInt8& fade
) const;
```

**Parameters**

- `fade` : Output fade value.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the style type. This function is intended for use in future versions of AutoCAD Map, when styles other than DefaultStyleType::kEntityFade may be available. See also SetStyleType().

```
AcMapDMDefaultStyle::DefaultStyleType GetStyleType() const;
```

Returns

Returns the DefaultStyleType style type.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determines whether the fade is cleared. See also ClearFade().

```cpp
bool IsFadeCleared() const;
```

Returns true if the fade is cleared, or false if it is not.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the fade property. In this release of AutoCAD Map, this function simply turns the target entity to ACI color 9 (light gray). Future releases may permit other fade colors. See also GetFade().

```
Acad::ErrorStatus SetFade(
    Adesk::UInt8 fade
);
```

**Parameters**

| fade | Input fade value. |

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](mailto:https://toolsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the style type. This function is intended for use in future versions of AutoCAD Map, when styles other than DefaultStyleType::kEntityFade may be available. See also GetStyleType().

```
Acad::ErrorStatus SetStyleType(
    AcMapDMDefaultStyle::DefaultStyleType type
);
```

Parameters
- **type**: Input DefaultStyleType style type.

Returns
- Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Builds stylization information for later regeneration.

```cpp
virtual Acad::ErrorStatus UnApply(
    void*& pCookie,
    AcDbEntity* pEnt,
    Adesk::UInt32 flag = 0
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pCookie</td>
<td>Input memory value used by styles to transfer cached data across time from one stylization method to another. Cookie memory is considered to be opaque for existing styles, and should not be modified by the caller.</td>
</tr>
<tr>
<td>pEnt</td>
<td>Input entity.</td>
</tr>
<tr>
<td>flag</td>
<td>Reserved. For internal use only.</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

**Remarks**

The caller can retrieve this style's cookie, stored previously by another stylization function.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Enumerates the preview entity types.

```cpp
enum EPreviewEntityType {
    kPreviewArc = 1,
    kPreviewLine,
    kPreviewDefault = kPreviewLine,
    kPreviewPoint,
    kPreviewPolyline,
    kPreviewPolygon,
    kPreviewText,
    kPreviewBlock,
    kPreviewCustom
};
```

File

DmDisplayElement.h

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>kPreviewArc</td>
<td>Arc.</td>
</tr>
<tr>
<td>kPreviewLine</td>
<td>Line.</td>
</tr>
<tr>
<td>kPreviewDefault</td>
<td>Default preview entity type (a kPreviewLine).</td>
</tr>
<tr>
<td>kPreviewPoint</td>
<td>Point.</td>
</tr>
<tr>
<td>kPreviewPolyline</td>
<td>Polyline.</td>
</tr>
<tr>
<td>kPreviewPolygon</td>
<td>Polygon.</td>
</tr>
<tr>
<td>kPreviewText</td>
<td>Text.</td>
</tr>
<tr>
<td>kPreviewBlock</td>
<td>Block.</td>
</tr>
<tr>
<td>kPreviewCustom</td>
<td>Not supported in this release of AutoCAD Map.</td>
</tr>
</tbody>
</table>

Remarks

These preview types determine the type that will be shown for the thumbnail preview of styles that are applied to the element.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems
registering this software please contact us at support@toolsfactory.com.
Destroy an instance of this class.

\textbf{virtual} \texttt{~AcMapDMElement()};


\textbf{Returns}

Returns nothing.

Created with a commercial version of \texttt{Doc-O-Matic}. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at \texttt{support@toolsfactory.com}.  

\textbf{Links}

\texttt{AcMapDMElement Class}, \texttt{AcMapDMElement Class}

\texttt{AcMapDMElement:: \texttt{~AcMapDMElement Destructor}}

\texttt{AcMapDMElement Class | AcMapDMElement Class}
Constructs an instance of this class.

AcMapDMElement();
Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMElement Class, AcMapDMElement Class
AcMapDMElement:: AcquireEntities Method
AcMapDMElement Class | AcMapDMElement Class

Runs the query and retrieves and stores the IDs of objects that meet the selection criteria.

`virtual Acad::ErrorStatus AcquireEntities();`

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
This function is called by AutoCAD when the AUDIT command is executed.

```cpp
virtual Acad::ErrorStatus audit(
    AcDbAuditInfo* pAuditInfo
);
```

### Parameters

**pAuditInfo**

The AcDbAuditInfo object pointed to by pAuditInfo contains member functions that are used to determine what to do and also to report the results of the audit operation on the object.

### Returns

Returns Acad::eOk if successful, Acad::eFixedAllErrors if all errors were corrected, Acad::eLeftErrorsUnfixed if there were errors that were left unfixed; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Clones objects that are from the external source at AcquireEntities() time.

```cpp
virtual bool ClonesObjectsFromExternalSource() const;
```

Returns or true if objects from an external source were cloned successfully; returns false if entities were acquired from the current drawing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Clears the acquired entities that are stored in this element.

```cpp
virtual Acad::ErrorStatus DismissEntities();
```

Returns Acad::eOk if successful; otherwise, returns a different error code.

Remarks

A derived function can perform extra cleanup.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMEElement Class, AcMapDMEElement Class
AcMapDMEElement::: DismissStylization Method
AcMapDMEElement Class | AcMapDMEElement Class

Clears the current stylization.

```c
virtual Acad::ErrorStatus DismissStylization(
    bool bDismissEntities
);
```

Parameters | Description
------------|-------------
bDismissEntities | Input true to clear the acquired entities.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMElement Class, AcMapDMElement Class
AcMapDMElement:: dwgInFields Method
AcMapDMElement Class | AcMapDMElement Class

Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dwgInFields(
    AcDbDwgFiler* pFiler
);
```

Parameters
- **pFiler**
  - Input filer to use to read the object's data.

Returns
- Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMEElement Class, AcMapDMEElement Class
AcMapDMElement:: dwgOutFields Method
AcMapDMElement Class | AcMapDMEElement Class

Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dwgOutFields(
    AcDbDwgFiler* pFiler
) const;
```

Parameters  Description
pFiler       Input filer to use to write the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMElement Class, AcMapDMElement Class
AcMapDMElement:: dxfInFields Method
AcMapDMElement Class | AcMapDMElement Class

Lets this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dxfInFields(
    AcDbDxfFiler* pFiler
);
```

Parameters

Description

pFiler Input filer to use to read the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMElement Class, AcMapDMElement Class
AcMapDMElement:: dxfOutFields Method
AcMapDMElement Class | AcMapDMElement Class

Lets this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dxfOutFields(
    AcDbDxfFiler* pFiler
) const;
```

Parameters

- **pFiler**: Input filer to use to write the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links

AcMapDMElement Class, AcMapDMElement Class
AcMapDMElement:: EnableStyle Method
AcMapDMElement Class | AcMapDMElement Class

Enables the style reference's style.

```cpp
virtual Acad::ErrorStatus EnableStyle(
    AcDbObjectId styleRefId,
    bool bNewEnableStatus
);
```

Parameters

- **styleRefId**: Input ID of the style reference.
- **bNewEnableStatus**: Input true to enable the style, or false to disable it.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Filters out those entities from the selection that, during stylization, acquired stylization entities of another element.

```cpp
void FilterOutStylizationEntities(
    AcDbObjectIdArray& prunedSelection,
    const AcDbObjectIdArray& selectedIds
) const;
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>prunedSelection</td>
<td>Output array of IDs of the filtered objects.</td>
</tr>
<tr>
<td>selectedIds</td>
<td>Input array containing all IDs of acquired objects.</td>
</tr>
</tbody>
</table>

Returns nothing.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Retrieves the current selection (defined by calling AcquireEntities()).

```cpp
virtual Acad::ErrorStatus GetAcquiredEntities(
    AcDbObjectIdArray& Ids
) const;
```

**Parameters**

<table>
<thead>
<tr>
<th>Ids</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ids</td>
<td>Output array of object IDs of the acquired objects.</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://toolsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapDMElement Class, AcMapDMElement Class
AcMapDMElement:: GetAcquisitionCriteria Method
AcMapDMElement Class | AcMapDMElement Class

Retrieves the acquisition criteria for this element.

```
virtual Acad::ErrorStatus GetAcquisitionCriteria(  
    AcMapDMDataSourceDescriptor* pDataSourceDsc  
) const;
```

**Parameters**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Output AcMapDMDataSourceDescriptor data-source descriptor.</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves an iterator over a collection of style references (for the corresponding style set) applied to this element.

```
AcMapDMAllStyleReferencesIterator* GetAllStyleReferencesIterator(
    double dScale = 0.,
    bool bForward = true,
    bool bSkipProxies = false
) const;
```

**Parameters**

- **dScale**
  - Input scale. The default value is zero, the current scale.

- **bForward**
  - Input true for forward traversal.

- **bSkipProxies**
  - Input true to skip over StyleReferences that refer to proxy Styles

**Returns**

Returns the `AcMapDMAllStyleReferencesIterator`.

**Remarks**

The iterator will iterate all StyleReferences, and StyleId() may return an Id to an AcDbProxyObject, unless the bSkipProxies argument is passed as true.

Created with a commercial version of [Doc-O-Matic](#). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Retrieves the custom preview entity name. See also \texttt{SetCustomPreviewEntityName()}. 

\begin{verbatim}
virtual const ACHAR* GetCustomPreviewEntityName() const;
\end{verbatim}

Returns NULL for all preview entity types except \texttt{kPreviewBlock} and \texttt{kPreviewCustom}. For \texttt{kPreviewBlock}, a block name is returned. For \texttt{kPreviewCustom}, a service name is returned - service names are not yet supported in AutoCAD Map.

Created with a commercial version of \texttt{Doc-O-Matic}. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at \texttt{support@toolsfactory.com}. 

Links
\texttt{AcMapDMElement Class}, \texttt{AcMapDMElement Class}
\texttt{AcMapDMElement::GetCustomPreviewEntityName Method}
\texttt{AcMapDMElement Class | AcMapDMElement Class}
Links
AcMapDMElement Class, AcMapDMElement Class
AcMapDMElement:: GetPreviewBlockId Method
AcMapDMElement Class | AcMapDMElement Class

Retrieves the AcDbObjectId of the block used for preview, if preview is of type kPreviewBlock. See also SetPreviewBlockId().

AcDbObjectId GetPreviewBlockId() const;
Returns

Returns AcDbObjectId with kNull value for all preview entity types except kPreviewBlock

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the preview entity type of this element. See also SetPreviewEntityType().

```cpp
virtual EPreviewEntityTypetype GetPreviewEntityType() const;
```

Returns

Returns the EPreviewEntityTypetype thumbnail preview type.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the preview entity type. See also `GetPreviewEntityType()`.

```cpp
virtual Acad::ErrorStatus SetPreviewEntityType(
    EPreviewEntityType ePreviewType
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ePreviewType</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com/). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Determines whether this element has a specified style reference.

```cpp
bool HasStyleReference(
    const AcDbObjectId& styleRefId,
    double dScale
);
```

**Parameters**

- `styleRefId` Input ID of the style reference.
- `dScale` Input scale.

**Returns**

Returns true if this element has the specified style reference; otherwise, returns false.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determined whether this element has a style associated with it at a specified scale.

```cpp
bool IsStyleApplied(
    const AcDbObjectId& styleId,
    double dScale,
    bool bIgnoreDisabled = true
) const;
```

Parameters

- **styleId**: Input ID of the style.
- **dScale**: Input scale.
- **bIgnoreDisabled**: Input true to ignore styles that are disabled at the specified scale. The default value is true.

Returns

Returns true if the style is applied.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determine whether this element is stylized.

```cpp
bool IsStylized() const;
```

Returns true if this element is stylized.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determines whether the style set on this element is unique. See also MakeStyleSetUnique().

```cpp
bool IsUniqueStyleSet(
    double dScale = 0.
) const;
```

Parameters

- **dScale**
  - Input scale. The default value is zero, the current scale.

Returns

Returns true if the style set is unique, or false if the style set is shared among multiple scales.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Determine whether this element is visible at the specified scale. See also SetVisible().

```cpp
virtual bool isVisible(
    double dScale = 0.
) const;
```

**Parameters**

- `dScale`: Input scale at which visibility is evaluated. The default value is zero, the current scale.

**Returns**

Returns true if the element is visible at the specified scale; otherwise, returns false.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMElemet Class, AcMapDMElemet Class
AcMapDMElemet:: MakeStyleSetUnique Method
AcMapDMElemet Class | AcMapDMElemet Class

Makes unique the style set associated with this element. See also IsUniqueStyleSet().

```cpp
Acad::ErrorStatus MakeStyleSetUnique(
    bool bIsUnique,
    double dScale = 0.
);
```

### Parameters

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input true to copy the style set associated with the specified map scale and use this copy for the specified and lower scales. Input false to delete the style set associated with the specified map scale, making the specified scale refer to the style set associated with higher scales.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input scale at which to make the style set unique. The default value is zero, the current scale.</td>
</tr>
</tbody>
</table>

### Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Invoked when an AutoCAD Map project is initialized.

```
virtual bool OnMapProjectInitialized(
    AcMapProject* pMapProject
);
```

**Parameters**

- `pMapProject` Input AutoCAD Map project.

**Returns**

Returns true if successful.

Created with a commercial version of [Doc-O-Matic](https://www.toolsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Invoked when an object is appended to the database.

**virtual** Acad::ErrorStatus OnObjectAppended();

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Order an array of objects based on their current draw order.

```cpp
Acad::ErrorStatus OrderEntitiesByDrawOrder(
    AcDbObjectIdArray & ids
);
```

Parameters

- **ids**
  - Description: Input/Output object id array.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Place this array of objects at the top of draw order.

```
Acad::ErrorStatus PlaceEntitiesInFrontOfDrawOrder(
    AcDbObjectIdArray & ids
);
```

**Parameters**

- **ids**
  - Input/Output object id array.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.toolsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapDMElement Class, AcMapDMElement Class
AcMapDMElement:: QueueObjectsForRegenerate Method
AcMapDMElement Class | AcMapDMElement Class

Puts all queried objects that are part of this element on the regeneration queue.

```cpp
int QueueObjectsForRegenerate() const;
```

Returns the number of objects queued for regeneration.

Remarks
Your application must trigger regeneration explicitly.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Removes a style from this element.

```cpp
virtual Acad::ErrorStatus RemoveStyle(
    AcDbObjectId styleRefId
);
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>styleRefId</td>
<td>Input ID of the style reference.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Removes a style from this element.

```cpp
Acad::ErrorStatus RemoveStyle(
    const AcDbObjectId& styleRefId,
    double dScale
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>styleRefId</td>
<td>Input ID of the style reference.</td>
</tr>
<tr>
<td>dScale</td>
<td>Input scale.</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the current selection (defined by calling AcquireEntities()).

```c++
Acad::ErrorStatus SetAcquiredEntities(
    AcDbObjectIdArray& Ids
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ids</td>
<td>Input array of object IDs of the acquired objects.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Remarks

Called by derived Elements during AcquireEntities() to store their acquisition set.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the acquisition criteria for this element.

```cpp
virtual Acad::ErrorStatus SetAcquisitionCriteria(
    const AcMapDMDataSourceDescriptor* pDataSourceDsc);
```

**Parameters**

- `pDataSourceDsc`: Input `AcMapDMDataSourceDescriptor` data-source descriptor.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

**Remarks**

This function determines which objects become part of this element.

Created with a commercial version of [Doc-O-Matic](https://toolsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the AcDbObjectId of the block for kPreviewBlock preview entity type. See also GetPreviewBlockId().

```
Acad::ErrorStatus SetPreviewBlockId(
    AcDbObjectId blockId
);
```

**Parameters**

<table>
<thead>
<tr>
<th>blockId</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input AcDbObjectId of the block.</td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the element visibility at the specified scale. See also IsVisible().

```cpp
virtual Acad::ErrorStatus SetVisible(
    bool bNewVal,
    double dScale = 0
);
```

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bNewVal</td>
<td>Input true to make the element visible at the specified scale, or false to make it invisible.</td>
</tr>
<tr>
<td>dScale</td>
<td>Input scale at which visibility is set. The default value is zero, the current scale.</td>
</tr>
</tbody>
</table>

### Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Invoked from within erase() before the erase actually occurs. See also subErase() in the AutoCAD ObjectARX Developer's Guide.

```cpp
virtual Acad::ErrorStatus subErase(
    Adesk::Boolean erasing
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Description</th>
<th>Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Passed-in copy of the erasing argument that was passed to the erase() function call that triggered this subErase() call.</td>
<td>erasing</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Stylizes the current or updated selection at the current scale.

```cpp
virtual Acad::ErrorStatus UpdateStylization(
    bool bAcquireEntities
);```

**Parameters**
- `bAcquireEntities`: Input true to acquire entities.

**Returns**
- Returns Acad::eOk if successful. Returns Acad::eAmbiguousOutput if `bAcquireEntities` is true but the previous selection has not been dismissed.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Destroys an instance of this class.

```
virtual ~AcMapDMEntityStyle();
```

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Constructs an instance of this class.

AcMapDMEntityStyle();

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Applies the style to an entity.

```cpp
virtual Acad::ErrorStatus Apply(
    void*& pCookie,
    AcDbEntity* pEnt,
    Adesk::UInt32 flag = 0
);
```

**Parameters**

- **pCookie**: Input memory value used by styles to transfer cached data across time from one stylization method to another. Cookie memory is considered to be opaque for existing styles, and should not be modified by the caller.
- **pEnt**: Input entity to apply the style to. This entity must have been opened for write.
- **flag**: Reserved. For internal use only.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Clears the color. See also IsColorCleared().

```cpp
Acad::ErrorStatus clearColor();
```

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Clears the linetype. See also IsLinetypeCleared().

```
Acad::ErrorStatus ClearLinetype();
```

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Clears the linetype scale. See also IsLinetypeScaleCleared().

```
Acad::ErrorStatus ClearLinetypeScale();
Returns Acad::eOk if successful; otherwise, returns a different error code.
```

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determines whether the linetype scale is cleared. See also ClearLinetypeScale().

```cpp
bool IsLinetypeScaleCleared() const;
```

Returns true if the linetype scale is cleared.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Clears the lineweight. See also IsLineWeightCleared().

```cpp
Acad::ErrorStatus ClearLineWeight();
```

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.doc-o-matic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Clears the plotstyle. See also IsPlotstyleNameCleared().

Acad::ErrorStatus ClearPlotstyleName();

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMEntityStyle Class, AcMapDMEntityStyle Class
AcMapDMEntityStyle:: IsPlotstyleNameCleared Method
AcMapDMEntityStyle Class | AcMapDMEntityStyle Class

Determines whether the plotstyle is cleared. See also ClearPlotstyleName().

```cpp
bool IsPlotstyleNameCleared() const;
```

Returns true if the plotstyle is cleared.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Clones the object. See also clone() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual AcRxObject* clone() const;
```

Returns a pointer to the clone, or NULL if cloning was unsuccessful.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMEntityStyle Class, AcMapDMEntityStyle Class
AcMapDMEntityStyle::: copyFrom Method
AcMapDMEntityStyle Class | AcMapDMEntityStyle Class

Copies the contents of an object into the messaged object, if feasible. See also copyFrom() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus copyFrom(
    const AcRxObject* other
);
```

Parameters

- **other**: Input pointer to the object to copy from.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Clears the stylization on an entity.

```cpp
virtual Acad::ErrorStatus Dismiss(
    void*& pCookie,
    const AcDbObjectId id,
    Adesk::UInt32 flag = 0
);
```

**Parameters**

- **pCookie**: Input memory value used by styles to transfer cached data across time from one stylization method to another. Cookie memory is considered to be opaque for existing styles, and should not be modified by the caller.
- **id**: Input entity ID.
- **flag**: Reserved. For internal use only.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapDMEntityStyle Class, AcMapDMEntityStyle Class
AcMapDMEntityStyle:: dwgInFields Method
AcMapDMEntityStyle Class | AcMapDMEntityStyle Class

Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dwgInFields(
    AcDbDwgFiler* pFiler
);
```

Parameters

- **pFiler**: Input filer to use to read the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.doctools.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dwgOutFields(
    AcDbDwgFiler* pFiler
) const;
```

**Parameters**
- `pFiler`: Input filer to use to write the object's data.

**Returns**
- Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
lets this object read its data. see also dxfinfields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dxfInFields(
   AcDbDxfFiler* pFiler
);    
```

**Parameters**

- **pFiler**
  - Input filer to use to read the object's data.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](#). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapDMEntityStyle Class, AcMapDMEntityStyle Class
AcMapDMEntityStyle:: dxfOutFields Method
AcMapDMEntityStyle Class | AcMapDMEntityStyle Class

Lets this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dxfOutFields(
    AcDbDxfFiler* pFiler
) const;
```

Parameters
- **pFiler**: Input filer to use to write the object's data.

Returns
- Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the color. See also SetColor().

```cpp
Acad::ErrorStatus GetColor(
    const AcCmColor*& pColor
) const;
```

**Parameters**

- `pColor`: Output color value.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the linetype id. See also SetLinetype().

Acad::ErrorStatus GetLinetype(
    AcDbObjectId& linetypeId
) const;

Parameters
linetypeId    Output linetype id.

Returns
Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the linetype scale. See also SetLinetypeScale().

```c++
Acad::ErrorStatus GetLinetypeScale(
    double& dLinetypeScale
) const;
```

**Parameters**
- `dLinetypeScale` Output linetype scale.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Retrieves the lineweight. See also SetLineWeight().

```cpp
Acad::ErrorStatus GetLineWeight(
   AcDb::LineWeight& lw
) const;
```

**Parameters**
- `lw` Output lineweight.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Determine whether the color is cleared. See also ClearColor().

```cpp
bool IsColorCleared() const;
```

Returns true if the color is cleared.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determines whether the linetype is cleared. See also ClearLinetype().

```cpp
bool IsLinetypeCleared() const;
```

Returns true if the linetype is cleared.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determines whether the lineweight is cleared. See also ClearLineWeight().

```cpp
bool IsLineWeightCleared() const;
```

Returns true if the lineweight is cleared.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the color. See also GetColor().

\[
\text{Acad::ErrorStatus SetColor(} \\
\quad \text{AcCmColor color)} \\
\]

Parameters | Description
---|---
color | Input color.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.
Sets the linetype. See also GetLinetype().

```
Acad::ErrorStatus SetLinetype(
    const AcDbObjectId linetypeId
);
```

Parameters

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>linetypeId</td>
</tr>
<tr>
<td>Input linetype id.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the linetype scale. See also GetLinetypeScale().

```cpp
Acad::ErrorStatus SetLinetypeScale(  
    double dLinetypeScale  
);
```

**Parameters**

- **dLinetypeScale**
  Input linetype scale.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](#). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the lineweight. See also GetLineWeight().

Acad::ErrorStatus SetLineWeight(
    AcDb::LineWeight lw
);

Parameters Description
lw Input lineweight.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMEntityStyle Class, AcMapDMEntityStyle Class
AcMapDMEntityStyle:: UnApply Method
AcMapDMEntityStyle Class | AcMapDMEntityStyle Class

Removes the style from a specified entity.

```cpp
virtual Acad::ErrorStatus UnApply(
    void*& pCookie,
    AcDbEntity* pEnt,
    Adesk::UInt32 flag = 0
);
```

Parameters
- **pCookie**: Input memory value used by styles to transfer cached data across time from one stylization method to another. Cookie memory is considered to be opaque for existing styles, and should not be modified by the caller.
- **pEnt**: Input entity.
- **flag**: Reserved. For internal use only.

Returns
Returns Acad::eOk if successful; otherwise, returns a different error code.

Remarks
The caller can retrieve this style's cookie, stored previously by another stylization function.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves and stylizes an entity.

```cpp
virtual Acad::ErrorStatus Update(
    void*& pCookie,
    const AcDbObjectId id,
    Adesk::UInt32 flag = 0
);
```

**Parameters**

- **pCookie**
  - Output memory value used by styles to transfer cached data across time from one stylization method to another. Cookie memory is considered to be opaque for existing styles, and should not be modified by the caller.

- **id**
  - Input entity ID.

- **flag**
  - Reserved. For internal use only.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Destroys an instance of this class.

virtual ~AcMapDMFeatureDataSourceDescriptor();

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Constructs an instance of this class.

AcMapDMFeatureDataSourceDescriptor();

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

```cpp
virtual ~AcMapDMFeatureElement();
```

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMFeatureElement Class, AcMapDMFeatureElement Class
AcMapDMFeatureElement:: AcMapDMFeatureElement Constructor
AcMapDMFeatureElement Class | AcMapDMFeatureElement Class

Constructs an instance of this class.

AcMapDMFeatureElement();
Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Runs an ADE query against the drawing scope defined in the data-source descriptor.

```c++
virtual Acad::ErrorStatus AcquireEntities();
```

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Remarks

See also AcMapDMDataSourceDescriptor::SetDrawingList().

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Clones objects from the source drawings.

```cpp
virtual bool ClonesObjectsFromExternalSource() const;
```

Returns false.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Let this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dwgInFields(
    AcDbDwgFiler* pFiler
);
```

**Parameters**

- **pFiler**
  - Input filer to use to read the object's data.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapDMFeatureElement Class, AcMapDMFeatureElement Class
AcMapDMFeatureElement:: dwgOutFields Method
AcMapDMFeatureElement Class | AcMapDMFeatureElement Class

Let's this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dwgOutFields(
    AcDbDwgFiler* pFiler
) const;
```

Parameters

- `pFiler`: Input filer to use to write the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMFeatureElement Class, AcMapDMFeatureElement Class
AcMapDMFeatureElement:: dxfInFields Method
AcMapDMFeatureElement Class | AcMapDMFeatureElement Class

Lets this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```c++
virtual Acad::ErrorStatus dxfInFields(
    AcDbDxfFiler* pFiler
);
```

Parameters
description

- **pFiler**: Input filer to use to read the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Let this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dxfOutFields(
    AcDbDxfFiler* pFiler
) const;
```

- **Parameters**
  - `pFiler`: Input filer to use to write the object's data.

- **Returns**
  - Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the query definition's data-source descriptor.

```cpp
virtual Acad::ErrorStatus GetAcquisitionCriteria(
    AcMapDMDataSourceDescriptor* pDataSourceDsc
) const;
```

**Parameters**

- `pDataSourceDsc` Output pointer to a data-source descriptor.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Invoked when an AutoCAD Map project is initialized.

```cpp
virtual bool OnMapProjectInitialized(
    AcMapProject* pMapProject
);```

**Parameters**

- `pMapProject` Input AutoCAD Map project.

**Returns**

Returns true if successful.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the query definition's data-source descriptor.

```cpp
virtual Acad::ErrorStatus SetAcquisitionCriteria(
    const AcMapDMDataSourceDescriptor* pDataSourceDsc);
```

**Parameters**

- `pDataSourceDsc`: Input pointer to a data-source descriptor.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

**Remarks**

This function determines which entities become part of this element.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

```cpp
virtual ~AcMapDMGroup();
```

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMGroup Class, AcMapDMGroup Class
AcMapDMGroup:: AcMapDMGroup Constructor
AcMapDMGroup Class | AcMapDMGroup Class

Constructs an instance of this class.

AcMapDMGroup();
Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
This function is called by AutoCAD when the AUDIT command is executed.

```cpp
virtual Acad::ErrorStatus audit(
    AcDbAuditInfo* pAuditInfo
);
```

**Parameters**

- **pAuditInfo**
  
  The AcDbAuditInfo object pointed to by pAuditInfo contains member functions that are used to determine what to do and also to report the results of the audit operation on the object.

**Returns**

Returns Acad::eOk if successful, Acad::eFixedAllErrors if all errors were corrected, Acad::eLeftErrorsUnfixed if there were errors that were left unfixed; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Clears the current stylization.

```cpp
virtual Acad::ErrorStatus DismissStylization(
    bool bDismissEntities
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bDismissEntities</td>
<td>Input true to clear the acquired entities.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMGroup Class, AcMapDMGroup Class
AcMapDMGroup:: dwgInFields Method
AcMapDMGroup Class | AcMapDMGroup Class

Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dwgInFields(
    AcDbDwgFiler* pFiler
);
```

**Parameters**

- **pFiler**: Input filer to use to read the object's data.

**Returns**

- Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.doc-o-matic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapDMGroup Class, AcMapDMGroup Class
AcMapDMGroup:: dwgOutFields Method
AcMapDMGroup Class | AcMapDMGroup Class

Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dwgOutFields(
    AcDbDwgFiler* pFiler
) const;
```

Parameters Description
---
pFiler Input filer to use to write the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMGroup Class, AcMapDMGroup Class
AcMapDMGroup:: dxfInFields Method
AcMapDMGroup Class | AcMapDMGroup Class

Lets this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dxfInFields(
    AcDbDxfFiler* pFiler
);
```

Parameters

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pFiler</td>
</tr>
</tbody>
</table>

Input filer to use to read the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links

AcMapDMGroup Class, AcMapDMGroup Class
AcMapDMGroup:: dxfOutFields Method
AcMapDMGroup Class | AcMapDMGroup Class

Lets this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dxfOutFields(AcDbDxfFiler* pFiler) const;
```

**Parameters**

- `pFiler` Input filer to use to write the object's data.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links

AcMapDMGroup Class, AcMapDMGroup Class

AcMapDMGroup:: erased Method

AcMapDMGroup Class | AcMapDMGroup Class

Lets this object listen to erase-notifications from items that it owns. See also erased() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual void erased(
    const AcDbObject* dbObj,
    Adesk::Boolean pErasing = true
);
```

Parameters

- **dbObj**
  - Description: Input object that was erased.

- **pErasing**
  - Description: Input true to listen as an erase is happening. The default value is true.

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determine whether children are visible or invisible at a specified scale.

```cpp
bool HasChildrenInSpecifiedVisibleState(
    bool bState,
    double dScale
);
```

**Parameters**

- `bState`: Input true to check visibility.
- `dScale`: Input scale.

**Returns**

Returns true if children are visible at the specified scale; otherwise, returns false.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determine whether this group is empty.

\texttt{virtual bool IsEmpty() const;}

Returns true if the group is empty.

Created with a commercial version of \texttt{Doc-O-Matic}. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at \texttt{support@toolsfactory.com}. 

Links
\texttt{AcMapDMGroup Class, AcMapDMGroup Class}
\texttt{AcMapDMGroup:: IsEmpty Method}
\texttt{AcMapDMGroup Class | AcMapDMGroup Class}
Determines whether this group is visible at the specified scale. See also SetVisible() and HasChildrenInSpecifiedVisibleState().

```cpp
virtual bool IsVisible(
    double dScale
) const;
```

**Parameters**

- **Input scale at which visibility is evaluated.**

**Returns**

Returns true if the group is visible at the specified scale; otherwise, returns false.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves a new iterator over the contents of this group.

```cpp
AcMapDMAAllItemsIterator* NewAllItemsIterator(
    bool bForward = true,
    bool bSkipProxies = false
);
```

**Parameters**

- `bForward`: Input true for a forward traversal.
- `bSkipProxies`: Input true to skip proxy Items

**Returns**

Returns a new AcMapDMAAllItemsIterator if successful, or NULL if unsuccessful.

**Remarks**

This iterator will step on every Item, even if it is in a proxy state, unless true is passed in the `bSkipProxies` argument.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Invoked when an AutoCAD Map project is initialized.

```cpp
virtual bool OnMapProjectInitialized(
    AcMapProject* pMapProject
);
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pMapProject</td>
<td>Input AutoCAD Map project.</td>
</tr>
</tbody>
</table>

Returns

Returns true if successful.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Invoked when an object is appended to the database.

```cpp
virtual Acad::ErrorStatus OnObjectAppended();
```

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMGroup Class, AcMapDMGroup Class
AcMapDMGroup:: RemoveItem Method
AcMapDMGroup Class | AcMapDMGroup Class

Removes an item from this group.

virtual Acad::ErrorStatus RemoveItem(
    const AcDbObjectId& Id
);

Parameters Description
Id Input ID of the item to remove.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the IDs of the elements that have references to the specified style at the specified scale.

```cpp
virtual Acad::ErrorStatus SelectElementsWithStyleApplied(
    AcDbObjectIdArray& aElementIds,
    const AcDbObjectId& styleId,
    double dDisplayScale,
    bool bSkipDisabled = true
);
```

**Parameters**

- **aElementIds**: Output IDs of the elements that refer to the style.
- **styleId**: Input ID of the AcMapDMStyle object.
- **dDisplayScale**: Input display scale.
- **bSkipDisabled**: Input true to skip styles that are disabled at the specified scale. The default value is true.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the group visibility at the specified scale. See also IsVisible() and HasChildrenInSpecifiedVisibleState().

```cpp
virtual Acad::ErrorStatus SetVisible(
    bool bNewVal,
    double dScale
);
```

**Parameters**

- **bNewVal**
  - Input true to make the group visible at the specified scale, or false to make it invisible.
- **dScale**
  - Input scale at which visibility is set.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://example.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Invoked from within erase() before the erase actually occurs. See also subErase() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```
virtual Acad::ErrorStatus subErase(
   Adesk::Boolean erasing
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Passed-in copy of the erasing argument that was passed to the erase() function call that triggered this subErase() call.</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Stylizes the current or updated selection at the current scale.

```cpp
virtual Acad::ErrorStatus UpdateStylization(
    bool bAcquireEntities
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bAcquireEntities</td>
<td>Input true to acquire entities.</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful. Returns Acad::eAmbiguousOutput if bAcquireEntities is true but the previous selection has not been dismissed.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Enumerates the levels of legend detail.

```cpp
enum ELegendDetailLevel {
    kLegendExpanded = 1,
    kLegendDefault = kLegendExpanded,
    kLegendCompressed
};
```

File

DmDisplayItem.h

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>kLegendExpanded</td>
<td>Expanded.</td>
</tr>
<tr>
<td>kLegendDefault</td>
<td>Default (kLegendExpanded).</td>
</tr>
<tr>
<td>kLegendCompressed</td>
<td>Compressed.</td>
</tr>
</tbody>
</table>

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Destroys an instance of this class.

```cpp
virtual ~AcMapDMItem();
```

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Constructs an instance of this class.

AcMapDMItem();

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
This function is called by AutoCAD when the AUDIT command is executed.

```cpp
virtual Acad::ErrorStatus audit(
    AcDbAuditInfo* pAuditInfo
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pAuditInfo</td>
<td>The AcDbAuditInfo object pointed to by pAuditInfo contains member functions that are used to determine what to do and also to report the results of the audit operation on the object.</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful, Acad::eFixedAllErrors if all errors were corrected, Acad::eLeftErrorsUnfixed if there were errors that were left unfixed; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Clears the current stylization.

```cpp
virtual Acad::ErrorStatus DismissStylization(
    bool bDismissEntities
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bDismissEntities</td>
<td>Input true to clear the acquired entities.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapDMItem Class, AcMapDMItem Class
AcMapDMItem:: dwgInFields Method
AcMapDMItem Class | AcMapDMItem Class

Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

virtual Acad::ErrorStatus dwgInFields(
    AcDbDwgFiler* pFiler
);

Parameters Description
pFiler Input filer to use to read the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMItem Class, AcMapDMItem Class
AcMapDMItem:: dwgOutFields Method
AcMapDMItem Class | AcMapDMItem Class

Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dwgOutFields(
    AcDbDwgFiler* pFiler
) const;
```

Parameters

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pFiler</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMItem Class, AcMapDMItem Class
AcMapDMItem:: dxfInFields Method
AcMapDMItem Class | AcMapDMItem Class

Lets this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

virtual Acad::ErrorStatus dxfInFields(
    AcadDxfFiler* pFiler
);

Parameters Description
pFiler Input filer to use to read the object's data.

Returns
Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMItem Class, AcMapDMItem Class
AcMapDMItem::dxfOutFields Method
AcMapDMItem Class | AcMapDMItem Class

Lets this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dxfOutFields(
    AcDbDxfFiler* pFiler
) const;
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pFiler</td>
<td>Input filer to use to write the object's data.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the name of this item. See also SetName().

\textbf{virtual const} ACHAR* GetName() \textbf{const};

Returns the item name if successful; otherwise, returns NULL.

Created with a commercial version of \textbf{Doc-O-Matic}. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at \texttt{support@toolsfactory.com}.
Returns the implementation object.

```cpp
virtual AcMapDMImpItem* Implementation();
```

Returns the implementation, or NULL if unsuccessful.

Remarks

This function is for internal use only; do not call it directly.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determines whether this item is visible at the specified scale. See also SetVisible().

```cpp
virtual bool IsVisible(
    double dScale
) const;
```

Parameters

- **dScale**: Input scale at which visibility is evaluated.

Returns

Returns true if the element is visible at the specified scale; otherwise, returns false.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the level of legend detail of this item. See also SetLegendDetailLevel().

ELegendDetailLevel LegendDetailLevel() const;

Returns

Returns the ELegendDetailLevel value.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the level of legend detail of this item. See also LegendDetailLevel().

```cpp
void SetLegendDetailLevel(
    ELegendDetailLevel eLevel
);
```

Parameters

<table>
<thead>
<tr>
<th></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>eLevel</td>
<td>Input ELegendDetailLevel value.</td>
</tr>
</tbody>
</table>

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMItem Class, AcMapDMItem Class
AcMapDMItem:: MapId Method
AcMapDMItem Class | AcMapDMItem Class

Retrieves the map ID.

virtual AcDbObjectId MapId() const;

Returns

Returns the map ID.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the AutoCAD Map project.

\texttt{AcMapProject* MapProject();}

Returns

Returns the AutoCAD Map project.

Created with a commercial version of \texttt{Doc-O-Matic}. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at \texttt{support@toolsfactory.com}.
Invoked when an AutoCAD Map project is initialized.

```cpp
virtual bool OnMapProjectInitialized(
    AcMapProject* pMapProject)
```

**Parameters**
- `pMapProject` Input AutoCAD Map project.

**Returns**
Returns true if successful.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Invoked when an object is appended to the database.

```cpp
virtual Acad::ErrorStatus OnObjectAppended();
```

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the item visibility at the specified scale. See also IsVisible().

```cpp
virtual Acad::ErrorStatus SetVisible(
    bool bNewVal,
    double dScale
);
```

**Parameters**

- **bNewVal**
  - Description: Input true to make the item visible at the specified scale, or false to make it invisible.
- **dScale**
  - Description: Input scale at which visibility is set.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](http://www.docomaticsoft.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Invoked from within close() before the close actually occurs. The default implementation of this function returns Acad::eOk. See also subClose() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus subClose();
```

Returns Acad::eOk if successful.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMItem Class, AcMapDMItem Class
AcMapDMItem:: subErase Method
AcMapDMItem Class | AcMapDMItem Class

Invoked from within erase() before the erase actually occurs. See also subErase() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus subErase(
    Adesk::Boolean erasing
);
```

Parameters  Description
            Passed-in copy of the erasing argument that was passed to the erase() function call that triggered this subErase() call.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Stylizes the current or updated selection at the current scale.

```cpp
virtual Acad::ErrorStatus UpdateStylization(
    bool bAcquireEntities);
```

**Parameters**

- `bAcquireEntities` Input true to acquire entities.

**Returns**

Returns Acad::eOk if successful. Returns Acad::eAmbiguousOutput if `bAcquireEntities` is true but the previous selection has not been dismissed.

**Remarks**

Do not override this function.

Created with a commercial version of [Doc-O-Matic](#). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Grants control of deep clone operations to the object. In the default implementation, the object is cloned and appended to the owner object pOwnerObject. See also wblockClone() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus wblockClone(
    AcRxObject* pOwnerObject,
    AcDbObject*& pClonedObject,
    AcDbIdMapping& idMap,
    Adesk::Boolean isPrimary = true
) const;
```

**Parameters**

- **pOwnerObject**
  Input object to append the clones to.

- **pClonedObject**
  Output the cloned object, or NULL if not cloned.

- **idMap**
  Input current ID map.

- **isPrimary**
  Input true if this object is primary, or false if it is owned.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

```
virtual ~AcMapDMLayerDataSourceDescriptor();
```

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Constructs an instance of this class.

AcMapDMLayerDataSourceDescriptor();

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

**virtual ~AcMapDMLayerElement();**

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://toolsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links

AcMapDMLayerElement Class, AcMapDMLayerElement Class

AcMapDMLayerElement:: AcMapDMLayerElement Constructor

AcMapDMLayerElement Class | AcMapDMLayerElement Class

Constructs an instance of this class.

AcMapDMLayerElement();

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links

AcMapDMLayerElement Class, AcMapDMLayerElement Class
AcMapDMLayerElement:: AcquireEntities Method
AcMapDMLayerElement Class | AcMapDMLayerElement Class

Runs an ADE query against the drawing scope defined in the data-source descriptor.

```
virtual Acad::ErrorStatus AcquireEntities();
```

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Clones objects from the source drawings.

\texttt{virtual bool ClonesObjectsFromExternalSource()} \texttt{const};

Returns false.

Created with a commercial version of \texttt{Doc-O-Matic}. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at \texttt{support@toolsfactory.com}.
Links
AcMapDMLayerElement Class, AcMapDMLayerElement Class
AcMapDMLayerElement:: dwgInFields Method
AcMapDMLayerElement Class | AcMapDMLayerElement Class

Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dwgInFields(
   AcDbDwgFiler* pFiler
);
```

Parameters Description
pFiler Input filer to use to read the object's data.

Returns
Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMLayerElement Class, AcMapDMLayerElement Class
AcMapDMLayerElement:: dwgOutFields Method
AcMapDMLayerElement Class | AcMapDMLayerElement Class

Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dwgOutFields(
    AcDbDwgFiler* pFiler
) const;
```

Parameters Description
pFiler Input filer to use to write the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMLayerElement Class, AcMapDMLayerElement Class
AcMapDMLayerElement:: dxfInFields Method
AcMapDMLayerElement Class | AcMapDMLayerElement Class

Lets this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dxfInFields(
    AcDbDxfFiler* pFiler
);
```

Parameters Description
pFiler Input filer to use to read the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMLayerElement Class, AcMapDMLayerElement Class
AcMapDMLayerElement:: dxfOutFields Method
AcMapDMLayerElement Class | AcMapDMLayerElement Class

Lets this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dxfOutFields(
    AcDbDxfFiler* pFiler
) const;
```

Parameters Description
pFiler Input filer to use to write in the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the query definition's data-source descriptor.

```cpp
virtual Acad::ErrorStatus GetAcquisitionCriteria(
    AcMapDMDataSourceDescriptor* pDataSourceDsc)
const;
```

**Parameters**
- `pDataSourceDsc`: Output pointer to the data-source descriptor.

**Returns**
- Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Invoked when an AutoCAD Map project is initialized.

```cpp
virtual bool OnMapProjectInitialized(
    AcMapProject* pMapProject
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pMapProject</td>
<td>Input AutoCAD Map project.</td>
</tr>
</tbody>
</table>

Returns true if successful.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the query definition's data-source descriptor.

```cpp
virtual Acad::ErrorStatus SetAcquisitionCriteria(
    const AcMapDMDataSourceDescriptor* pDataSourceDsc);
```

Parameters

<table>
<thead>
<tr>
<th>Description</th>
<th>pDataSourceDsc</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input pointer to a data-source descriptor.</td>
<td></td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Remarks

This function determines which entities become part of this element.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMLegend Class, AcMapDMLegend Class
AcMapDMLegend:: ~AcMapDMLegend Destructor
AcMapDMLegend Class | AcMapDMLegend Class

Destroys an instance of this class.

virtual ~AcMapDMLegend();

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMLegend Class, AcMapDMLegend Class
AcMapDMLegend::: AcMapDMLegend Constructor
AcMapDMLegend Class | AcMapDMLegend Class

Constructs an instance of this class.

AcMapDMLegend();
Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMLegend Class, AcMapDMLegend Class
AcMapDMLegend::: DismissTableContents Method
AcMapDMLegend Class | AcMapDMLegend Class

Clears the contents of the legend.

virtual Acad::ErrorStatus DismissTableContents();

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Let's this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dwgInFields(
    AcDbDwgFiler* pFiler
);
```

**Parameters**

- **pFiler**: Input filer to use to read the object's data.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapDMLegend Class, AcMapDMLegend Class
AcMapDMLegend:: dwgOutFields Method
AcMapDMLegend Class | AcMapDMLegend Class

Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```
virtual Acad::ErrorStatus dwgOutFields(
    AcDbDwgFiler* pFiler
) const;
```

Parameters
pFiler Input filer to use to write the object's data.

Returns
Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMLegend Class, AcMapDMLegend Class

AcMapDMLegend:: dxfInFields Method
AcMapDMLegend Class | AcMapDMLegend Class

Lets this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dxfInFields(
    AcDbDxfFiler* pFiler
);
```

Parameters

- **pFiler**: Input filer to use to read the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMLegend Class, AcMapDMLegend Class
AcMapDMLegend:: dxfOutFields Method
AcMapDMLegend Class | AcMapDMLegend Class

Lets this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dxfOutFields(
    AcDbDxfFiler* pFiler
) const;
```

**Parameters**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pFiler</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMLegend Class, AcMapDMLegend Class
AcMapDMLegend:: Implementation Method
AcMapDMLegend Class | AcMapDMLegend Class

Returns the implementation object.

```cpp
virtual AcMapDMImpLegend* Implementation();
```

Returns the implementation object, or NULL if unsuccessful.

Remarks

This function is for internal use only; do not call it directly.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Clears the contents of the legend and updates it with current data from the display manager.

```cpp
virtual Acad::ErrorStatus UpdateTableContents(
    AcMapDMMap& map,
    bool bFirstUpdate
);
```

### Parameters

- **map**: Input `AcMapDMMap` containing the legend. This parameter indicates how the size of the legend is calculated. A true value will set the width of the legend to be 20% of the current viewport width. A false value will retain the legend's current width. In either case, the height of the legend is determined by the contents loaded from the display manager.

### Returns

Returns `Acad::eOk` if successful; otherwise, returns a different error code.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at `support@toolsfactory.com`.
This is record AcMapDMMap::DOMode.

```cpp
enum DOMode {
    kDOTextOnTop = 0x0001,
    kDOHatchOnBottom = 0x0002
};
```

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

```cpp
virtual ~AcMapDMMap();
```

Returns nothing.

Created with a commercial version of [Doc-O-Matic](http://www.dcom.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
**AcMapDMMAP Class**, [AcMapDMMAP Class](#)

AcMapDMMAP:: AcMapDMMAP Constructor
**AcMapDMMAP Class** | [AcMapDMMAP Class](#)

Constructs an instance of this class.

```
AcMapDMMAP();
```

Returns

Returns nothing.

Created with a commercial version of [Doc-O-Matic](#). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
This function is called by AutoCAD when the AUDIT command is executed.

```cpp
virtual Acad::ErrorStatus audit(
    AcDbAuditInfo* pAuditInfo
);
```

**Parameters**

- **pAuditInfo**
  - The `AcDbAuditInfo` object pointed to by `pAuditInfo` contains member functions that are used to determine what to do and also to report the results of the audit operation on the object.

**Returns**

- Returns `Acad::eOk` if successful, `Acad::eFixedAllErrors` if all errors were corrected, `Acad::eLeftErrorsUnfixed` if there were errors that were left unfixed; otherwise, returns a different error code.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
AcMapDMMap Class, AcMapDMMap Class
AcMapDMMap:: CreateLegend Method
AcMapDMMap Class | AcMapDMMap Class

Creates a new legend for the specified layout.

```cpp
Acad::ErrorStatus CreateLegend(
    AcMapDMLegend*& pLegend,
    AcDbObjectId layoutId = 0
);
```

Parameters

<table>
<thead>
<tr>
<th>pLegend</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Output created AcMapDMLegend.</td>
</tr>
</tbody>
</table>

| layoutId | Input ID of the layout. |

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Remarks

If the legend already exists, it is overwritten.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Clones this object and all the objects that it refers to.

```cpp
static Acad::ErrorStatus deepCloneObject(
    AcDbObjectId& newId,
    AcDbObjectId& existingId
);
```

**Parameters**

- **newId**: Output ID of the cloned object.
- **existingId**: Input ID of an existing object.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Clones this object and all the objects that it refers to.

```cpp
static Acad::ErrorStatus deepCloneObjects(
    AcDbIdMapping& idMap,
    AcDbObjectIdArray& existingIdArray
);
```

**Parameters**

- **idMap**: Output ID mapping.
- **existingIdArray**: Input array of IDs of existing objects.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Deletes the contents of the legend of the specified layout.

```cpp
Acad::ErrorStatus DeleteLegend(
    AcDbObjectId layoutId = 0
);
```

**Parameters**

- `layoutId`  
  Description: Input ID of the layout.

**Returns**

- Returns Acad::eOk if successful. Returns Acad::eKeyNotFound if no legend exists for the specified layout.

**Remarks**

This function deletes the legend itself, and removes the layout from the internal list of layout/legend pairs.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Clears the current stylization.

```cpp
virtual Acad::ErrorStatus DismissStylization(
    bool bDismissEntities
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bDismissEntities</td>
<td>Input true to clear the acquired entities.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.doc-o-matic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Let this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dwgInFields(
    AcDbDwgFiler* pFiler
);
```

**Parameters**
- `pFiler` Input filer to use to read the object's data.

**Returns**
- Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Let's this object write its data. See also `dwgOutFields()` in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dwgOutFields(
    AcDbDwgFiler* pFiler
) const;
```

**Parameters**
- `pFiler` Input filer to use to write the object's data.

**Returns**
- Returns `Acad::eOk` if successful; otherwise, returns a different error code.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapDMMap Class, AcMapDMMap Class
AcMapDMMap::dxfInFields Method
AcMapDMMap Class | AcMapDMMap Class

Lets this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dxfInFields(
    AcDbDxfFiler* pFiler
);
```

Parameters

- `pFiler` Input filer to use to read the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMMap Class, AcMapDMMap Class
AcMapDMMap:: dxfOutFields Method
AcMapDMMap Class | AcMapDMMap Class

Lets this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```c++
virtual Acad::ErrorStatus dxfOutFields(
    AcDbDxfFiler* pFiler
) const;
```

Parameters

- **pFiler**: Input filer to use to write the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMMap Class, AcMapDMMap Class
AcMapDMMap:: erased Method
AcMapDMMap Class | AcMapDMMap Class

Lets this object listen to erase-notifications from items that it owns. See also erased() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```
virtual void erased(
    const AcDbObject* dbObj,
    Adesk::Boolean pErasing = true
);
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dbObj</td>
<td>Input object that was erased.</td>
</tr>
<tr>
<td>pErasing</td>
<td>Input true to listen as an erase is happening. The default value is true.</td>
</tr>
</tbody>
</table>

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the draw-order iterator. The first function call iterates elements in the order that they are arranged in the map tree. This iterator may return Items that are in a proxy state, unless the bSkipProxies argument is passed as true. See also IsDrawOrderDefined().

```csharp
AcMapDMAllDrawOrderItemsIterator* GetAllDrawOrderItemsIterator(
    bool bForward = true,
    bool bSkipProxies = false
);
```

**Parameters**

- **bForward**  
  Input true to traverse forward. The default value is true.

- **bSkipProxies**  
  Input true to skip over proxy Items. The default value is false;

**Returns**

Returns the `AcMapDMAllDrawOrderItemsIterator`.

Created with a commercial version of [Doc-O-Matic](mailto:). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:).
Gets the draw order mode currently in use for the map.

```cpp
int GetDOMode() const;
```

Returns

A bit flag containing the current draw order mode of the map. The bits are one of the values in DOMode enumeration.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the legend of the specified layout.

\[
\begin{align*}
\text{Acad::ErrorStatus} & \quad \text{GetLegend}(
\quad \text{AcMapDMLegend*} \& \ pLegend,
\quad \text{AcDb::OpenMode} \quad \text{openMode},
\quad \text{AcDbObjectId} \quad \text{layoutId} = 0
\); \\
\text{Parameters} & \quad \text{Description} \\
pLegend & \quad \text{Output AcMapDMLegend object.} \\
\text{openMode} & \quad \text{Input open mode (typically AcDb::kForWrite).} \\
\text{layoutId} & \quad \text{Input ID of the layout.}
\end{align*}
\]

Returns

Returns Acad::eOk if successful. Returns Acad::eKeyNotFound if no legend exists for the specified layout.

Created with a commercial version of \text{Doc-O-Matic}. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at \text{support@toolsfactory.com}.
Retrieves the legend of the specified layout.

```cpp
 Acad::ErrorStatus GetLegendId(
   AcDbObjectId& legendId,
   AcDbObjectId layoutId = 0
);
```

**Parameters**

- `legendId`: Output ID of the legend.
- `layoutId`: Input ID of the layout.

**Returns**

Returns `Acad::eOk` if successful. Returns `Acad::eKeyNotFound` if no legend exists for the specified layout.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMMap Class, AcMapDMMap Class
AcMapDMMap:: GetLineTypeScale Method
AcMapDMMap Class | AcMapDMMap Class

Retrieves the linetype scale applied to the stylization. The linetype scale, a positive value, is assigned to the AutoCAD system variable LTSCALE at update time. See also SetLineTypeScale().

```cpp
double GetLineTypeScale(
    double dScale = 0.)
) const;
```

Parameters
- **dScale**: Input threshold scale value.

Returns
- Returns the linetype scale.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the name of this map. See also SetName().

```cpp
virtual const ACHAR* GetName() const;
```

Returns the map name if successful; otherwise, returns NULL.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the table style. See also SetTableStyle().

```cpp
const ACHAR* GetTableStyle() const;
```

Returns the name of the table style.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMMap Class, AcMapDMMap Class
AcMapDMMap:: GetThumbnailDimensions Method
AcMapDMMap Class | AcMapDMMap Class

Retrieves the thumbnail width and height factors. These width and height factors are multiplied by the text height in the legend to determine the size of the thumbnail width and height. See also SetThumbnailDimensions().

```cpp
void GetThumbnailDimensions(
    double& dHeight,
    double& dWidth
) const;
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dHeight</td>
<td>Output height factor.</td>
</tr>
<tr>
<td>dWidth</td>
<td>Output width factor.</td>
</tr>
</tbody>
</table>

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the thumbnail width and height factors. These width and height factors are multiplied by the text height in the legend to determine the size of the thumbnail width and height. See also `GetThumbnailDimensions()`.

```cpp
Acad::ErrorStatus SetThumbnailDimensions(
  double dHeight,
  double dWidth
);
```

Parameters

dHeight Input height factor.
dWidth Input width factor.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Determines whether a box is drawn around the thumbnail graphic. See also SetThumbnailFraming().

```cpp
bool GetThumbnailFraming() const;
```

Returns true if a box is drawn.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determines whether the draw order is defined. Draw order is defined on demand when GetDrawOrderIterator()is first called. Draw-order functionality is available in Display Manager user interface in the list view.

```cpp
bool IsDrawOrderDefined() const;
```

Returns

Returns true if draw order is defined.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMMap Class, AcMapDMMap Class
AcMapDMMap:: IsLinkToFileEnabled Method
AcMapDMMap Class | AcMapDMMap Class

Determines whether linked-file capability is enabled.

```cpp
bool IsLinkToFileEnabled(
    double dScale = 0.
) const;
```

Parameters

<table>
<thead>
<tr>
<th>Description</th>
<th>Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input threshold scale value.</td>
<td>dScale</td>
</tr>
</tbody>
</table>

Returns

Returns true if the capability is enabled.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determines whether this map is stylized.

```cpp
bool IsStylized() const;
```

Returns true if the map is stylized.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Determined whether this map is visible at the specified scale. See also SetVisible().

```cpp
virtual bool IsVisible(
    double dScale
) const;
```

Parameters

- **dScale**: Input scale threshold at which visibility is evaluated.

Returns

Returns true if the map is visible at the specified scale; otherwise, returns false.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the ID of this map.

```cpp
virtual AcDbObjectId MapId() const;
```

Returns

Returns the map ID.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Invoked when an AutoCAD Map project is initialized.

```cpp
virtual bool OnMapProjectInitialized(
    AcMapProject* pMapProject
);
```

### Parameters

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pMapProject</td>
</tr>
</tbody>
</table>

### Returns

Returns true if successful.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapDMMap Class, AcMapDMMap Class
AcMapDMMap:: OnObjectAppended Method
AcMapDMMap Class | AcMapDMMap Class

Invoked when an object is appended to the database.

\texttt{virtual Acad::ErrorStatus OnObjectAppended();}

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of \texttt{Doc-O-Matic}. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMMap Class, AcMapDMMap Class
AcMapDMMap:: OnStyleModified Method
AcMapDMMap Class | AcMapDMMap Class

Makes the system to refresh the contents of map when the given style is modified.

```
 Acad::ErrorStatus OnStyleModified(
    AcDbObjectId styleId
);
```

Parameters
- **styleId**
  
  Input id of modified style.

Returns
Returns Acad::eOk if successful; otherwise, returns a different error code.

Remarks

This method is meant to be called by an application whenever a custom style is modified so that the system could invoke appropriate methods on the custom style to allow it to update itself.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMMAP Class, AcMapDMMAP Class
AcMapDMMAP:: Requery Method
AcMapDMMAP Class | AcMapDMMAP Class

Requeries objects belonging to the map.

`Acad::ErrorStatus Requery();`

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Remarks

This method erases previous selection and requeries to obtain new set of objects.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Resumes fixing of the draw order of entities. The draw order is fixed during this call. Any change that affects the draw order after this method is called until the next SuspendFixingDO method will be reflected immediately. This method must be used in conjunction with SuspendFixingDO() and these two methods must always be paired.

```c++
int ResumeFixingDO(
    bool bFixDO = true
);
```

**Parameters**

- `bFixDO`  
  Description: Input true if draw order must be fixed.

**Returns**

Returns the outstanding reference counts before fixing draw order can be resumed. If `bFixDO` is true, the draw order will be fixed right away if SuspendFixingDO/ResumeFixingDO calls match. If it's false, then the reference count maintained to track Suspend/Resume calls is decremented by one and draw order is not fixed.

Created with a commercial version of [Doc-O-Matic](mailto:info@Doc-O-Matic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Saves the current database in its current state of stylization at the current scale to the filename specified in a previous call to `AcMapDMMap::SetLinkedFileName()`. The linked file contains a stylized map that can be plotted or published to DWF. Clicking the Update button in the AutoCAD Map user interface invokes `SaveLinkedFile()`. `SaveLinkedFile()` does not call `AcMapDMMap::UpdateStylization();` it is your application's responsibility to prepare the stylized model. `SaveLinkedFile()` calls `AcMapDMMap::GetCurrentScale()`, followed by `AcMapDMMap::GetLinkedFileName()`, with the current scale to obtain the target filename.

```
Acad::ErrorStatus SaveLinkedFile(
    void
);
```

**Parameters**  
Void.

**Returns**  
Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Stylizes the current or updated selection at the current scale.

```cpp
virtual Acad::ErrorStatus UpdateStylization(
    bool bAcquireEntities
);
```

Parameters

- **bAcquireEntities**: Input true to acquire entities.

Returns

Returns Acad::eOk if successful. Returns Acad::eAmbiguousOutput if bAcquireEntities is true but the previous selection has not been dismissed.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMMap Class, AcMapDMMap Class
AcMapDMMap:: SelectElementsWithStyleApplied Method
AcMapDMMap Class | AcMapDMMap Class

Retrieves elements that reference the specified style at the specified scale threshold.

```cpp
virtual Acad::ErrorStatus SelectElementsWithStyleApplied(
    AcDbObjectIdArray& aElementIds,
    const AcDbObjectId& styleId,
    double dDisplayScale,
    bool bSkipDisabled = true);
```

Parameters

- **aElementIds**: Output IDs of the elements that refer to the style.
- **styleId**: Input AcMapDMSStyle object ID.
- **dDisplayScale**: Input display scale.
- **bSkipDisabled**: Input true to skip styles that are disabled at the specified scale. The default value is true.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the draw order mode for the map.

```c
void SetDOMode(
    int mode
);
```

Parameters | Description
--- | ---
mode | A bit flag containing the desired draw order mode.

Returns

Nothing.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the linetype scale for a specified scale. See also GetLineTypeScale().

```cpp
Acad::ErrorStatus SetLineTypeScale(
    double sLinetypeScale,
    double dScale = 0.
);
```

Parameters

- **sLinetypeScale**: Input linetype scale to set.
- **dScale**: Input threshold scale value.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.
Sets the status of the linked file capability.

```cpp
Acad::ErrorStatus SetLinkToFileEnabled(
    bool bNewValue,
    double dScale = 0.
);
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bNewValue</td>
<td>Input true to enable, or false to disable, the capability.</td>
</tr>
<tr>
<td>dScale</td>
<td>Input threshold scale value.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets whether a box is drawn around the thumbnail graphic. See also GetThumbnailFraming().

```cpp
void SetThumbnailFraming(
    bool bDrawFrame
);
```

Parameters  

<table>
<thead>
<tr>
<th>bDrawFrame</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input true to draw a box.</td>
<td></td>
</tr>
</tbody>
</table>

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the map visibility at the specified scale. See also IsVisible().

```cpp
virtual Acad::ErrorStatus SetVisible(
    bool NewVal,
    double dScale
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input true to make the map visible at the specified scale, or false to make it invisible.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input scale threshold at which visibility is set.</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Invoked from within erase() before the erase actually occurs. See also subErase() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus subErase(
    Adesk::Boolean erasing
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Passed-in copy of the erasing argument that was passed to the erase() function call that triggered this subErase() call.</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Suspends fixing of the draw order of entities.

```c
int SuspendFixingDO();
```

Returns

Returns the outstanding reference counts before fixing DO can be resumed.

Remarks

Any change in draw order after this call is made is not effected until ResumeFixingDO method is called. Users of this method are strongly advised to correctly pair the Suspend/Resume calls. If this method is not called (later followed by ResumeFixingDO), then any change in DO is immediately fixed and displayed. This method is recommended for use during changes (addition, moving, / etc.) involving multiple items.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

```cpp
virtual ~AcMapDMMMapIterator();
```

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determine whether the iterator has reached the end of the collection.

```
virtual bool Done() const = 0;
```

Returns

Returns true if the iterator has reached the end of the collection; otherwise, returns false.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the current object in the iteration.

```cpp
virtual Acad::ErrorStatus GetObj(
    AcMapDMMap*& pMap,
    AcDb::OpenMode mode
) const = 0;
```

**Parameters**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pMap</td>
</tr>
<tr>
<td>mode</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Advances to the next element in the iteration.

```
virtual bool Next() = 0;
```

Returns true if successful; otherwise, returns false.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMMapIterator Class, AcMapDMMapIterator Class
AcMapDMMapIterator::: ObjectId Method
AcMapDMMapIterator Class | AcMapDMMapIterator Class

Retrieves the ID of the current object in the iteration.

```cpp
def ObjectId() const { return 0; }
```

Returns a valid ID if successful.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMMapManager Class, AcMapDMMapManager Class
AcMapDMMapManager:: ~AcMapDMMapManager Destructor
AcMapDMMapManager Class | AcMapDMMapManager Class

Destroys an instance of this class.

*virtual* ~AcMapDMMapManager();

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Constructs an instance of this class.

AcMapDMMapManager();

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMMapManager Class, AcMapDMMapManager Class
AcMapDMMapManager:: AddDefaultMap Method
AcMapDMMapManager Class | AcMapDMMapManager Class

Creates a default map object.

```
Acad::ErrorStatus AddDefaultMap(
    AcDbObjectId& Id
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Id</td>
<td>Output ID of the default map.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMMapManager Class, AcMapDMMapManager Class
AcMapDMMapManager:: CreateNewMap Method
AcMapDMMapManager Class | AcMapDMMapManager Class

Adds a new map with the default settings.

Acad::ErrorStatus CreateNewMap(
    AcDbObjectId& Id
);

Parameters Description
Id Output ID of the map.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Remarks

The new map is named "New_Map#", where # is a serial number starting at 1.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links

AcMapDMMapManager Class, AcMapDMMapManager Class
AcMapDMMapManager::: dwgInFields Method
AcMapDMMapManager Class | AcMapDMMapManager Class

Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dwgInFields(
    AcDbDwgFiler* pFiler
);
```

Parameters

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pFiler</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Let's this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dwgOutFields(
    AcDbDwgFiler* pFiler
) const;
```

### Parameters

- **pFiler**
  - Input filer to use to write the object's data.

### Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMMapManager Class, AcMapDMMapManager Class
AcMapDMMapManager::: dxfInFields Method
AcMapDMMapManager Class | AcMapDMMapManager Class

Lets this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dxfInFields(
    AcDbDxfFiler* pFiler
);
```

Parameters

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pFiler</td>
</tr>
</tbody>
</table>

Input filer to use to read the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Let this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dxfOutFields(
    AcDbDxfFiler* pFiler
) const;
```

**Parameters**

- pFiler: Input filer to use to write in the object's data.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the ID of the current map.

```
Acad::ErrorStatus GetCurrent(
    AcDbObjectId& Id
) const;
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Id</td>
<td>Output ID of the current map ID.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determines whether a map with the specified ID exists.

```cpp
bool Has(
    AcDbObjectId objId
) const;
```

Parameters

- **objId**
  - Input ID of the map to search for.

Returns

Returns true if the map exists.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns the implementation object.

\begin{verbatim}
AcMapDMImpMapManager* Implementation();
\end{verbatim}

Returns the implementation object, or NULL if unsuccessful.

Remarks

This function is for internal use only; do not call it directly.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns a new map iterator.

```c
AcMapDMMapIterator* NewIterator();
```

Returns an `AcMapDMMapIterator` if successful; otherwise, returns NULL.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Counts the number of maps.

Adesk::UInt32 NumMaps() const;

Returns the number of maps.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Removes the map with the specified ID.

```cpp
Acad::ErrorStatus Remove(
    AcDbObjectId Id
);
```

**Parameters**

- **Id**: Input ID of the map to remove.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

**Remarks**

The default map cannot be removed. If the current map is removed, the default map becomes the current map.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the map with the specified ID to be the current map.

```cpp
Acad::ErrorStatus SetCurrent(
    const AcDbObjectId& Id
);
```

**Parameters**

- **Id**  
  Input ID of the map.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Clones the specified map to the destination database. See also wblockClone() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
static Acad::ErrorStatus wblockClone(
    const AcDbObjectId& mapId,
    AcDbDatabase* pDestDb
);
```

**Parameters**

- **mapId**: Input map ID.
- **pDestDb**: Input destination database.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Deep clones the specified objects and appends them to the specified container. The objects can come from multiple source databases, and must match the type of owner specified, but must be from a different database than the ownerId object. See also wbblockCloneObjects() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
static Acad::ErrorStatus wbblockCloneObjects(
    AcDbObjectIdArray& objectIds,
    AcDbObjectId& ownerId,
    AcDbIdMapping& idMap,
    AcDb::DuplicateRecordCloning drc,
    bool deferXlation = false
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>objectIds</td>
<td>Input array of IDs of objects to clone. This array can contain only AcMapDMItem-derived objects (AcMapDmElement and AcMapDMGroup instances, for example).</td>
</tr>
<tr>
<td>ownerId</td>
<td>Input ID of the container object that will own the clones. This ID can refer to only an AcMapDMGroup-derived object in the destination database.</td>
</tr>
<tr>
<td>idMap</td>
<td>Input array of AcDbIdPair objects to use to translate object ID relationships.</td>
</tr>
<tr>
<td>drc</td>
<td>Input action for duplicate records.</td>
</tr>
<tr>
<td>deferXlation</td>
<td>Input true to defer ID translation. The default value is false.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Invoked when the current scale is modified.

```cpp
virtual void CurrentScaleModified(
    const AcMapDMMap* pMap,
    double dNewScale,
    double dOldScale
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pMap</td>
<td>Input pointer to the AcMapDMMap object.</td>
</tr>
<tr>
<td>dNewScale</td>
<td>Input value of new scale.</td>
</tr>
<tr>
<td>dOldScale</td>
<td>Input value of the old scale.</td>
</tr>
</tbody>
</table>

Returns

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.toolsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Invoked just before stylization is dismissed.

```
virtual void DismissStylizationBegin(
    const AcMapDMMap* pMap
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pMap</td>
<td>Input pointer to the AcMapDMMap object.</td>
</tr>
</tbody>
</table>

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Invoked when a stylization dismissal is cancelled.

```cpp
virtual void DismissStylizationCancel(
    const AcMapDMMap* pMap
);
```

**Parameters**

- `pMap`: Input pointer to the AcMapDMMap object.

**Returns**

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Invoked just after stylization is dismissed.

```cpp
virtual void DismissStylizationEnd(
    const AcMapDMMap* pMap
);
```

### Parameters

- **pMap**
  - Description: Input pointer to the AcMapDMMap object.

### Returns

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Invoked when a display-management item is added to the map.

```cpp
virtual void ItemAppended(const AcMapDMItem * pItem);
```

Parameters

<table>
<thead>
<tr>
<th>pItem</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pItem</td>
<td>Input pointer to the added AcMapDMItem object.</td>
</tr>
</tbody>
</table>

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Invoked when a display-management item is erased from the map.

```cpp
virtual void ItemErased(
    const AcMapDMItem * pItem,
    bool bErased
);
```

**Parameters**
- `pItem` Input pointer to the erased `AcMapDMItem` object.
- `bErased` Input true to erase the item.

**Returns**

Returns nothing.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapDMMapReactor Class, AcMapDMMapReactor Class
AcMapDMMapReactor:: ItemModified Method
AcMapDMMapReactor Class | AcMapDMMapReactor Class

Invoked when a display-management item is modified.

```cpp
virtual void ItemModified(
    const AcMapDMItem * pItem
);
```

Parameters

<table>
<thead>
<tr>
<th>pItem</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input pointer to the modified AcMapDMItem object.</td>
<td></td>
</tr>
</tbody>
</table>

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMMapReactor Class, AcMapDMMapReactor Class
AcMapDMMapReactor:: ScaleAdded Method
AcMapDMMapReactor Class | AcMapDMMapReactor Class

Invoked when a scale is added.

```cpp
virtual void ScaleAdded(
    double dScale,
    bool bIsCopy
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dScale</td>
<td>Input scale value.</td>
</tr>
<tr>
<td>bIsCopy</td>
<td>Input true if the new scale is added by copying styles.</td>
</tr>
</tbody>
</table>

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Invoked when a scale is erased.

```cpp
virtual void ScaleErased(
    double dScale
);
```

**Parameters**

- **dScale**: Input scale value.

**Returns**

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Invoked when a scale is modified.

```cpp
virtual void ScaleModified(
    double dNewScale,
    double dOldScale
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Description</th>
<th>Input value of the new scale.</th>
</tr>
</thead>
<tbody>
<tr>
<td>dNewScale</td>
<td></td>
</tr>
<tr>
<td>dOldScale</td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

Returns nothing.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at **support@toolsfactory.com**.
Invoked when a style is added.

```cpp
virtual void StyleAppended(
    const AcMapDMStyle* pStyle
);
```

**Parameters**

- **pStyle**
  - Description: Input pointer to the added AcMapDMStyle object.

**Returns**

Returns nothing.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Invoked when a style is erased.

```cpp
virtual void StyleErased(
    const AcMapDMStyle* pStyle,
    bool bErased
);```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pStyle</td>
<td>Input pointer to the erased AcMapDMStyle object.</td>
</tr>
<tr>
<td>bErased</td>
<td>Input true to erase the style.</td>
</tr>
</tbody>
</table>

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Invoked when a style is modified.

```cpp
virtual void StyleModified(
    const AcMapDMStyle* pStyle
);
```

Parameters

- **pStyle**: Input pointer to the modified AcMapDMStyle object.

Returns

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.docOMATIC.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Invoked when a style reference is added.

```cpp
virtual void StyleReferenceAppended(
    const AcMapDMElement* pElement,
    const AcMapDMStyleReference* pStyleRef,
    double dScale
);
```

**Parameters**

- **pElement**
  - Input pointer to the AcMapDMElement object that the style is added to.

- **pStyleRef**
  - Input pointer to the added `AcMapDMStyleReference` object.

- **dScale**
  - Input scale at which the style reference is applied.

**Returns**

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Invoked when a style reference is erased.

```cpp
virtual void StyleReferenceErased(
    const AcMapDMStyleReference* pStyleRef,
    bool bErased
);
```

Parameters

- **pStyleRef**: Input pointer to the erased `AcMapDMStyleReference` object.
- **bErased**: Input true to erase the style reference.

Returns

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Invoked when a style reference is modified.

```cpp
virtual void StyleReferenceModified(
    const AcMapDMStyleReference* pStyleRef);
```

Parameters | Description
--------- |---------------------
pStyleRef | Input pointer to the modified `AcMapDMStyleReference` object.

Returns

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Invoked just before stylization occurs.

```cpp
virtual void UpdateStylizationBegin(
    const AcMapDMMap* pMap
);
```

Parameters Description
pMap Input pointer to the AcMapDMMap object.

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Invoked when a stylization is cancelled.

```cpp
virtual void UpdateStylizationCancel(
    const AcMapDMMap* pMap
);
```

**Parameters**
- `pMap`: Input pointer to the AcMapDMMap object.

**Returns**
Returns nothing.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Invoked just after stylization occurs.

```cpp
virtual void UpdateStylizationEnd(
    const AcMapDMMap* pMap
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pMap</td>
<td>Input pointer to the AcMapDMMap object.</td>
</tr>
</tbody>
</table>

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Invoked when a category is added to the style library.

```cpp
virtual void CategoryAppended(
    const AcMapDMStyleLibrary* pStyleLibrary,
    const AcMapDMStyleCategory* pCategory
);
```

Parameters

- **pStyleLibrary**: Input pointer to the AcMapDMStyleLibrary object.
- **pCategory**: Input pointer to the AcMapDMStyleCategory object.

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Invoked when a category is modified.

```cpp
virtual void CategoryModified(
    const AcMapDMStyleCategory * pCategory);
```

**Parameters**
- `pCategory`: Input pointer to the `AcMapDMStyleCategory` object.

**Returns**
Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Invoked when a category is detached from the style library.

```cpp
virtual void CategoryUnAppended(
    const AcMapDMStyleLibrary* pStyleLibrary,
    const AcMapDMStyleCategory * pCategory,
    bool bErased
);
```

Parameters

- **pStyleLibrary**: Input pointer to the `AcMapDMStyleLibrary` object.
- **pCategory**: Input pointer to the `AcMapDMStyleCategory` object.
- **bErased**: Input true to erase the category as well.

Returns

Returns nothing.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at **support@toolsfactory.com**.
Invoked when a new map is appended or after a reactor is attached to a project that has map(s) defined.

```cpp
virtual void MapAppended(
    AcMapDMMap* pMap
);
```

Parameters

- `pMap` Input pointer to a AcMapDMMapobject.

Returns

Returns nothing.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMProjectReactor Class, AcMapDMProjectReactor Class
AcMapDMProjectReactor:: MapGoodBye Method
AcMapDMProjectReactor Class | AcMapDMProjectReactor Class

Invoked when a map is unloaded.

```c
virtual void MapGoodBye(
    AcMapDMMap* pMap
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pMap</td>
<td>Input pointer to the unloaded AcMapDMMap object.</td>
</tr>
</tbody>
</table>

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Invoked just before a map is set as the current map.

```cpp
virtual void MapSetCurrentBegin(
    AcMapDMMap* pOldCurrentMap
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pOldCurrentMap</td>
<td>Input pointer to the previous AcMapDMMap object.</td>
</tr>
</tbody>
</table>

Returns

Returns nothing.

Created with a commercial version of [Doc-O-Matic](#). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Invoked just after a map is set as the current map.

```cpp
virtual void MapSetCurrentEnd(
    AcMapDMMap* pNewCurrentMap
);
```

Parameters

- **pNewCurrentMap**: Input pointer to the new current `AcMapDMMap` object.

Returns

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Invoked when setting a map as current fails.

**virtual void** MapSetCurrentFails(
    AcMapDMMap* pOldCurrentMap
);

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pOldCurrentMap</td>
<td>Input pointer to the previous AcMapDMMap object.</td>
</tr>
</tbody>
</table>

Returns nothing.

Created with a commercial version of [Doc-O-Matic](#). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Invoked when a map is detached.

```c++
virtual void MapUnAppended(
    AcMapDMMap* pMap,
    bool bErased
);
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pMap</td>
<td>Input pointer to the AcMapDMMap object.</td>
</tr>
<tr>
<td>bErased</td>
<td>Input true to erase the map as well.</td>
</tr>
</tbody>
</table>

Returns

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://toolsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Invoked when a style is added to the category.

```cpp
virtual void StyleAppended(
    const AcMapDMStyleCategory * pCategory,
    const AcMapDMStyle * pStyle
);
```

Parameters | Description
--- | ---
pCategory | Input pointer to the `AcMapDMStyleCategory` object.
pStyle | Input pointer to the `AcMapDMStyle` object.

Returns

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links

AcMapDMProjectReactor Class, AcMapDMProjectReactor Class
AcMapDMProjectReactor:: StyleModified Method
AcMapDMProjectReactor Class | AcMapDMProjectReactor Class

Invoked when a style is modified.

```cpp
virtual void StyleModified(
    const AcMapDMStyle* pStyle
);
```

Parameters

- **pStyle**: Input pointer to the AcMapDMStyle object.

Returns

- Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Invoked when a style is detached.

```cpp
virtual void StyleUnAppended(
    const AcMapDMStyle* pStyle,
    bool bErased
);
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pStyle</td>
<td>Input pointer to the AcMapDMStyle object.</td>
</tr>
<tr>
<td>bErased</td>
<td>Input true to erase the style as well.</td>
</tr>
</tbody>
</table>

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

\textit{virtual} \texttt{~AcMapDMQueryDataSourceDescriptor}();

Returns

Returns nothing.

Created with a commercial version of \textit{Doc-O-Matic}. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at \texttt{support@toolsfactory.com}. 
Constructs an instance of this class.

```
AcMapDMQueryDataSourceDescriptor();
```

Returns

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Retrieves the query definition.

```cpp
virtual Acad::ErrorStatus GetQuery(
    struct resbuf*& pRb
) const;
```

**Parameters**

- `pRb`: Output resbuf object that contains the query.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the query definition.

```cpp
virtual Acad::ErrorStatus SetQuery(
    const struct resbuf* pRb
);
```

Parameters

- **pRb**: Input resbuf object that contains the query.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Destroys an instance of this class.

```
virtual ~AcMapDMRasterDataSourceDescriptor();
```

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Constructs an instance of this class.

AcMapDMRasterDataSourceDescriptor();
Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

```
virtual ~AcMapDMRasterElement();
```

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Constructs an instance of this class.

AcMapDMRasterElement();
Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Runs the query against the current drawing.

```cpp
virtual Acad::ErrorStatus AcquireEntities();
```

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Clones objects from the source drawings.

```cpp
virtual bool ClonesObjectsFromExternalSource() const;
```

Returns false (because objects from only the current drawing are selected).

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMRasterElement Class, AcMapDMRasterElement Class
AcMapDMRasterElement:: dwgInFields Method
AcMapDMRasterElement Class | AcMapDMRasterElement Class

Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dwgInFields(
    AcDbDwgFiler* pFiler
);
```

Parameters

<table>
<thead>
<tr>
<th>Description</th>
<th>pFiler</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input filer to use to read the object's data.</td>
<td></td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Virtual Acad::ErrorStatus dwgOutFields(const AcDbDwgFiler* pFiler);

Parameters  Description
pFiler  Input filer to use to write the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMRasterElement Class, AcMapDMRasterElement Class
AcMapDMRasterElement:: dxfInFields Method
AcMapDMRasterElement Class | AcMapDMRasterElement Class

Lets this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dxfInFields(
    AcDbDxfFiler* pFiler
);
```

Parameters

- **pFiler**: Input filer to use to read the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links

AcMapDMRasterElement Class, AcMapDMRasterElement Class
AcMapDMRasterElement:: dxfOutFields Method
AcMapDMRasterElement Class | AcMapDMRasterElement Class

Lets this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dxfOutFields(
    AcDbDxfFiler* pFiler
) const;
```

Parameters

<table>
<thead>
<tr>
<th>Description</th>
<th>pFiler</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input filer to use to write in the object's data.</td>
<td></td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieve the query definition's data-source descriptor.

```cpp
virtual Acad::ErrorStatus GetAcquisitionCriteria(
    AcMapDMDataSourceDescriptor* pDataSourceDsc
) const;
```

Parameters  Description
pDataSourceDsc  Output pointer to the data-source descriptor.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the query definition's data-source descriptor.

```cpp
virtual Acad::ErrorStatus SetAcquisitionCriteria(
    const AcMapDMDataSourceDescriptor* pDataSourceDsc
);```

**Parameters**
- `pDataSourceDsc`: Input pointer to a data-source descriptor.

**Returns**
- Returns Acad::eOk if successful; otherwise, returns a different error code.

**Remarks**

This function determines which entities become part of this element.

Created with a commercial version of [Doc-O-Matic](https://docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

```
virtual ~AcMapDMRasterStyle();
```

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Constructs an instance of this class.

AcMapDMRasterStyle();

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Applies a style to a raster.

```cpp
virtual Acad::ErrorStatus Apply(
    void* & pCookie,
    AcDbEntity* pEnt,
    Adesk::UInt32 flag = 0
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pCookie</td>
<td>Input memory value used by styles to transfer cached data across time from one stylization method to another. Cookie memory is considered to be opaque for existing styles, and should not be modified by the caller.</td>
</tr>
<tr>
<td>pEnt</td>
<td>Input entity to apply the style to.</td>
</tr>
<tr>
<td>flag</td>
<td>Reserved. For internal use only.</td>
</tr>
</tbody>
</table>

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Clears image brightness. See also IsBrightnessCleared().

```
Acad::ErrorStatus ClearBrightness();
```

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](http://www.doc-o-matic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Clears image contrast. See also IsContrastCleared().

```cpp
Acad::ErrorStatus ClearContrast();
```

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Clears image fade. See also IsFadeCleared().

```
Acad::ErrorStatus ClearFade();
```

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.doc-o-matic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Clears image transparency. See also IsTransparencyCleared().

```cpp
Acad::ErrorStatus ClearTransparency();
```

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determines whether image transparency is cleared. See also ClearTransparency().

```cpp
bool IsTransparencyCleared() const;
```

Returns

Returns true if image transparency is cleared.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Clones the object. See also clone() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```
virtual AcRxObject* clone() const;
```

Returns a pointer to the clone, or NULL if cloning was unsuccessful.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMRasterStyle Class, AcMapDMRasterStyle Class
AcMapDMRasterStyle::: copyFrom Method
AcMapDMRasterStyle Class | AcMapDMRasterStyle Class

Copies the contents of an object into the messaged object, if feasible. See also copyFrom() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus copyFrom(
    const AcRxObject* other
);
```

Parameters
other Input pointer to the object to copy from.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Deletes the style's cookie during database destruction.

```cpp
virtual void DeleteCookie(
    void*& pCookie
);
```

**Parameters**

- **pCookie**  
  Input cookie to delete.

**Returns**

Returns nothing.

**Remarks**

Because the database is in the midst of destruction, do not attempt any additional work by using the cookie at this point, other than its deletion.

Created with a commercial version of [Doc-O-Matic](https://example.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Clears the stylization of a raster.

```cpp
virtual Acad::ErrorStatus Dismiss(
    void*& pCookie,
    const AcDbObjectId id,
    Adesk::UInt32 flag = 0
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pCookie</td>
<td>Input memory value used by styles to transfer cached data across time from one stylization method to another. Cookie memory is considered to be opaque for existing styles, and should not be modified by the caller.</td>
</tr>
<tr>
<td>id</td>
<td>Input entity ID.</td>
</tr>
<tr>
<td>flag</td>
<td>Reserved. For internal use only.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMRasterStyle Class, AcMapDMRasterStyle Class
AcMapDMRasterStyle:: dwgInFields Method
AcMapDMRasterStyle Class | AcMapDMRasterStyle Class

Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```
virtual Acad::ErrorStatus dwgInFields(
    AcDbDwgFiler* pFiler
);
```

Parameters Description
pFiler Input filer to use to read the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMRasterStyle Class, AcMapDMRasterStyle Class
AcMapDMRasterStyle:: dwgOutFields Method
AcMapDMRasterStyle Class | AcMapDMRasterStyle Class

Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dwgOutFields(
    AcDbDwgFiler* pFiler
) const;
```

Parameters

- **pFiler**: Input filer to use to write the object's data.

Returns

- Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com/). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links

AcMapDMRasterStyle Class, AcMapDMRasterStyle Class
AcMapDMRasterStyle:: dxfInFields Method
AcMapDMRasterStyle Class | AcMapDMRasterStyle Class

Lets this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dxfInFields(
    AcDbDxfFiler* pFiler
);
```

Parameters Description
pFiler Input filer to use to read the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Let this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dxfOutFields(
    AcDbDxfFiler* pFiler
) const;
```

**Parameters**
- **pFiler**: Input filer to use to write in the object's data.

**Returns**
- Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Enables or disables the style of a raster.

```cpp
virtual Acad::ErrorStatus Enable(
    void* & pCookie,
    const AcDbObjectId id,
    bool bEnable,
    Adesk::UInt32 flag = 0
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pCookie</td>
<td>Input memory value used by styles to transfer cached data across time from one stylization method to another. Cookie memory is considered to be opaque for existing styles, and should not be modified by the caller.</td>
</tr>
<tr>
<td>id</td>
<td>Input entity ID.</td>
</tr>
<tr>
<td>bEnable</td>
<td>Input true to enable the style.</td>
</tr>
<tr>
<td>flag</td>
<td>Reserved. For internal use only.</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves image brightness. See also SetBrightness().

```cpp
Acad::ErrorStatus GetBrightness(
    Adesk::Int8& brightness
) const;
```

**Parameters**
- **brightness**
  - Output brightness value.

**Returns**
- Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Retrieves image contrast. See also SetContrast().

```cpp
Acad::ErrorStatus GetContrast(
    Adesk::Int8& contrast
) const;
```

Parameters

- `contrast`: Output contrast value.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves image fade. See also SetFade().

```
Acad::ErrorStatus GetFade(
    Adesk::Int8& fade
) const;
```

Parameters

- **fade**: Output fade value.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves image transparency. See also SetTransparency().

```cpp
Acad::ErrorStatus GetTransparency(
    bool& bTransp
) const;
```

Parameters

- `bTransp`: Output transparency value.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determine whether image brightness is cleared. See also ClearBrightness().

```cpp
bool IsBrightnessCleared() const;
```

Returns true if image brightness is cleared.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMRasterStyle Class, AcMapDMRasterStyle Class
AcMapDMRasterStyle:: IsContrastCleared Method
AcMapDMRasterStyle Class | AcMapDMRasterStyle Class

Determines whether image contrast is cleared. See also ClearContrast().

```cpp
bool IsContrastCleared() const;
```

Returns true if image contrast is cleared.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determines whether image fade is cleared. See also ClearFade().

```cpp
bool IsFadeCleared() const;
```

Returns true if image fade is cleared.

Created with a commercial version of [Doc-O-Matic](https://www.docsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets image brightness. See also GetBrightness().

```cpp
Acad::ErrorStatus SetBrightness(
    Adesk::Int8 brightness
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Description</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>brightness</td>
<td>Input brightness value.</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets image contrast. See also GetContrast().

```cpp
Acad::ErrorStatus SetContrast(
    Adesk::Int8 contrast
);
```

Parameters Description
contrast Input contrast value.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets image fade. See also GetFade().

`Acad::ErrorStatus SetFade(
    Adesk::Int8 fade
);`

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fade</td>
<td>Input fade value.</td>
</tr>
</tbody>
</table>

Returns `Acad::eOk` if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets image transparency. See also GetTransparency().

\texttt{Acad::ErrorStatus SetTransparency(}
\texttt{    bool bTransp}
\texttt{);}  

Parameters | Description  
--- | ---  
bTransp | Input transparency value. 

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of \texttt{Doc-O-Matic}. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at \texttt{support@toolsfactory.com}. 

Links
AcMapDMRasterStyle Class, AcMapDMRasterStyle Class
AcMapDMRasterStyle:: SetTransparency Method
AcMapDMRasterStyle Class | AcMapDMRasterStyle Class
Removes the style from a raster.

```cpp
virtual Acad::ErrorStatus UnApply(
    void*& pCookie,
    AcDbEntity* pEnt,
    Adesk::UInt32 flag = 0
);
```

**Parameters**

- **pCookie**
  - Input memory value used by styles to transfer cached data across time from one stylization method to another. Cookie memory is considered to be opaque for existing styles, and should not be modified by the caller.

- **pEnt**
  - Input entity.

- **flag**
  - Reserved. For internal use only.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

**Remarks**

The caller can retrieve this style's cookie, stored previously by another stylization function.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves and stylizes a raster.

```cpp
virtual Acad::ErrorStatus Update(
    void* & pCookie,
    const AcDbObjectId id,
    Adesk::Uint32 flag = 0
);
```

**Parameters**

- **pCookie**
  - Output memory value used by styles to transfer cached data across time from one stylization method to another. Cookie memory is considered to be opaque for existing styles, and should not be modified by the caller.

- **id**
  - Input entity ID.

- **flag**
  - Reserved. For internal use only.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

```cpp
virtual ~AcMapDMSEAnnotationStyle();
```

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Constructs an instance of this class.

```cpp
AcMapDMSEAnnotationStyle(
    Adesk::Boolean bCreateAsSymbolStyle = Adesk::kFalse)
```

**Parameters**

<table>
<thead>
<tr>
<th>bCreateAsSymbolStyle</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adesk::Boolean</td>
<td>If kTrue, then this annotation style is marked as a symbol style.</td>
</tr>
</tbody>
</table>

**Returns**

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Clones the object. See also clone() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual AcRxObject* clone() const;
```

Returns a pointer to the clone, or NULL if cloning was unsuccessful.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links

AcMapDMSEAnnotationStyle Class, AcMapDMSEAnnotationStyle Class
AcMapDMSEAnnotationStyle:: ColorIsExpression Method
AcMapDMSEAnnotationStyle Class | AcMapDMSEAnnotationStyle Class

Determines whether the color is an expression.

```cpp
bool ColorIsExpression() const;
```

Returns

Returns true if the color is an expression.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Copies the contents of an object into the messaged object, if feasible. See also copyFrom() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus copyFrom(const AcRxObject* other);
```

**Parameters**

- `other` Input pointer to the object to copy from.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Deletes the style's cookie during database destruction.

```cpp
virtual void DeleteCookie(
    void*& pCookie
);
```

**Parameters**

- `pCookie`  
  Input opaque memory value.

**Returns**

Returns nothing.

**Remarks**

Because the database is in the midst of destruction, do not attempt any additional work by using the cookie at this point, other than its deletion.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Removes stylization information that is used during regeneration.

```c++
virtual Acad::ErrorStatus Dismiss(
    void*& pCookie,
    const Acad::ObjectId id,
    Adesk::UInt32 flag = 0
);
```

**Parameters**

- **pCookie**
  - Description: Output cookie stored by Update().
- **id**
  - Description: Input object ID.
- **flag**
  - Description: Reserved. For internal use only.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
virtual Acad::ErrorStatus dwgInFields(
    AcDbDwgFiler* pFiler
);

Parameters

pFiler Input filer to use to read the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMSEAnnotationStyle Class, AcMapDMSEAnnotationStyle Class
AcMapDMSEAnnotationStyle:: dwgOutFields Method
AcMapDMSEAnnotationStyle Class | AcMapDMSEAnnotationStyle Class

Lets this object write its data. See also dwgOutFields() in the AutoCAD
ObjectARX Developer's Guide. This overloaded function is called by the system
as needed; it is unlikely that you will need to call it directly.

**virtual Acad::ErrorStatus dwgOutFields(
    AcDbDwgFiler* pFiler
) const;**

Parameters
pFiler Input filer to use to write the object's data.

Returns
Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this
message disappear you need to register this software. If you have problems
registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapDMSEAnnotationStyle Class, AcMapDMSEAnnotationStyle Class
AcMapDMSEAnnotationStyle:: dxfInFields Method
AcMapDMSEAnnotationStyle Class | AcMapDMSEAnnotationStyle Class

Lets this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dxfInFields(
    AcDbDxfFiler* pFiler
);
```

Parameters Description
pFiler Input filer to use to read the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Let this object write its data. See also dxfOutFields() in the AutoCAD
ObjectARX Developer's Guide. This overloaded function is called by the system
as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dxfOutFields(
    AcDbDxfFiler* pFiler
) const;
```

**Parameters**

- *pFiler*  
  Input filer to use to write in the object's data.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this
message disappear you need to register this software. If you have problems
registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Enables or disables the style for the specified entity.

```cpp
virtual Acad::ErrorStatus Enable(
    void*& pCookie,
    const AcDbObjectId id,
    bool bEnable,
    Adesk::UInt32 flag = 0
);
```

**Parameters**
- **pCookie**: Input opaque memory value.
- **id**: Input object ID.
- **bEnable**: Input true to enable the style.
- **flag**: Reserved. For internal use only.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.doc-o-matic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Retrieves the annotation template ID.

```cpp
Acad::ErrorStatus GetAnnotationTemplate(
    AcDbObjectId& templateID
) const;
```

**Parameters**
- `templateID` Output annotation template ID.

**Returns**
- Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Retrieves the color.

```cpp
Acad::ErrorStatus GetColor(
    AcCmColor& color
) const;
```

Parameters | Description
--- | ---
color | Output color.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](#). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](#).
Retrieves the layer id.

```cpp
Acad::ErrorStatus GetLayer(
    AcDbObjectId& layerId
) const;
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>layerId</td>
<td>Output layer AcDbObjectId.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the linetype id.

```cpp
Acad::ErrorStatus GetLinetype(
    AcDbObjectId& linetypeId
) const;
```

**Parameters**
- `linetypeId`:
  - **Description**: Output linetype `AcDbObjectId`.  

**Returns**
- Returns `Acad::eOk` if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Retrieves the lineweight.

```cpp
Acad::ErrorStatus GetLineWeight(
   AcDb::LineWeight& lw
) const;
```

**Parameters**
- `lw`

**Description**
- Output lineweight.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.toolsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Retrieves the rotation.

```cpp
Acad::ErrorStatus GetRotation(
    double& dRotation
) const;
```

Parameters

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dRotation</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the scale.

```cpp
Acad::ErrorStatus GetScale(
    double& dScale
) const;
```

**Parameters**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dScale</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the AcDbObjectId of any stylization entities created during an Update().

```cpp
virtual Acad::ErrorStatus GetStylizationEntities(
    AcDbObjectIdArray& output,
    const AcDbObjectId id,
    void* pCookie,
    Adesk::UInt32 flag = 0
)
```

**Parameters**

- **output**: Output array of stylization entities.
- **id**: Target entity against which the style was applied.
- **pCookie**: Input opaque memory value.
- **flag**: Reserved. For internal use only.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Indicates that this style is a symbol style instead of an annotation style.

Adesk::Boolean IsSymbolStyle() const;
Returns

Returns kTrue if this is a symbol style; otherwise, returns kFalse.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determines whether the layer is an expression.

```cpp
bool LayerIsExpression() const;
```

Returns true if the layer is an expression.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determined whether the linetype is an expression.

```cpp
bool LinetypeIsExpression() const;
```

Returns true if the linetype is an expression.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links

AcMapDMSEAnnotationStyle Class, AcMapDMSEAnnotationStyle Class
AcMapDMSEAnnotationStyle:: LineWeightIsExpression Method
AcMapDMSEAnnotationStyle Class | AcMapDMSEAnnotationStyle Class

Determines whether the lineweight is an expression.

```cpp
bool LineWeightIsExpression() const;
```

Returns true if the lineweight is an expression.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links

AcMapDMSEAnnotationStyle Class, AcMapDMSEAnnotationStyle Class

AcMapDMSEAnnotationStyle:: RotationIsExpression Method

AcMapDMSEAnnotationStyle Class | AcMapDMSEAnnotationStyle Class

Determines whether the rotation is an expression.

```cpp
bool RotationIsExpression() const;
```

Returns

Returns true if the rotation is an expression.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMSEAnnotationStyle Class, AcMapDMSEAnnotationStyle Class
AcMapDMSEAnnotationStyle:: ScaleIsExpression Method
AcMapDMSEAnnotationStyle Class | AcMapDMSEAnnotationStyle Class

Determines whether the scale is an expression.

```cpp
bool ScaleIsExpression() const;
```

Returns true if the scale is an expression.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the annotation template ID or block ID.

Acad::ErrorStatus SetAnnotationTemplate(
   AcDbObjectId annotationTemplateID
);

Parameters

<table>
<thead>
<tr>
<th></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>annotationTemplateID</td>
<td>Input annotation template ID or block ID.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the color.

`Acad::ErrorStatus SetColor(
    const AcCmColor& color
);`

Parameters | Description
--- | ---
color | Input color.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://toolsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the layer id.

```cpp
Acad::ErrorStatus SetLayer(
    const AcDbObjectId layerId
);
```

Parameters

- **layerId**
  
  Input layer AcDbObjectId.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.
Sets the linetype id.

```cpp
Acad::ErrorStatus SetLinetype(
    const AcDbObjectId linetypeId
);
```

Parameters Description
linetypeId Input linetype AcDbObjectId.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the lineweight.

```cpp
Acad::ErrorStatus SetLineWeight(
    AcDb::LineWeight lw
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>lw</td>
<td>Input lineweight.</td>
</tr>
</tbody>
</table>

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](#). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the rotation.

```
Acad::ErrorStatus SetRotation(
    double dRotation
);
```

Parameters | Description
---|---
dRotation | Input rotation.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the scale.

```cpp
Acad::ErrorStatus SetScale(
    double dScale
);```

**Parameters**

- `dScale`: Input scale.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](http://www.toolsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Builds stylization information for later regeneration.

```cpp
virtual Acad::ErrorStatus Update(
    void*& pCookie,
    const AcDbObjectId id,
    Adesk::UInt32 flag = 0
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pCookie</td>
<td>Input opaque memory value for later access by Dismiss().</td>
</tr>
<tr>
<td>id</td>
<td>Input object ID.</td>
</tr>
<tr>
<td>flag</td>
<td>Reserved. For internal use only.</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

**Remarks**

Derived classes can perform any one-time Update()-time actions here. At Dismiss() time, the implementation can do any required work to clean up storage created at Update() time.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

```cpp
virtual ~AcMapDMSEHatchStyle();
```

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Constructs an instance of this class.

AcMapDMSEHatchStyle();

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMSEHatchStyle Class, AcMapDMSEHatchStyle Class
AcMapDMSEHatchStyle:: clone Method
AcMapDMSEHatchStyle Class | AcMapDMSEHatchStyle Class

Clones the object. See also clone() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

    virtual AcRxObject* clone() const;

Returns

Returns a pointer to the clone, or NULL if cloning was unsuccessful.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determines whether the color is an expression.

```cpp
bool ColorIsExpression() const;
```

Returns true if the color is an expression.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Copies the contents of an object into the messaged object, if feasible. See also copyFrom() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus copyFrom(
   const AcRxObject* other
);
```

**Parameters**

- `other`: Input pointer to the object to copy from.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.
Deletes the style's cookie during database destruction.

```cpp
virtual void DeleteCookie(
    void*& pCookie
);
```

Parameters | Description
---|---
pCookie | Input opaque memory value.

Returns

Returns nothing.

Remarks

Because the database is in the midst of destruction, do not attempt any additional work by using the cookie at this point, other than its deletion.

Created with a commercial version of [Doc-O-Matic](http://toolsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMSEHatchStyle Class, AcMapDMSEHatchStyle Class
AcMapDMSEHatchStyle:: Dismiss Method
AcMapDMSEHatchStyle Class | AcMapDMSEHatchStyle Class

Removes stylization information that is used during regeneration.

```cpp
virtual Acad::ErrorStatus Dismiss(
    void* & pCookie,
    const AcDbObjectId id,
    Adesk::UInt32 flag = 0
);
```

Parameters

| pCookie | Output cookie stored by Update(). |
| id      | Input object ID.                  |
| flag    | Reserved. For internal use only. |

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMSEHatchStyle Class, AcMapDMSEHatchStyle Class
AcMapDMSEHatchStyle:: dwgInFields Method
AcMapDMSEHatchStyle Class | AcMapDMSEHatchStyle Class

Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dwgInFields(
    AcDbDwgFiler* pFiler
);
```

Parameters

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pFiler</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dwgOutFields(
    AcDbDwgFiler* pFiler
) const;
```

**Parameters**
- pFiler: Input filer to use to write the object's data.

**Returns**
- Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapDMSEHatchStyle Class, AcMapDMSEHatchStyle Class
AcMapDMSEHatchStyle:: dxfInFields Method
AcMapDMSEHatchStyle Class | AcMapDMSEHatchStyle Class

Lets this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dxfInFields(
    AcDbDxfFiler* pFiler
);
```

**Parameters**

- **pFiler**
  - Description: Input filer to use to read the object's data.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMSEHatchStyle Class, AcMapDMSEHatchStyle Class
AcMapDMSEHatchStyle:: dxfOutFields Method
AcMapDMSEHatchStyle Class | AcMapDMSEHatchStyle Class

Lets this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dxfOutFields(
    AcDbDxfFiler* pFiler
) const;
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pFiler</td>
<td>Input filer to use to write in the object's data.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Enables or disables the style for the specified entity.

```cpp
virtual Acad::ErrorStatus Enable(
    void* & pCookie,
    const AcDbObjectId id,
    bool bEnable,
    Adesk::UInt32 flag = 0
);
```

Parameters

- `pCookie` Input opaque memory value.
- `id` Input object ID.
- `bEnable` Input true to enable the style.
- `flag` Reserved. For internal use only.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the color.

\texttt{Acad::ErrorStatus GetColor(}
\hspace{2em} \texttt{AcCmColor& color}
\hspace{2em} \texttt{)} \texttt{const;}

\textbf{Parameters} \hspace{2em} \textbf{Description}
\begin{itemize}
  \item \texttt{color} \hspace{2em} \text{Output color.}
\end{itemize}

\textbf{Returns}

Returns \texttt{Acad::eOk} if successful; otherwise, returns a different error code.

Created with a commercial version of \texttt{Doc-O-Matic}. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at \texttt{support@toolsfactory.com}.
Retrieves the layer id.

```cpp
Acad::ErrorStatus GetLayer(
    AcDbObjectId& layerId
) const;
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>layerId</td>
<td>Output layer AcDbObjectId.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the rotation.

```cpp
Acad::ErrorStatus GetRotation(
    double& dRotation
) const;
```

Parameters  

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dRotation</td>
<td>Output rotation.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Retrieves the scale.

```cpp
Acad::ErrorStatus GetScale(
    double& dScale
) const;
```

Parameters  Description
---  -----------------
dScale  Output scale.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the AcDbObjectId of any stylization entities created during an Update().

```cpp
virtual Acad::ErrorStatus GetStylizationEntities(
    AcDbObjectIdArray& output,
    const AcDbObjectId id,
    void* pCookie,
    Adesk::UInt32 flag = 0
);
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>output</td>
<td>Output array of stylization entities.</td>
</tr>
<tr>
<td>id</td>
<td>Target entity against which the style was applied.</td>
</tr>
<tr>
<td>pCookie</td>
<td>Input opaque memory value.</td>
</tr>
<tr>
<td>flag</td>
<td>Reserved. For internal use only.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determines whether the layer is an expression.

```cpp
bool LayerIsExpression() const;
```

Returns true if the layer is an expression.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMSEHatchStyle Class, AcMapDMSEHatchStyle Class
AcMapDMSEHatchStyle:: RotationIsExpression Method
AcMapDMSEHatchStyle Class | AcMapDMSEHatchStyle Class

Determines whether the rotation is an expression.

```cpp
bool RotationIsExpression() const;
```

Returns

Returns true if the rotation is an expression.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determines whether the scale is an expression.

```cpp
bool ScaleIsExpression() const;
```

Returns true if the scale is an expression.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the color.

```cpp
Acad::ErrorStatus SetColor(
    const AcCmColor& color
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>color</td>
<td>Input color value.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the layer id.

```
Acad::ErrorStatus SetLayer(
    const AcDbObjectId layerId
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>layerId</td>
<td>Input layer AcDbObjectId.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the rotation.

```c
Acad::ErrorStatus SetRotation(
    double dRotation
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Description</th>
<th>Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input rotation.</td>
<td>dRotation</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the scale.

\begin{verbatim}
Acad::ErrorStatus SetScale(
    double dScale
);
\end{verbatim}

**Parameters**
- dScale: Input scale.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapDMSEHatchStyle Class, AcMapDMSEHatchStyle Class
AcMapDMSEHatchStyle:: Update Method
AcMapDMSEHatchStyle Class | AcMapDMSEHatchStyle Class

Builds stylization information for later regeneration.

```cpp
virtual Acad::ErrorStatus Update(
    void* & pCookie,
    const AcDbObjectId id,
    Adesk::UInt32 flag = 0
);
```

Parameters

- **pCookie**
  - Input opaque memory value for later access by Dismiss().
- **id**
  - Input object ID.
- **flag**
  - Reserved. For internal use only.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Remarks

Derived classes can perform any one-time Update()-time actions here. At Dismiss() time, the implementation can do any required work to clean up storage created at Update() time.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMSETextStyle Class
AcMapDMSETextStyle:: Justification Enumeration
AcMapDMSETextStyle Class

Enumerates the types of text justification.

```c
enum Justification {
    kLeft,
    kCenter,
    kMiddle,
    kRight,
    kTopLeft,
    kTopCenter,
    kTopRight,
    kMiddleLeft,
    kMiddleCenter,
    kMiddleRight,
    kBottomLeft,
    kBottomCenter,
    kBottomRight,
    kNone
};
```

File
DmSETextStyle.h

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>kLeft</td>
<td>Left.</td>
</tr>
<tr>
<td>kCenter</td>
<td>Center.</td>
</tr>
<tr>
<td>kMiddle</td>
<td>Middle.</td>
</tr>
<tr>
<td>kRight</td>
<td>Right.</td>
</tr>
<tr>
<td>kTopLeft</td>
<td>Top left.</td>
</tr>
<tr>
<td>kTopCenter</td>
<td>Top center.</td>
</tr>
<tr>
<td>kTopRight</td>
<td>Top right.</td>
</tr>
<tr>
<td>kMiddleLeft</td>
<td>Middle left.</td>
</tr>
<tr>
<td>kMiddleCenter</td>
<td>Middle center.</td>
</tr>
<tr>
<td>kMiddleRight</td>
<td>Middle right.</td>
</tr>
<tr>
<td>kBottomLeft</td>
<td>Bottom left.</td>
</tr>
<tr>
<td>kBottomCenter</td>
<td>Bottom center.</td>
</tr>
</tbody>
</table>
Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

```
virtual ~AcMapDMSETextStyle();
```

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Constructs an instance of this class.

AcMapDMSETextStyle();
Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMSETextStyle Class, AcMapDMSETextStyle Class
AcMapDMSETextStyle:: clone Method
AcMapDMSETextStyle Class | AcMapDMSETextStyle Class

Clones the object. See also clone() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

virtual AcRxObject* clone() const;

Returns

Returns a pointer to the clone, or NULL if cloning was unsuccessful.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determine whether the color is an expression.

```cpp
bool ColorIsExpression();
```

Returns

Returns true if the color is an expression.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Copies the contents of an object into the messaged object, if feasible. See also copyFrom() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus copyFrom(
    const AcRxObject* other
);
```

**Parameters**

- `other`: Input pointer to the object to copy from.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Deletes the style's cookie during database destruction.

```c++
virtual void DeleteCookie(
    void*& pCookie
);
```

**Parameters**

<table>
<thead>
<tr>
<th>pCookie</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input opaque memory value.</td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

Returns nothing.

**Remarks**

Because the database is in the midst of destruction, do not attempt any additional work by using the cookie at this point, other than its deletion.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Removes stylization information that is used during regeneration.

```cpp
virtual Acad::ErrorStatus Dismiss(
    void*& pCookie,
    const AcDbObjectId id,
    Adesk::UInt32 flag = 0
);
```

- **Parameters**
  - `pCookie` Output cookie stored by Update().
  - `id` Input object ID.
  - `flag` Reserved. For internal use only.

- **Returns**
  - Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links

AcMapDMSETextStyle Class, AcMapDMSETextStyle Class
AcMapDMSETextStyle:: dwgInFields Method
AcMapDMSETextStyle Class | AcMapDMSETextStyle Class

Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

virtual Acad::ErrorStatus dwgInFields(
    AcDbDwgFiler* pFiler
);

Parameters Description
pFiler Input filer to use to read the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
AcMapDMSETextStyle Class, AcMapDMSETextStyle Class

AcMapDMSETextStyle::dwgOutFields Method

AcMapDMSETextStyle Class | AcMapDMSETextStyle Class

Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dwgOutFields(
    AcDbDwgFiler* pFiler
) const;
```

Parameters

- **pFiler**
  - Input filer to use to write the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at **support@toolsfactory.com**.
Links
AcMapDMSETextStyle Class, AcMapDMSETextStyle Class
AcMapDMSETextStyle:: dxfInFields Method
AcMapDMSETextStyle Class | AcMapDMSETextStyle Class

Lets this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

`virtual Acad::ErrorStatus dxfInFields(
    AcDbDxfFiler* pFiler
);`

Parameters Description
pFiler Input filer to use to read the object's data.

Returns
Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMSETextStyle Class, AcMapDMSETextStyle Class
AcMapDMSETextStyle:: dxfOutFields Method
AcMapDMSETextStyle Class | AcMapDMSETextStyle Class

Lets this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dxfOutFields(
    AcDbDxfFiler* pFiler
) const;
```

**Parameters**

**Description**

- `pFiler` Input filer to use to write in the object's data.

**Returns**

- Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Enables or disables the style for the specified entity.

```cpp
virtual Acad::ErrorStatus Enable(
    void*& pCookie,
    const AcDbObjectId id,
    bool bEnable,
    Adesk::UInt32 flag = 0
);
```

Parameters

- **pCookie**: Input opaque memory value.
- **id**: Input object ID.
- **bEnable**: Input true to enable the style.
- **flag**: Reserved. For internal use only.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Retrieves the text color.

```cpp
Acad::ErrorStatus GetColor(
    AcCmColor& color
) const;
```

**Parameters**
- `color` Output text color.

**Returns**

Returns `Acad::eOk` if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the text height.

```
 Acad::ErrorStatus GetHeight(
    double& dHeight
) const;
```

Parameters | Description
---|---
dHeight | Output height.

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the text justification.

```cpp
Acad::ErrorStatus GetJustification(
    Justification& justification
) const;
```

**Parameters**

- `justification` — Output Justification value.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the text layer id.

```cpp
Acad::ErrorStatus GetLayer(
    AcDbObjectId& layerId
) const;
```

Parameters

<table>
<thead>
<tr>
<th>layerId</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Output layer AcDbObjectId.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the text rotation.

```cpp
Acad::ErrorStatus GetRotation(
    double& dRotation
) const;
```

Parameters | Description
---|---
dRotation | Output rotation.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Retrieves the text style id.

Acad::ErrorStatus GetStyle(
    AcDbObjectId& styleId
) const;

Parameters

styleId Output style AcDbObjectId.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the Ac Db Object Ids of any stylization entities created during an Update().

```cpp
virtual Acad::ErrorStatus GetStylizationEntities(
    AcDbObjectIdArray& output,
    const AcDbObjectId id,
    void* pCookie,
    Adesk::UInt32 flag = 0
);
```

Parameters
- **output**: Output array of stylization entities.
- **id**: Target entity against which the style was applied.
- **pCookie**: Input opaque memory value.
- **flag**: Reserved. For internal use only.

Returns
Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMSETextStyle Class, AcMapDMSETextStyle Class
AcMapDMSETextStyle:: HeightIsExpression Method
AcMapDMSETextStyle Class | AcMapDMSETextStyle Class

Determines whether the height is an expression.

```cpp
bool HeightIsExpression();
```

Returns

Returns true if the height is an expression.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links

AcMapDMSETextStyle Class, AcMapDMSETextStyle Class
AcMapDMSETextStyle:: JustificationIsExpression Method
AcMapDMSETextStyle Class | AcMapDMSETextStyle Class

Determines whether justification is an expression.

```cpp
bool JustificationIsExpression();
```

Returns true if justification is an expression.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determines whether the layer is an expression.

```cpp
bool LayerIsExpression();
```

Returns true if the layer is an expression.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determine whether the rotation is an expression.

```cpp
bool RotationIsExpression();
```

Returns true if the rotation is an expression.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the text color.

```cpp
Acad::ErrorStatus SetColor(
    AcCmColor color
);
```

Parameters | Description
---|---
color | Input text color.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the text height.

```cpp
Acad::ErrorStatus SetHeight(
    double dHeight
);
```

Parameters | Description
-----------|------------
dHeight    | Input text height.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the text justification.

```cpp
Acad::ErrorStatus SetJustification(
    Justification justification
);
```

**Parameters**

- **justification**
  
  **Description**
  
  Input Justification value.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the layer by id.

```cpp
Acad::ErrorStatus SetLayer(
    const AcDbObjectId layerId
);
```

- **Parameters**
  - `layerId`: Input layer AcDbObjectId.

- **Returns**
  - Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the text rotation.

`Acad::ErrorStatus SetRotation(`

```cpp
doctorate dRotation
```

Parameters Description
---
dRotation Input rotation.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the text style by id.

```cpp
Acad::ErrorStatus SetStyle(
    const AcDbObjectId styleId
);
```

Parameters

- `styleId`: Input style `AcDbObjectId`.

Returns

Returns `Acad::eOk` if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Determine whether the style is an expression.

```cpp
bool StyleIsExpression();
```

Returns true if the style is an expression.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Builds stylization information for later regeneration.

```cpp
virtual Acad::ErrorStatus Update(
    void*& pCookie,
    const AcDbObjectId id,
    Adesk::UInt32 flag = 0
);
```

**Parameters**

- **pCookie**: Input opaque memory value for later access by Dismiss().
- **id**: Input object ID.
- **flag**: Reserved. For internal use only.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

**Remarks**

Derived classes can perform any one-time Update()-time actions here. At Dismiss() time, the implementation can do any required work to clean up storage created at Update() time.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapDMStyle Class
AcMapDMStyle:: CachableProperties Enumeration
AcMapDMStyle Class

Enumerates ADE non-geometric DOT variables.

```cpp
enum CachableProperties {
    kUndefined,
    kTag,
    kString,
    kBlockName,
    kClassName,
    kImageName,
    kLayerName,
    kLinetypeName,
    kPlotStyleName,
    kShapeName,
    kTextStyleName,
    kURL,
    kTopoName,
    kLineWeight,
    kColorACI,
    kTrueColor
};
```

File
DmDisplayStyle.h

Parameters | Description
--- | ---
kUndefined | For internal use only.
kTag | AcDbAttribute tag. Pass as ACHAR*.
kString | AcDbText or attribute string value. Pass as ACHAR*.
kBlockName | Block name. Pass as ACHAR*.
kClassName | Feature classification class name. Pass as ACHAR*.
kImageName | Image filename. Pass as ACHAR*.
kLayerName | Layer name. Pass as ACHAR*.
kLinetypeName | Linetype name. Pass as ACHAR*.
kPlotStyleName | Plotstyle name. Pass as ACHAR*.
kShapeName | Shape name. Pass as ACHAR*. 
kTextStyleName  Text style name. Pass as ACHAR*.
kURL          kURL (exact string). Pass as ACHAR*.
kTopoName     ADE Topology name. Pass as ACHAR*.
kLineWeight   Lineweight. Pass as int.
kColorACI     ACI color. Pass as int.
kTrueColor    True color. Pass as AcCmColor.

Remarks

Styles that modify one of the listed properties must cache the stylized version of the value at Update() time by using the appropriate CacheStylizedPropValue() function.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Applies this style to a specified entity.

```cpp
virtual Acad::ErrorStatus Apply(
    void*& pCookie,
    AcDbEntity* pEnt,
    Adesk::UInt32 flag = 0
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pCookie</td>
<td>Input memory value used by styles to transfer cached data across time from one stylization method to another. Cookie memory is considered to be opaque for existing styles, and should not be modified by the caller.</td>
</tr>
<tr>
<td>pEnt</td>
<td>Input entity to apply the style to. This entity must have been opened for write.</td>
</tr>
<tr>
<td>flag</td>
<td>Reserved. For internal use only.</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Caches the stylized color property value.

```cpp
Acad::ErrorStatus CacheStylizedPropValue(
    AcDbObjectId entityId,
    CachableProperties key,
    const AcCmColor& color
) const;
```

**Parameters**
- `entityId`: Input ID of the entity.
- `key`: Input CachableProperties key.
- `color`: Input color value.

**Returns**
Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](#). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Caches the stylized integer property value.

```cpp
Acad::ErrorStatus CacheStylizedPropValue(
    AcDbObjectId entityId,
    CachableProperties key,
    int value
) const;
```

**Parameters**

- **entityId**: Input ID of the entity.
- **key**: Input CachableProperties key.
- **value**: Input integer value.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links

AcMapDMStyle Class, AcMapDMStyle Class
AcMapDMStyle:: copyFrom Method
AcMapDMStyle Class | AcMapDMStyle Class

Copies the contents of an object into the messaged object, if feasible. See also copyFrom() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus copyFrom(
    const AcRxObject* other
);
```

Parameters

- `other`: Input pointer to the object to copy from.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMStyle Class, AcMapDMStyle Class
AcMapDMStyle:: DecrementRef Method
AcMapDMStyle Class | AcMapDMStyle Class

Decrements the reference count to indicate there is one fewer client using this style. If the reference count becomes zero, the style erases itself. You do not need to call this function directly unless you are implementing an AcMapDMStyleReference class. See also IncrementRef().

Adesk::UInt32 DecrementRef();
Returns

Returns number of references to this style.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Deletes the style's cookie during database destruction.

```cpp
virtual void DeleteCookie(
    void*& pCookie
);
```

### Parameters

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pCookie</td>
</tr>
</tbody>
</table>

### Returns

Returns nothing.

### Remarks

Because the database is in the midst of destruction, do not attempt any additional work by using the cookie at this point, other than its deletion.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Removes the stylization information that is used during regeneration.

```cpp
virtual Acad::ErrorStatus Dismiss(
    void*& pCookie,
    const AcDbObjectId id,
    Adesk::UInt32 flag = 0
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pCookie</td>
<td>Input memory value used by styles to transfer cached data across time from one stylization method to another. Cookie memory is considered to be opaque for existing styles, and should not be modified by the caller.</td>
</tr>
<tr>
<td>id</td>
<td>Input object ID.</td>
</tr>
<tr>
<td>flag</td>
<td>Reserved. For internal use only.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://toolsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMStyle Class, AcMapDMStyle Class
AcMapDMStyle::dwgInFields Method
AcMapDMStyle Class | AcMapDMStyle Class

Let's this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

**virtual** Acad::ErrorStatus dwgInFields(
    AcDbDwgFiler* pFiler
);

Parameters Description
pFiler Input filer to use to read the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMStyle Class, AcMapDMStyle Class
AcMapDMStyle::dwgOutFields Method
AcMapDMStyle Class | AcMapDMStyle Class

Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```
virtual Acad::ErrorStatus dwgOutFields(
    AcDbDwgFiler* pFiler
) const;
```

Parameters Description
pFiler Input filer to use to write the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMStyle Class, AcMapDMStyle Class
AcMapDMStyle:: dxfInFields Method
AcMapDMStyle Class | AcMapDMStyle Class

Lets this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```c++
virtual Acad::ErrorStatus dxfInFields(
    AcDbDxfFiler* pFiler
);
```

Parameters Description
pFiler Input filer to use to read the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMStyle Class, AcMapDMStyle Class
AcMapDMStyle:: dxfOutFields Method
AcMapDMStyle Class | AcMapDMStyle Class

Lets this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dxfOutFields(
    AcDbDxfFiler* pFiler
) const;
```

Parameters Description
pFiler Input filer to use to write the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
**Enable Method**

Enables or disables the style for the specified entity.

```cpp
virtual Acad::ErrorStatus Enable(
    void* & pCookie,
    const AcDbObjectId id,
    bool bEnable,
    Adesk::UInt32 flag = 0
);
```

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pCookie</td>
<td>Input memory value used by styles to transfer cached data across time from one stylization method to another. Cookie memory is considered to be opaque for existing styles, and should not be modified by the caller.</td>
</tr>
<tr>
<td>id</td>
<td>Input entity ID.</td>
</tr>
<tr>
<td>bEnable</td>
<td>Input true to enable the style.</td>
</tr>
<tr>
<td>flag</td>
<td>Reserved. For internal use only.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the name of this style. See also SetName().

`virtual const ACHAR* GetName() const;`

Returns the style name, or NULL if unsuccessful.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the AcDbObjectId of any stylization entities created during an Update().

```cpp
virtual Acad::ErrorStatus GetStylizationEntities(
    AcDbObjectIdArray& output,
    const AcDbObjectId id,
    void* pCookie,
    Adesk::UInt32 flag = 0
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>output</td>
<td>Output array of stylization entities.</td>
</tr>
<tr>
<td>id</td>
<td>Target entity against which the style was applied.</td>
</tr>
<tr>
<td></td>
<td>Input memory value used by styles to transfer cached data across time from one stylization method to another. Cookie memory is considered to be opaque for existing styles, and should not be modified by the caller.</td>
</tr>
<tr>
<td>pCookie</td>
<td></td>
</tr>
<tr>
<td>flag</td>
<td>Reserved. For internal use only.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns the implementation object.

```cpp
virtual AcMapDMImpStyle* Implementation();
```

Returns the implementation object, or NULL if unsuccessful.

Remarks

This function is for internal use only; do not call it directly.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links

AcMapDMStyle Class, AcMapDMStyle Class
AcMapDMStyle:: IncrementRef Method
AcMapDMStyle Class | AcMapDMStyle Class

Increments the reference count to indicate that there is one more client using this style. You do not need to call this function directly unless you are implementing an AcMapDMStyleReference class. The caller must call DecrementRef() when finished with this style.

Adesk::UInt32 IncrementRef();
Returns

Returns the number of references to this style.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determines whether this style has multiple references.

```cpp
bool IsMultiplyReferenced() const;
```

Returns true if multiple references exist.

Created with a commercial version of [Doc-O-Matic](https://www.docOMATIC.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Invoked when the an AutoCAD Map project is initialized.

```cpp
virtual bool OnMapProjectInitialized(
    AcMapProject* pProject
);
```

**Parameters**

- `pProject` Input AutoCAD Map project.

**Returns**

Returns true if successful.

**Remarks**

The style can perform any project-related initialization at this time, and should return true if successful, or false otherwise.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Invoked after a style has been appended to the database, allowing the appended style to do any necessary post-append initialization.

```cpp
virtual void OnObjectAppended();
```

Returns

Returns nothing.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the AutoCAD Map project.

AcMapProject* Project();

Returns

Returns the AutoCAD Map project, or NULL if unsuccessful.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Invoked from within close() before the close actually occurs. The default implementation of this function returns Acad::eOk. See also subClose() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus subClose();
```

Returns Acad::eOk if successful.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Invoked from within erase() before the erase actually occurs. See also subErase() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus subErase(
    Adesk::Boolean erasing
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Passed-in copy of the erasing argument that was passed to the erase() function call that triggered this subErase() call.</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Removes the style from a specified entity.

```c++
virtual Acad::ErrorStatus UnApply(
    void* & pCookie,
    AcDbEntity* pEnt,
    Adesk::UInt32 flag = 0
);
```

**Parameters**

- **pCookie**: Input memory value used by styles to transfer cached data across time from one stylization method to another. Cookie memory is considered to be opaque for existing styles, and should not be modified by the caller.
- **pEnt**: Input entity.
- **flag**: Reserved. For internal use only.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

**Remarks**

The caller can retrieve this style's cookie, stored previously by another stylization function.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Builds stylization information for later regeneration.

```cpp
virtual Acad::ErrorStatus Update(
    void*& pCookie,
    const AcDbObjectId id,
    Adesk::UInt32 flag = 0
);
```

### Parameters

- **pCookie**: Output memory value used by styles to transfer cached data across time from one stylization method to another. Cookie memory is considered to be opaque for existing styles, and should not be modified by the caller.
- **id**: Input object ID.
- **flag**: Reserved. For internal use only.

### Returns

- Returns Acad::eOk if successful; otherwise, returns a different error code.

### Remarks

Derived classes can perform any one-time Update()-time actions here. AtDismiss() time, the implementation can do any required work to clean up storage created at Update() time.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Clones this style. See also wblockClone() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus wblockClone(
    AcRxObject* pOwnerObject,
    AcDbObject*& pClonedObject,
    AcDbIdMapping& idMap,
    Adesk::Boolean isPrimary = true
) const;
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pOwnerObject</td>
<td>Input object to append the clone to.</td>
</tr>
<tr>
<td>pClonedObject</td>
<td>Returned cloned object, or NULL if not cloned.</td>
</tr>
<tr>
<td>idMap</td>
<td>Input current object map ID.</td>
</tr>
<tr>
<td>isPrimary</td>
<td>Input true if this object is primary, or false if it is owned.</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

```cpp
virtual ~AcMapDMStyleCategory(
    void
);
```

Parameters | Description
---|---
void | Void.

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Constructs an instance of this class.

```cpp
AcMapDMStyleCategory(  
    void  
);
```

Parameters Description
void Void.

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMStyleCategory Class, AcMapDMStyleCategory Class
AcMapDMStyleCategory:: Append Method
AcMapDMStyleCategory Class | AcMapDMStyleCategory Class

Adds a style to this category.

Acad::ErrorStatus Append(
   AcDbObjectId& Id,
   AcMapDMStyle* pStyle
);

Parameters   Description
Id            Output ID of the style.
pStyle        Input AcMapDMStyle object.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Copies a style and appends it to this category.

```
Acad::ErrorStatus AppendStyleCopy(
    AcDbObjectId& newId,
    AcDbObjectId& existingId
);
```

**Parameters**

- **newId**
  - Description: Output ID of the new style.

- **existingId**
  - Description: Input style ID.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://toolsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Copies multiple styles and appends them to this category.

```cpp
Acad::ErrorStatus AppendStyleCopy(
    AcDbObjectIdArray& newIdArray,
    AcDbObjectIdArray& existingIdArray)
```

Parameters

- **newIdArray**: Output array of IDs of the new styles.
- **existingIdArray**: Input array of IDs of existing styles.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
This function is called by AutoCAD when the AUDIT command is executed.

```cpp
virtual Acad::ErrorStatus audit(
    AcDbAuditInfo* pAuditInfo
);
```

### Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>pAuditInfo</code></td>
<td>The <code>AcDbAuditInfo</code> object pointed to by <code>pAuditInfo</code> contains member functions that are used to determine what to do and also to report the results of the audit operation on the object.</td>
</tr>
</tbody>
</table>

### Returns

Returns `Acad::eOk` if successful, `Acad::eFixedAllErrors` if all errors were corrected, `Acad::eLeftErrorsUnfixed` if there were errors that were left unfixed; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Let this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dwgInFields(
    AcDbDwgFiler* pFiler
);
```

Parameters

- **pFiler**: Input filer to use to read the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMStyleCategory Class, AcMapDMStyleCategory Class
AcMapDMStyleCategory:: dwgOutFields Method
AcMapDMStyleCategory Class | AcMapDMStyleCategory Class

Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dwgOutFields(
    AcDbDwgFiler* pFiler
) const;
```

Parameters | Description
--- | ---
pFiler | Input filer to use to write the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.doc-o-matic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMStyleCategory Class, AcMapDMStyleCategory Class
AcMapDMStyleCategory::dxfInFields Method
AcMapDMStyleCategory Class | AcMapDMStyleCategory Class

Lets this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dxfInFields(
    AcDbDxfFiler* pFiler
);
```

Parameters
pFiler

Input filer to use to read the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMStyleCategory Class, AcMapDMStyleCategory Class
AcMapDMStyleCategory:: dxfOutFields Method
AcMapDMStyleCategory Class | AcMapDMStyleCategory Class

Lets this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

virtual Acad::ErrorStatus dxfOutFields(
    AcDbDxfFiler* pFiler
) const;

Parameters Description
pFiler Input filer to use to write in the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMStyleCategory Class, AcMapDMStyleCategory Class

AcMapDMStyleCategory:: erased Method
AcMapDMStyleCategory Class | AcMapDMStyleCategory Class

Lets this object listen to erase-notifications from items that it owns. See also erased() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

**virtual void** erased(
    const AcDbObject* dbObj,
    Adesk::Boolean bErasing = true
);

Parameters  Description
---  ---
dbObj  Input object that was erased.
bErasing  Input true to listen as an erase is happening. The default value is true.

Returns

Returns nothing.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Finds a specified style within this category.

```cpp
bool Find(
    Adesk::UInt16& nIndex,
    const AcDbObjectId& Id
) const;
```

**Parameters**

- `nIndex`  
  Output index of the style, if found.
- `Id`  
  Input ID of the style to find.

**Returns**

Returns true if style is found; otherwise, returns false.

Created with a commercial version of [Doc-O-Matic](https://toolsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the ID of a style at a specified position.

\[
\text{Acad::ErrorStatus GetAt(}
\text{    AcDbObjectId} \& \text{ Id,}
\text{    Adesk::UInt16 nIndex}
\text{) const;}
\]

Parameters Description
Id Output ID of the style.
nIndex Input position of the style.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves a style at a specified position, opened in a specified mode.

```cpp
Acad::ErrorStatus GetAt(
    AcMapDMStyle*& pStyle,
    Adesk::UInt16 nIndex,
    AcDb::OpenMode mode
) const;
```

**Parameters**

- `pStyle`: Output AcMapDMStyle object.
- `nIndex`: Input position of the style.
- `mode`: Input mode in which the style is to be opened.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Retrieves a style, opened in a specified mode.

```cpp
Acad::ErrorStatus GetAt(
    AcMapDMStyle*& pStyle,
    const AcDbObjectId& Id,
    AcDb::OpenMode mode
) const;
```

### Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pStyle</td>
<td>Output AcMapDMStyle object.</td>
</tr>
<tr>
<td>Id</td>
<td>Input ID of the style.</td>
</tr>
<tr>
<td>mode</td>
<td>Input mode in which the style is to be opened.</td>
</tr>
</tbody>
</table>

### Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the name of this category. See also SetName().

```cpp
virtual const ACHAR* GetName() const;
```

Returns the category name if successful; otherwise, returns NULL.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determines whether this category has a specified style.

```cpp
bool Has(
    const AcDbObjectId& objId
) const;
```

Parameters
- `objId`: Input ID of the style.

Returns

Returns true if the category has the specified style.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns the implementation object.

```c
AcMapDMImpStyleCategory* Implementation();
```

Returns the implementation object, or NULL if unsuccessful.

Remarks

This function is for internal use only; do not call it directly.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Inserts a style at a specified position.

```
Acad::ErrorStatus InsertAt(
    AcDbObjectId& Id,
    AcMapDMStyle* pStyle,
    Adesk::UInt16 nIndex
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Id</td>
<td>Output ID of the style.</td>
</tr>
<tr>
<td>pStyle</td>
<td>Input AcMapDMStyle object.</td>
</tr>
<tr>
<td>nIndex</td>
<td>Input position.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Inserts a copy of a style at a specified position.

```
Acad::ErrorStatus InsertStyleCopyAt(
    AcDbObjectId& newId,
    AcDbObjectId& existingId,
    Adesk::UInt16 nIndex
);
```

**Parameters**

- **newId** Output ID of the inserted style.
- **existingId** Input ID of the existing style.
- **nIndex** Input position of the inserted style.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMStyleCategory Class, AcMapDMStyleCategory Class
AcMapDMStyleCategory::InsertStyleCopyAt Method
AcMapDMStyleCategory Class | AcMapDMStyleCategory Class

Inserts multiple styles at a specified position.

```
Acad::ErrorStatus InsertStyleCopyAt(
   AcDbObjectIdArray& newIdArray,
   AcDbObjectIdArray& existingIdArray,
   Adesk::UInt16 nIndex
);
```

**Parameters**

- `newIdArray` Output array of IDs of the inserted styles.
- `existingIdArray` Input array of IDs of existing styles.
- `nIndex` Input position of the inserted styles.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Moves a style to a new position.

```cpp
Acad::ErrorStatus Move(
    Adesk::UInt16 nOldIndex,
    Adesk::UInt16 nNewIndex
);
```

**Parameters**

- `nOldIndex`: Input old position.
- `nNewIndex`: Input new position.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Moves a style to a new position.

```cpp
Acad::ErrorStatus Move(
    const AcDbObjectId& Id,
    Adesk::UInt16 nNewIndex
);
```

### Parameters
- **Id**: Input style ID.
- **nNewIndex**: Input new position.

### Returns
Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Counts the number of styles within this category.

Adesk::UInt16 NumStyles() const;

Returns the number of styles.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Removes a style.

```cpp
Acad::ErrorStatus Remove(
    const AcDbObjectId& Id
);
```

Parameters Description
Id Input ID of the style to remove.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Removes a style at a specified position.

```
Acad::ErrorStatus RemoveAt(
    Adesk::UInt16 nIndex
);
```

### Parameters

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>nIndex</td>
</tr>
</tbody>
</table>

### Description

Input position of the style to remove.

### Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Invoked from within close() before the close actually occurs. The default implementation of this function returns Acad::eOk. See also subClose() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus subClose();
```

Returns Acad::eOk if successful.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com/). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Invoked from within erase() before the erase actually occurs. See also subErase() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus subErase(
    Adesk::Boolean erasing);
```

**Parameters**
- **erasing**

  Passed-in copy of the erasing argument that was passed to the erase() function call that triggered this subErase() call.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](#). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Destroys an instance of this class.

```cpp
virtual ~AcMapDMStyleLibrary( 
    void
);
```

Parameters

- Description:

Returns

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Constructs an instance of this class.

```c
AcMapDMStyleLibrary(
    void
);
```

Parameters | Description
---|---
void | Void.

Returns

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapDMStyleLibrary Class, AcMapDMStyleLibrary Class
AcMapDMStyleLibrary:: Append Method
AcMapDMStyleLibrary Class | AcMapDMStyleLibrary Class

Add a style category to this library.

```
Acad::ErrorStatus Append(
    AcDbObjectId& Id,
    AcMapDMStyleCategory* pStyleCategory
);
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Id</td>
<td>Output ID of the category.</td>
</tr>
<tr>
<td>pStyleCategory</td>
<td>Input AcMapDMStyleCategory object.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
This function is called by AutoCAD when the AUDIT command is executed.

```cpp
virtual Acad::ErrorStatus audit(
    AcDbAuditInfo* pAuditInfo
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pAuditInfo</td>
<td>The AcDbAuditInfo object pointed to by pAuditInfo contains member functions that are used to determine what to do and also to report the results of the audit operation on the object.</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful, Acad::eFixedAllErrors if all errors were corrected, Acad::eLeftErrorsUnfixed if there were errors that were left unfixed; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapDMStyleLibrary Class, AcMapDMStyleLibrary Class
AcMapDMStyleLibrary::dwgInFields Method
AcMapDMStyleLibrary Class | AcMapDMStyleLibrary Class

Lets this object read its data. See also dwgInFields() in the AutoCAD
ObjectARX Developer's Guide. This overloaded function is called by the system
as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dwgInFields(
    AcDbDwgFiler* pFiler
);
```

Parameters | Description
--- | ---
pFiler | Input filer to use to read the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this
message disappear you need to register this software. If you have problems
registering this software please contact us at support@toolsfactory.com.
AcMapDMStyleLibrary::dwgOutFields Method

```
virtual Acad::ErrorStatus dwgOutFields(
    AcDbDwgFiler* pFiler
) const;
```

### Parameters

- **pFiler**: Input filer to use to write the object's data.

### Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.toolsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapDMStyleLibrary Class, AcMapDMStyleLibrary Class
AcMapDMStyleLibrary::dxfInFields Method
AcMapDMStyleLibrary Class | AcMapDMStyleLibrary Class

Lets this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```
virtual Acad::ErrorStatus dxfInFields(
    AcDbDxfFiler* pFiler
);
```

Parameters | Description
--- | ---
pFiler | Input filer to use to read the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMStyleLibrary Class, AcMapDMStyleLibrary Class
AcMapDMStyleLibrary::dxfOutFields Method
AcMapDMStyleLibrary Class | AcMapDMStyleLibrary Class

Lets this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dxfOutFields(
   AcDbDxfFiler* pFiler
) const;
```

Parameters

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pFiler</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
 lets this object listen to erase-notifications from items that it owns. See also erased() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual void erased(
    const AcDbObject* dbObj,
    Adesk::Boolean bErasing = true
);
```

**Parameters**

- **dbObj**
  Input object that was erased.

- **bErasing**
  Input true to listen as an erase is happening. The default value is true.

**Returns**

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Retrieves a category at a specified position.

```cpp
Acad::ErrorStatus GetAt(
    AcDbObjectId& Id,
    Adesk::UInt16 nIndex
) const;
```

Parameters | Description
---|---
Id | Output ID of the category.
nIndex | Input position of the category.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves a category at a specified position, opened in a specified mode.

```
Acad::ErrorStatus GetAt(
    AcMapDMStyleCategory*& pStyleCategory,
    Adesk::UInt16 nIndex,
    AcDb::OpenMode mode
) const;
```

**Parameters**
- `pStyleCategory` Output `AcMapDMStyleCategory` object.
- `nIndex` Input position of the category.
- `mode` Input mode in which the category is to be opened.

**Returns**
Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves a category, opened in a specified mode.

\[
\text{Acad::ErrorStatus GetAt(}
\quad \text{AcMapDMStyleCategory}\,*&\ pStyleCategory,
\quad \text{const AcDbObjectId}\,*&\ Id,
\quad \text{AcDb::OpenMode}\ mode
\)\ const;
\]

**Parameters**

- **pStyleCategory**
  - Output AcMapDMStyleCategory object.
- **Id**
  - Input ID of the category.
- **mode**
  - Input mode in which the category is to be opened.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determines whether a specific category exists in this library.

```cpp
bool Has(
    const AcDbObjectId& Id
) const;
```

**Parameters**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Id</td>
</tr>
</tbody>
</table>

**Returns**

Returns true if the category exists.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMStyleLibrary Class, AcMapDMStyleLibrary Class
AcMapDMStyleLibrary:: Implementation Method
AcMapDMStyleLibrary Class | AcMapDMStyleLibrary Class

Returns the implementation object.

AcMapDMImpStyleLibrary* Implementation();
Returns

Returns the implementation object, or NULL if unsuccessful.

Remarks

This function is for internal use only; do not call it directly.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Inserts a category at a specified position.

```
Acad::ErrorStatus InsertAt(
    AcDbObjectId& Id,
    AcMapDMStyleCategory* pStyleCategory,
    Adesk::UInt16 nIndex
);
```

**Parameters**

- **Id**
  - Output ID of the category.
- **pStyleCategory**
  - Input `AcMapDMStyleCategory` object.
- **nIndex**
  - Input position.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links

AcMapDMStyleLibrary Class, AcMapDMStyleLibrary Class

AcMapDMStyleLibrary:: Move Method

AcMapDMStyleLibrary Class | AcMapDMStyleLibrary Class

Moves a category from one position to another in this library.

```cpp
Acad::ErrorStatus Move(
    Adesk::UInt16 nOldIndex,
    Adesk::UInt16 nNewIndex
);
```

Parameters

- **nOldIndex**
  - Input old position.
- **nNewIndex**
  - Input new position.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
AcMapDMStyleLibrary:: Move Method

AcMapDMStyleLibrary:: Move Method

Moves a category with a specified ID to another position in this library.

Acad::ErrorStatus Move(
    const AcDbObjectId& Id,
    Adesk::UInt16 nNewIndex
);

Parameters Description
Id Input ID of the category.
nNewIndex Input new position.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Counts the number of categories within this library.

Adesk::UInt16 NumCategories() const;
Returns

Returns the number of categories.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Removes a category from this library.

Acad::ErrorStatus Remove(
    const AcDbObjectId& Id
);

Parameters   Description
Id           Input ID of the category to remove.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Removes a category at a specified position from this library.

```cpp
Acad::ErrorStatus RemoveAt(
    Adesk::UInt16 nIndex
);
```

- **nIndex**: Input position of the category to remove.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

```
virtual ~AcMapDMStyleReference();
```

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Constructs an instance of this class.

AcMapDMStyleReference();

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
This function is called by AutoCAD when the AUDIT command is executed.

```cpp
virtual Acad::ErrorStatus audit(
    AcDbAuditInfo* pAuditInfo
);
```

**Parameters**

- `pAuditInfo`: The AcDbAuditInfo object pointed to by `pAuditInfo` contains member functions that are used to determine what to do and also to report the results of the audit operation on the object.

**Returns**

Returns Acad::eOk if successful, Acad::eFixedAllErrors if all errors were corrected, Acad::eLeftErrorsUnfixed if there were errors that were left unfixed; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMStyleReference Class, AcMapDMStyleReference Class
AcMapDMStyleReference:: dwgInFields Method
AcMapDMStyleReference Class | AcMapDMStyleReference Class

Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```
virtual Acad::ErrorStatus dwgInFields(
   AcDbDwgFiler* pFiler
);
```

Parameters                     Description
pFiler                        Input filer to use to read the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMStyleReference Class, AcMapDMStyleReference Class
AcMapDMStyleReference:: dwgOutFields Method
AcMapDMStyleReference Class | AcMapDMStyleReference Class

Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dwgOutFields(
    AcDbDwgFiler* pFiler
) const;
```

Parameters Description
pFiler Input filer to use to write the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMStyleReference Class, AcMapDMStyleReference Class
AcMapDMStyleReference:: dxfInFields Method
AcMapDMStyleReference Class | AcMapDMStyleReference Class

Lets this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

**virtual** Acad::ErrorStatus dxfInFields(
    AcDbDxfFiler* pFiler
);

Parameters | Description
---|---
pFiler | Input filer to use to read the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
AcMapDMStyleReference Class, AcMapDMStyleReference Class
AcMapDMStyleReference:: dxfOutFields Method
AcMapDMStyleReference Class | AcMapDMStyleReference Class

Lets this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dxfOutFields(
    AcDbDxfFiler* pFiler
) const;
```

Parameters Description
pFiler Input filer to use to write in the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns the implementation object.

\[\text{AcMapDMImpStyleReference}\star\ \text{Implementation() const;}\]

Returns the implementation object, or NULL if unsuccessful.

Remarks

This function is for internal use only; do not call it directly.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMStyleReference Class, AcMapDMStyleReference Class
AcMapDMStyleReference:: IsEnabled Method
AcMapDMStyleReference Class | AcMapDMStyleReference Class

Determines whether this reference is enabled. See also SetEnabled().

```cpp
bool IsEnabled() const;
```

Returns

Returns true if the reference is enabled, or false if it is disabled.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determine whether this reference is the only reference to the style that this reference refers to. See also MakeUniqueReference().

```cpp
bool IsUniqueReference() const;
```

Returns true if this is the only reference; otherwise, returns false.

Created with a commercial version of [Doc-O-Matic](https://toolsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Determines whether this reference is a unique reference and, if not, clones the referenced style and resets the style ID to the new copy. See also IsUniqueReference().

`Acad::ErrorStatus MakeUniqueReference();`

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Enables or disables this reference. See also IsEnabled().

```cpp
Acad::ErrorStatus SetEnabled(
    bool bNewStatus
);
```

**Parameters**

- **bNewStatus**: Input new enabled status.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMStyleReference Class, AcMapDMStyleReference Class
AcMapDMStyleReference:: StyleId Method
AcMapDMStyleReference Class | AcMapDMStyleReference Class

Retrieves the ID of the style that this reference refers to.

AcDbObjectId StyleId() const;
Returns

Returns the style ID if successful.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Invoked from within close() before the close actually occurs. The default implementation of this function returns Acad::eOk. See also subClose() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus subClose();
```

Returns Acad::eOk if successful.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Invoked from within erase() before the erase actually occurs. See also subErase() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus subErase(
    Adesk::Boolean erasing
);
```

### Parameters

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Passed-in copy of the erasing argument that was passed to the erase() function call that triggered this subErase() call.</td>
</tr>
</tbody>
</table>

### Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Grants control of deep clone operations to the object. In the default implementation, the object is cloned and appended to the owner object pOwnerObject. See also wblockClone() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus wblockClone(
    AcRxObject* pOwnerObject,
    AcDbObject*& pClonedObject,
    AcDbIdMapping& idMap,
    Adesk::Boolean isPrimary = true
) const;
```

Parameters

- **pOwnerObject**: Input object to append the clones to.
- **pClonedObject**: Output the cloned object, or NULL if not cloned.
- **idMap**: Input current ID map.
- **isPrimary**: Input true if this object is primary, or false if it is owned.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Applies the style to an entity.

```cpp
virtual Acad::ErrorStatus Apply(
    void*& pCookie,
    AcDbEntity* pEnt,
    Adesk::UInt32 flag = 0
);
```

**Parameters**

- **pCookie**: Input memory value used by styles to transfer cached data across time from one stylization method to another. Cookie memory is considered to be opaque for existing styles, and should not be modified by the caller.

- **pEnt**: Input entity to apply the style to.

- **flag**: Reserved. For internal use only.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Informs the display manager that a particular AcDbEntity was created as a result of applying a StylizationEntityStyle.

```cpp
Acad::ErrorStatus CacheStylizationEntity(
    const AcDbObjectId styleEntId
) const;
```

Parameters

- **styleEntId**: Input ID of the stylization entity.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Remarks

A derived class must call this function when it creates a new stylization entity.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at **support@toolsfactory.com**.
Removes stylization information that is used during regeneration.

```cpp
virtual Acad::ErrorStatus Dismiss(
    void*& pCookie,
    const AcDbObjectId id,
    Adesk::UInt32 flag = 0
);
```

### Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pCookie</td>
<td>Input memory value used by styles to transfer cached data across time from one stylization method to another. Cookie memory is considered to be opaque for existing styles, and should not be modified by the caller.</td>
</tr>
<tr>
<td>id</td>
<td>Input object ID.</td>
</tr>
<tr>
<td>flag</td>
<td>Reserved. For internal use only.</td>
</tr>
</tbody>
</table>

### Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Lets this object read its data. See also dwgInFields() in the AutoCAD
ObjectARX Developer's Guide. This overloaded function is called by the system
as needed; it is unlikely that you will need to call it directly.

\textbf{virtual} Acad::ErrorStatus dwgInFields(
       AcDbDwgFiler* pFiler
);

Parameters Description
pFiler Input filer to use to read the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of \textbf{Doc-O-Matic}. In order to make this
message disappear you need to register this software. If you have problems
registering this software please contact us at support@toolsfactory.com.
Links

AcMapDMStylizationEntityStyle Class, AcMapDMStylizationEntityStyle Class
AcMapDMStylizationEntityStyle:: dwgOutFields Method
AcMapDMStylizationEntityStyle Class | AcMapDMStylizationEntityStyle Class

Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dwgOutFields(
    AcDbDwgFiler* pFiler
) const;
```

Parameters  

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pFiler</td>
</tr>
</tbody>
</table>

Returns  

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Let this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

**virtual** Acad::ErrorStatus dxfInFields(
  AcDbDxfFiler* pFiler
);

Parameters
- **pFiler**: Input filer to use to read the object's data.

Returns
- Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMStylizationEntityStyle Class, AcMapDMStylizationEntityStyle Class
AcMapDMStylizationEntityStyle:: dxfOutFields Method
AcMapDMStylizationEntityStyle Class | AcMapDMStylizationEntityStyle Class

Lets this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dxfOutFields(
    AcDbDxfFiler* pFiler
) const;
```

Parameters

- **pFiler**
  - Input filer to use to write in the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Enables or disables the style for the specified entity.

```cpp
virtual Acad::ErrorStatus Enable(
    void* & pCookie,
    const AcDbObjectId id,
    bool bEnable,
    Adesk::UInt32 flag = 0
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pCookie</td>
<td>Input memory value used by styles to transfer cached data across time from one stylization method to another. Cookie memory is considered to be opaque for existing styles, and should not be modified by the caller.</td>
</tr>
<tr>
<td>id</td>
<td>Input object ID.</td>
</tr>
<tr>
<td>bEnable</td>
<td>Input true to enable the style.</td>
</tr>
<tr>
<td>flag</td>
<td>Reserved. For internal use only.</td>
</tr>
</tbody>
</table>

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMStylizationEntityStyle Class, AcMapDMStylizationEntityStyle Class
AcMapDMStylizationEntityStyle::
ExcludeStylizationEntityFromHatchStyle Method
AcMapDMStylizationEntityStyle Class | AcMapDMStylizationEntityStyle Class

Creates an island around the stylization entity.

```
const AcDbObjectId id,
const AcDbObjectId styleEntId
) const;
```

Parameters Description
id Input ID of the target entity.
styleEntId Input ID of the stylization entity.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Remarks

A derived class must call this function if it creates a stylization entity, and wants the AcMapDMSEHatchStyle to attempt to create an island around the stylization entity, should a stylization hatch happen to be applied to the same target entity.

In this release of AutoCAD Map, islanding works for only AcDbText entities.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the AcDbObjectId of any stylization entities created during an Update().

```cpp
virtual Acad::ErrorStatus GetStylizationEntities(
    AcDbObjectIdArray& output,
    const AcDbObjectId id,
    void* pCookie,
    Adesk::UInt32 flag = 0
);
```

**Parameters**

- **output**: Output array IDs of stylization entities.
- **id**: Target entity against which the style was applied.
- **pCookie**: Input opaque memory value.
- **flag**: Reserved. For internal use only.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Builds stylization information for later regeneration.

```cpp
virtual Acad::ErrorStatus UnApply(
    void*& pCookie,
    AcDbEntity* pEnt,
    Adesk::UInt32 flag = 0
);
```

**Parameters**

- **pCookie**: Input memory value used by styles to transfer cached data across time from one stylization method to another. Cookie memory is considered to be opaque for existing styles, and should not be modified by the caller.
- **pEnt**: Input entity ID.
- **flag**: Reserved. For internal use only.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

**Remarks**

The caller can retrieve this style's cookie, stored previously by another stylization function.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
A derived class must call this function to remove a stylization entity from association with a target entity, for the purposes of AcMapDMSEHatchStyle islanding.

Acad::ErrorStatus UnExcludeStylizationEntityFromHatchStyle(
    const AcDbObjectId id,
    const AcDbObjectId styleEntId
) const;

Parameters

id Input ID of the target entity.
styleEntId Input ID of the stylization entity.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Remarks

This should be done by the Style at dismiss time if the Style may reuse the same stylization entity against a different target entity. In this release of AutoCAD Map, islanding works for only AcDbText entities.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Builds stylization information for later regeneration.

```cpp
virtual Acad::ErrorStatus Update(
    void*& pCookie,
    const AcDbObjectId id,
    Adesk::UInt32 flag = 0
);
```

**Parameters**

- **pCookie**
  - Description: Output memory value used by styles to transfer cached data across time from one stylization method to another. Cookie memory is considered to be opaque for existing styles, and should not be modified by the caller.

- **id**
  - Description: Input object ID.

- **flag**
  - Description: Reserved. For internal use only.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

**Remarks**

Derived classes can perform any one-time Update()-time actions here. At Dismiss() time, the implementation can do any required work to clean up storage created at Update() time.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

`virtual ~AcMapDMThematicStyle();`

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicStyle Class, AcMapDMThematicStyle Class
AcMapDMThematicStyle:: AcMapDMThematicStyle Constructor
AcMapDMThematicStyle Class | AcMapDMThematicStyle Class

Constructs an instance of this class.

AcMapDMThematicStyle();
Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Applies the style to an entity.

```cpp
virtual Acad::ErrorStatus Apply(
    void*& pCookie,
    AcDbEntity* pEnt,
    Adesk::UInt32 flag = 0
);
```

### Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pCookie</td>
<td>Input memory value used by styles to transfer cached data across time from one stylization method to another. Cookie memory is considered to be opaque for existing styles, and should not be modified by the caller.</td>
</tr>
<tr>
<td>pEnt</td>
<td>Input entity to apply the style to.</td>
</tr>
<tr>
<td>flag</td>
<td>Reserved. For internal use only.</td>
</tr>
</tbody>
</table>

### Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determines whether automatic recalculation of ranges is turned on or off.

```cpp
virtual bool AutoRecalculateRanges() const;
```

Returns true if automatic recalculation is turned on.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the expression used in theming.

```cpp
virtual const ACHAR * DataSourceExpression() const;
```

Returns the expression, or NULL if the expression is not set.

Created with a commercial version of [Doc-O-Matic](#). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Deletes the style's cookie during database destruction.

```cpp
virtual void DeleteCookie(
    void*& pCookie
);
```

**Parameters**

- `pCookie` Input cookie to delete.

**Returns**

Returns nothing.

**Remarks**

Because the database is in the midst of destruction, do not attempt any additional work by using the cookie at this point, other than its deletion.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Removes stylization information that is used during regeneration.

```cpp
virtual Acad::ErrorStatus Dismiss(
    void* & pCookie,
    const AcDbObjectId id,
    Adesk::UInt32 flag = 0
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pCookie</td>
<td>Input memory value used by styles to transfer cached data across time from one stylization method to another. Cookie memory is considered to be opaque for existing styles, and should not be modified by the caller.</td>
</tr>
<tr>
<td>id</td>
<td>Input object ID.</td>
</tr>
<tr>
<td>flag</td>
<td>Reserved. For internal use only.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicStyle Class, AcMapDMThematicStyle Class
AcMapDMThematicStyle:: dwgInFields Method
AcMapDMThematicStyle Class | AcMapDMThematicStyle Class

Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

**virtual** Acad::ErrorStatus dwgInFields(
    AcDbDwgFiler* pFiler
);

Parameters
- **pFiler**
  - Input filer to use to read the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at **support@toolsfactory.com**.
Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dwgOutFields(
    AcDbDwgFiler* pFiler
) const;
```

**Parameters**

- `pFiler`  
  Input filer to use to write the object's data.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicStyle Class, AcMapDMThematicStyle Class
AcMapDMThematicStyle::dxfInFields Method
AcMapDMThematicStyle Class | AcMapDMThematicStyle Class

Lets this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dxfInFields(
    AcDbDxfFiler* pFiler
);
```

**Parameters**
- **pFiler** Input filer to use to read the object's data.

**Returns**
Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapDMThematicStyle Class, AcMapDMThematicStyle Class
AcMapDMThematicStyle:: dxfOutFields Method
AcMapDMThematicStyle Class | AcMapDMThematicStyle Class

Lets this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dxfOutFields(
    AcDbDxfFiler* pFiler
) const;
```

Parameters
- **pFiler**: Input filer to use to write in the object's data.

Returns
Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Enables or disables the style for a specified entity.

```cpp
virtual Acad::ErrorStatus Enable(
    void*& pCookie,
    const AcDbObjectId id,
    bool bEnable,
    Adesk::UInt32 flag = 0
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pCookie</td>
<td>Input memory value used by styles to transfer cached data across time from one stylization method to another. Cookie memory is considered to be opaque for existing styles, and should not be modified by the caller.</td>
</tr>
<tr>
<td>id</td>
<td>Input object ID.</td>
</tr>
<tr>
<td>bEnable</td>
<td>Input true to enable the style.</td>
</tr>
<tr>
<td>flag</td>
<td>Reserved. For internal use only.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the IDs of any stylization entities created during an Update().

```cpp
virtual Acad::ErrorStatus GetStylizationEntities(
    AcDbObjectIdArray& output,
    const AcDbObjectId id,
    void* pCookie,
    Adesk::UInt32 flag = 0
);
```

**Parameters**

- `output`: Output array of IDs of stylization entities.
- `id`: Target entity against which the style was applied.
- `pCookie`: Input opaque memory value.
- `flag`: Reserved. For internal use only.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Retrieves the thematic table, opened in a specified mode.

```cpp
virtual Acad::ErrorStatus GetThematicTable(
    AcMapDMThematicTable* &pTable,
    AcDb::OpenMode openMode
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pTable</td>
<td>Output AcMapDMThematicTable thematic table.</td>
</tr>
<tr>
<td>openMode</td>
<td>Input mode in which the table is to be opened.</td>
</tr>
</tbody>
</table>

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Counts the number of groups for thematic ranges.

```cpp
virtual int GroupCount() const;
```

Returns the number of groups.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieve the name of the grouping algorithm.

```cpp
virtual const ACHAR * GroupingAlgorithmName() const;
```

Returns the name of the grouping algorithm.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the name of the rounding method used.

```cpp
virtual const ACHAR * GroupRoundingMethod() const;
```

Returns the name of the rounding method used.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Determines whether annotation alteration is set to modify a thematic stylization query result.

```cpp
virtual bool IsAlterAnnotationEnabled() const;
```

Returns true if alteration is on.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicStyle Class, AcMapDMThematicStyle Class
AcMapDMThematicStyle:: IsAlterBlockEnabled Method
AcMapDMThematicStyle Class | AcMapDMThematicStyle Class

Determine whether block alteration is set to modify a thematic stylization query result.

`virtual bool IsAlterBlockEnabled() const;`

Returns

Returns true if alteration is on.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determines whether color alteration is set to modify a thematic stylization query result.

```cpp
virtual bool IsAlterColorEnabled() const;
```

Returns

Returns true if alteration is on.

Created with a commercial version of [Doc-O-Matic](https://www.doc-o-matic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Determine whether hatch alteration is set to modify a thematic stylization query result.

```cpp
virtual bool IsAlterHatchEnabled() const;
```

Returns true if alteration is on.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determine whether linestyle alteration is set to modify a thematic stylization query result.

```cpp
virtual bool IsAlterLinestyleEnabled() const;
```

Returns true if alteration is on.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determined whether linetype alteration is set to modify a thematic stylization query result.

虚函数  

```cpp
virtual bool IsAlterLinetypeEnabled() const;
```

Returns

Returns true if alteration is on.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Determines whether lineweight alteration is set to modify a thematic stylization query result.

```
virtual bool IsAlterLineweightEnabled() const;
```

Returns true if alteration is on.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicStyle Class, AcMapDMThematicStyle Class
AcMapDMThematicStyle:: IsAlterPlotstyleEnabled Method
AcMapDMThematicStyle Class | AcMapDMThematicStyle Class

Determines whether plotstyle alteration is set to modify a thematic stylization query result.

**virtual bool** `IsAlterPlotstyleEnabled()` **const**;

Returns

Returns true if alteration is on.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determine whether text alteration is set to modify a thematic stylization query result.

```cpp
virtual bool IsAlterTextEnabled() const;
```

Returns

Returns true if alteration is on.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determines whether the scale-color-ramp flag is set.

```cpp
virtual bool IsScaleColorRamp() const;
```

Returns true if the flag is set.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com/). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Determines whether the scale-hatch-ramp flag is set.

```
virtual bool IsScaleHatchRamp() const;
```

Returns true if the flag is set.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determines whether the scale-linestyle-ramp flag is set.

```cpp
virtual bool IsScaleLinestyleRamp() const;
```

Returns true if the flag is set.

Created with a commercial version of [Doc-O-Matic](http://www.docsomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determine whether the scale-ramp-to-fit flag is set.

```cpp
virtual bool IsScaleRampToFit() const;
```

Returns true if the flag is set.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determines whether the data source is numeric.

```cpp
virtual bool IsSourceDataNumeric() const;
```

Returns true if the data source is numeric.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the name of the last-used color ramp.

```cpp
virtual const ACHAR * LastColorRampName() const;
```

Returns

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the name of the last-used hatch ramp.

```cpp
virtual const ACHAR * LastHatchRampName() const;
```

Returns the ramp name.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicStyle Class, AcMapDMThematicStyle Class
AcMapDMThematicStyle:: LastLinestyleRampName Method
AcMapDMThematicStyle Class | AcMapDMThematicStyle Class

Retrieves the name of the last-used linestyle ramp.

virtual const ACHAR * LastLinestyleRampName() const;

Returns

Returns the ramp name.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the linetype scale.

```cpp
virtual double LinetypeScale() const;
```

Returns the linetype scale.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the normalization expression.

```cpp
virtual const ACHAR * NormalizationExpression() const;
```

Returns the expression, or NULL if the expression is not set.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Invoked when an object is assigned to the database.

```cpp
virtual void OnObjectAppended();
```

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the range-table comparison operator.

```cpp
virtual AcMap::ERangeOperator RangeTableComparisonOperator() const;
```

Returns an range-operator enum value.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets annotation alteration.

```cpp
virtual Acad::ErrorStatus SetAlterAnnotation(
    bool bEnable = true
);
```

Parameters | Description
---|---
bEnable | Input true to turn alteration on.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets block alteration.

```
virtual Acad::ErrorStatus SetAlterBlock(
    bool bEnable = true
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bEnable</td>
<td>Input true to turn alteration on.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets color alteration.

```cpp
virtual Acad::ErrorStatus SetAlterColor(
    bool bEnable = true
);
```

### Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bEnable</td>
<td>Input true to turn alteration on.</td>
</tr>
</tbody>
</table>

### Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets hatch alteration.

```cpp
class AcMapDMThematicStyle
{
public:
    virtual Acad::ErrorStatus SetAlterHatch(bool bEnable = true) = 0;
};
```

Parameters

- **bEnable**: Input `true` to turn alteration on.

Returns

Returns `Acad::eOk` if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets linestyle alteration.

```cpp
virtual Acad::ErrorStatus SetAlterLinestyle(
    bool bEnable = true
);
```

**Parameters**

- **bEnable**
  - Description: Input true to turn alteration on.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](http://www.doc-omatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets linetype alteration.

```cpp
virtual Acad::ErrorStatus SetAlterLinetype(
    bool bEnable = true
);
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bEnable</td>
<td>Input true to turn alteration on.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets lineweight alteration.

```cpp
virtual Acad::ErrorStatus SetAlterLineweight(
    bool bEnable = true
);
```

**Parameters**

- **bEnable**: Input true to turn alteration on.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets plotstyle alteration.

```cpp
virtual Acad::ErrorStatus SetAlterPlotstyle(
    bool bEnable = true
);
```

Parameters | Description
---|---
bEnable | Input true to turn alteration on.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](#). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets text alteration.

```cpp
virtual Acad::ErrorStatus SetAlterText(
    bool bEnable = true)
```

Parameters  
- **bEnable**: Input `true` to turn alteration on.

Returns  
Returns `Acad::eOk` if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapDMThematicStyle Class, AcMapDMThematicStyle Class
AcMapDMThematicStyle:: SetAutoRecalculateRanges Method
AcMapDMThematicStyle Class | AcMapDMThematicStyle Class

Turns automatic recalculation on or off.

```cpp
virtual Acad::ErrorStatus SetAutoRecalculateRanges(
    bool bRecalculateRanges
);
```

Parameters

- **bRecalculateRanges**
  - Description: Input true to turn automatic recalculation on.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the number of groups for thematic ranges.

```cpp
virtual Acad::ErrorStatus SetGroupCount(
    int groupCount
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>groupCount</td>
<td>Input number of groups.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the linetype scale.

```cpp
virtual Acad::ErrorStatus SetLinetypeScale(
    double newLinetypeScale
);
```

### Parameters

- **newLinetypeScale**: Input linetype scale.

### Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the range-table comparison operator.

**virtual** Acad::ErrorStatus SetRangeTableComparisonOperator(
    AcMap::ERangeOperator newRangeOperator
);

Parameters

newRangeOperator Input range-operator value.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://toolsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the scale-color-ramp flag.

```cpp
virtual Acad::ErrorStatus SetScaleColorRamp(
    bool bIsScaleColorRamp
);
```

Parameters | Description
---|---
bIsScaleColorRamp | Input true to set the flag.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.toolsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the scale-hatch-ramp flag.

**virtual** Acad::ErrorStatus SetScaleHatchRamp(
    bool bIsScaleHatchRamp
);

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bIsScaleHatchRamp</td>
<td>Input true to set the flag.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the scale-linestyle-ramp flag.

```cpp
virtual Acad::ErrorStatus SetScaleLinestyleRamp(
    bool bIsScaleLinestyleRamp
);
```

**Parameters**

- `bIsScaleLinestyleRamp`: Input true to set the flag.

**Returns**

Returns `Acad::eOk` if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the scale-ramp-to-fit flag.

```cpp
virtual Acad::ErrorStatus SetScaleRampToFit(
    bool bIsScaleRampToFit
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bIsScaleRampToFit</td>
<td>Input true to set the flag.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the data source of the thematic stylization to be numeric.

```cpp
virtual Acad::ErrorStatus SetSourceDataNumeric(
    bool bSourceDataIsNumeric
);
```

Parameters

- **bSourceDataIsNumeric**: Input true to make the data source numeric.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the use of thousands separators.

```cpp
virtual Acad::ErrorStatus SetUseThousandsSeparators(
    bool bUseThousandsSeparators
);
```

**Parameters**

- `bUseThousandsSeparators`: Input true to use thousands separators.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Invoked from within erase() before the erase actually occurs. See also subErase() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus subErase(
    Adesk::Boolean erasing
);
```

**Parameters**

- **erasing**: Passed-in copy of the erasing argument that was passed to the erase() function call that triggered this subErase() call.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Builds stylization information for later regeneration.

```cpp
virtual Acad::ErrorStatus UnApply(
    void*& pCookie,
    AcDbEntity* pEnt,
    Adesk::UInt32 flag = 0
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pCookie</td>
<td>Input memory value used by styles to transfer cached data across time from one stylization method to another. Cookie memory is considered to be opaque for existing styles, and should not be modified by the caller.</td>
</tr>
<tr>
<td>pEnt</td>
<td>Input entity ID.</td>
</tr>
<tr>
<td>flag</td>
<td>Reserved. For internal use only.</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

**Remarks**

The caller can retrieve this style's cookie, stored previously by another stylization function.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Builds stylization information for later regeneration.

```cpp
virtual Acad::ErrorStatus Update(
    void*& pCookie,
    const AcDbObjectId id,
    Adesk::UInt32 flag = 0
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pCookie</td>
<td>Output memory value used by styles to transfer cached data across time from one stylization method to another. Cookie memory is considered to be opaque for existing styles, and should not be modified by the caller.</td>
</tr>
<tr>
<td>id</td>
<td>Input object ID.</td>
</tr>
<tr>
<td>flag</td>
<td>Reserved. For internal use only.</td>
</tr>
</tbody>
</table>

**Returns**

Returns `Acad::eOk` if successful; otherwise, returns a different error code.

**Remarks**

Derived classes can perform any one-time `Update()`-time actions here. At `Dismiss()` time, the implementation can do any required work to clean up storage created at `Update()` time.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Determines whether thousands separators are used.

```cpp
virtual bool UseThousandsSeparators() const;
```

Returns true if thousand separators are used.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicStyle Class, AcMapDMThematicStyle Class
AcMapDMThematicStyle:: ValuesToIgnore Method
AcMapDMThematicStyle Class | AcMapDMThematicStyle Class

Retrieves the values-to-ignore expression.

virtual const ACHAR * ValuesToIgnore() const;

Returns

Returns the expression, or NULL if the expression is not set.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

`virtual ~AcMapDMThematicTable();`

Returns

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Constructs an instance of this class.

AcMapDMThematicTable();

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTable Class, AcMapDMThematicTable Class
AcMapDMThematicTable:: AppendRow Method
AcMapDMThematicTable Class | AcMapDMThematicTable Class

Appends a specified row to the thematic table.

```cpp
virtual Acad::ErrorStatus AppendRow(
    AcMapDMThematicTableRow& row
);
```

Parameters

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>row</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Empties all contents of the thematic table.

```cpp
virtual Acad::ErrorStatus Clear();
```

Returns `Acad::eOk` if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapDMThematicTable Class, AcMapDMThematicTable Class
AcMapDMThematicTable:: dwgInFields Method
AcMapDMThematicTable Class | AcMapDMThematicTable Class

Lets this object read its data. See also dwgInFields() in the AutoCAD
ObjectARX Developer's Guide.

**virtual** Acad::ErrorStatus dwgInFields(
  AcDbDwgFiler* pFiler
);

Parameters    Description
pFiler    Input filer to use to read the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of **Doc-O-Matic**. In order to make this
message disappear you need to register this software. If you have problems
registering this software please contact us at support@toolsfactory.com.
Links

AcMapDMThematicTable Class, AcMapDMThematicTable Class
AcMapDMThematicTable:: dwgOutFields Method
AcMapDMThematicTable Class | AcMapDMThematicTable Class

Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide.

```cpp
virtual Acad::ErrorStatus dwgOutFields(
    AcDbDwgFiler* pFiler
) const;
```

Parameters Description
pFiler Input filer to use to write the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTable Class, AcMapDMThematicTable Class
AcMapDMThematicTable:: dxfInFields Method
AcMapDMThematicTable Class | AcMapDMThematicTable Class

Lets this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide.

```cpp
virtual Acad::ErrorStatus dxfInFields(
    AcDbDxfFiler* pFiler
);
```

Parameters

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pFiler</td>
</tr>
<tr>
<td>Input filer to use to read the object's data.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTable Class, AcMapDMThematicTable Class
AcMapDMThematicTable:: dxfOutFields Method
AcMapDMThematicTable Class | AcMapDMThematicTable Class

Lets this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide.

```cpp
virtual Acad::ErrorStatus dxfOutFields(
    AcDbDxfFiler* pFiler
) const;
```

Parameters
pFiler Input filer to use to write in the object's data.

Returns
Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Opens and retrieves an annotation cell item.

`virtual Acad::ErrorStatus GetCellAt(
    AcMapDMThematicTableItemAnnotation* & cellItem,
    Adesk::UInt32 row,
    AcDb::OpenMode openMode
);

Parameters

- **cellItem**: Output `AcMapDMThematicTableItemAnnotation` value.
- **row**: Input row number.
- **openMode**: Input open mode.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Opens and retrieves a block cell item.

```cpp
virtual Acad::ErrorStatus GetCellAt(
    AcMapDMThematicTableItemBlock*& cellItem,
    Adesk::UInt32 row,
    AcDb::OpenMode openMode
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cellItem</td>
<td>Output AcMapDMThematicTableItemBlock value.</td>
</tr>
<tr>
<td>row</td>
<td>Input row number.</td>
</tr>
<tr>
<td>openMode</td>
<td>Input open mode.</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Opens and retrieves a color cell item.

```cpp
virtual Acad::ErrorStatus GetCellAt(
    AcMapDMThematicTableItemColor* &cellItem,
    Adesk::UInt32 row,
    AcDb::OpenMode openMode
);
```

### Parameters

- **cellItem**
  - **Description**: Output `AcMapDMThematicTableItemColor` value.
- **row**
  - **Description**: Input row number.
- **openMode**
  - **Description**: Input open mode.

### Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Opens and retrieves a data cell item.

```cpp
virtual Acad::ErrorStatus GetCellAt(
    AcMapDMThematicTableItemDataValue* &cellItem,
    Adesk::UInt32 row,
    AcDb::OpenMode openMode
);
```

Parameters

- `cellItem` Output `AcMapDMThematicTableItemDataValue` value.
- `row` Input row number.
- `openMode` Input open mode.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Opens and retrieves a hatch cell item.

```cpp
virtual Acad::ErrorStatus GetCellAt(
    AcMapDMThematicTableItemHatch* & cellItem,
    Adesk::UInt32 row,
    AcDb::OpenMode openMode
);
```

**Parameters**

- **cellItem**: Output `AcMapDMThematicTableItemHatch` value.
- **row**: Input row number.
- **openMode**: Input open mode.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Opens and retrieves a legend-text cell item.

```cpp
virtual Acad::ErrorStatus GetCellAt(
    AcMapDMThematicTableItemLegendText* & cellItem,
    Adesk::UInt32 row,
    AcDb::OpenMode openMode
);
```

Parameters

- **cellItem**: Output
  - `AcMapDMThematicTableItemLegendText` value.
- **row**: Input row number.
- **openMode**: Input open mode.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Opens and retrieves a linestyle cell item.

```c++
virtual Acad::ErrorStatus GetCellAt(
    AcMapDMThematicTableItemLinestyle*& cellItem,
    Adesk::UInt32 row,
    AcDb::OpenMode openMode
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cellItem</td>
<td>Output AcMapDMThematicTableItemLinestyle value.</td>
</tr>
<tr>
<td>row</td>
<td>Input row number.</td>
</tr>
<tr>
<td>openMode</td>
<td>Input open mode.</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Opens and retrieves a linetype cell item.

```cpp
virtual Acad::ErrorStatus GetCellAt(
    AcMapDMThematicTableItemLinetype* &cellItem,
    Adesk::UInt32 row,
    AcDb::OpenMode openMode
);
```

**Parameters**

- `cellItem`  Output `AcMapDMThematicTableItemLinetype` value.
- `row`  Input row number.
- `openMode`  Input open mode.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Opens and retrieves a lineweight cell item.

```cpp
virtual Acad::ErrorStatus GetCellAt(
    AcMapDMThematicTableItemLineweight*& cellItem,
    Adesk::UInt32 row,
    AcDb::OpenMode openMode
);
```

**Parameters**

- **cellItem**
  - Output
  - `AcMapDMThematicTableItemLineweight` value.

- **row**
  - Input row number.

- **openMode**
  - Input open mode.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Opens and retrieves a plotstyle cell item.

**virtual** Acad::ErrorStatus GetCellAt(
    AcMapDMThematicTableItemPlotstyle* & cellItem,
    Adesk::UInt32 row,
    AcDb::OpenMode openMode
);

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cellItem</td>
<td>Output <strong>AcMapDMThematicTableItemPlotstyle</strong> value.</td>
</tr>
<tr>
<td>row</td>
<td>Input row number.</td>
</tr>
<tr>
<td>openMode</td>
<td>Input open mode.</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at **support@toolsfactory.com**.
Opens and retrieves a text cell item.

```cpp
virtual Acad::ErrorStatus GetCellAt(
    AcMapDMThematicTableItemText*& cellItem,
    Adesk::UInt32 row,
    AcDb::OpenMode openMode
);
```

Parameters

- **cellItem**
  - Output `AcMapDMThematicTableItemText` value.
- **row**
  - Input row number.
- **openMode**
  - Input open mode.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Opens and retrieves an entire row.

```cpp
virtual Acad::ErrorStatus GetRowAt(
    AcMapDMThematicTableRow& row,
    Adesk::Uint32 rowIndex,
    AcDb::OpenMode openMode
);
```

**Parameters**

- **row**
  - Output `AcMapDMThematicTableRow` object.

- **rowIndex**
  - Input row index.

- **openMode**
  - Input open mode.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns the implementation object.

```
virtual AcMapDMImpThematicTable* Implementation() const;
```

Returns the implementation object, or NULL if unsuccessful.

Remarks

This function is for internal use only; do not call it directly.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the annotation style for a specified cell.

```cpp
virtual Acad::ErrorStatus SetCellAt(
    Adesk:: UInt32 row,
    AcMapDMThematicTableItemAnnotation & cellItem)
);
```

**Parameters**

- `row` Input row number.
- `cellItem` Input `AcMapDMThematicTableItemAnnotation` value.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the block for a specified cell.

```cpp
virtual Acad::ErrorStatus SetCellAt(
    Adesk::UInt32 row,
    AcMapDMThematicTableItemBlock& cellItem
);
```

Parameters  | Description
-------------|-------------
row   | Input row number.
cellItem | Input `AcMapDMThematicTableItemBlock` value.

Returns

Returns `Acad::eOk` if successful; otherwise, returns a different error code.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the item color for a specified cell.

```cpp
virtual Acad::ErrorStatus SetCellAt(
    Adesk::UInt32 row,
    AcMapDMThematicTableItemColor& cellItem
);
```

**Parameters**
- **row**
  - Input row number.
- **cellItem**
  - Input `AcMapDMThematicTableItemColor` value.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the value for a specified cell.

```cpp
virtual Acad::ErrorStatus SetCellAt(
    Adesk::UInt32 row,
    AcMapDMThematicTableItemDataValue& cellItem
);
```

**Parameters**

- `row` : Input row number.
- `cellItem` : Input `AcMapDMThematicTableItemDataValue` value.

**Returns**

Returns `Acad::eOk` if successful; otherwise, returns a different error code.

**Remarks**

This function fails if `cellItem` already is appended to the database.

Created with a commercial version of [Doc-O-Matic](https://example.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the hatch for a specified cell.

```cpp
virtual Acad::ErrorStatus SetCellAt(
    Adesk::UInt32 row,
    AcMapDMThematicTableItemHatch& cellItem
);
```

**Parameters**

- `row` Input row number.
- `cellItem` Input `AcMapDMThematicTableItemHatch` value.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com/). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the legend text for a specified cell.

```cpp
virtual Acad::ErrorStatus SetCellAt(
    Adesk::UInt32 row,
    AcMapDMThematicTableItemLegendText& cellItem
);
```

**Parameters**

- **row**
  - Description: Input row number.

- **cellItem**
  - Description: Input `AcMapDMThematicTableItemLegendText` value.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the linestyle for a specified cell.

```cpp
virtual Acad::ErrorStatus SetCellAt(
    Adesk::UInt32 row,
    AcMapDMThematicTableItemLinestyle& cellItem
);
```

Parameters

- **row**: Input row number.
- **cellItem**: Input `AcMapDMThematicTableItemLinestyle` value.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the linetype for a specified cell.

```cpp
virtual Acad::ErrorStatus SetCellAt(
    Adesk::UInt32 row,
    AcMapDMThematicTableItemLinetype& cellItem)
);
```

Parameters

- **row**
  - Input row number.

- **cellItem**
  - Input `AcMapDMThematicTableItemLinetype` value.

Returns

Returns `Acad::eOk` if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.doctomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the lineweight for a specified cell.

```cpp
virtual Acad::ErrorStatus SetCellAt(
    Adesk::UInt32 row,
    AcMapDMThematicTableItemLineweight& cellItem
);
```

Parameters

- **row**: Input row number.
- **cellItem**: Input `AcMapDMThematicTableItemLineweight` value.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the plotstyle for a specified cell.

```cpp
virtual Acad::ErrorStatus SetCellAt(
    Adesk::UInt32 row,
    AcMapDMThematicTableItemPlotstyle& cellItem
);
```

Parameters

- **row**
  - Input row number.
- **cellItem**
  - Input `AcMapDMThematicTableItemPlotstyle` value.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the text for a specified cell.

```cpp
virtual Acad::ErrorStatus SetCellAt(
   Adesk::UInt32 row,
   AcMapDMThematicTableItemText& cellItem)
```

Parameters

- `row`: Input row number.
- `cellItem`: Input `AcMapDMThematicTableItemText` value.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the row at a specified position.

```cpp
virtual Acad::ErrorStatus SetRowAt(
    Adesk::UInt32 rowIndex,
    AcMapDMThematicTableRow& row
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rowIndex</td>
<td>Input row position.</td>
</tr>
<tr>
<td>row</td>
<td>Input <code>AcMapDMThematicTableRow</code> object.</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

```cpp
virtual ~AcMapDMThematicTableItemAnnotation();
```

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.org). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapDMThematicTableItemAnnotation Class,
AcMapDMThematicTableItemAnnotation Class
AcMapDMThematicTableItemAnnotation::
AcMapDMThematicTableItemAnnotation Constructor
AcMapDMThematicTableItemAnnotation Class |
AcMapDMThematicTableItemAnnotation Class

Constructs an instance of this class.

AcMapDMThematicTableItemAnnotation();
Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Constructs an instance of this class by using another instance of this class.

```
AcMapDMThematicTableItemAnnotation(
    const AcMapDMThematicTableItemAnnotation& other
);
```

Parameters
---
other
Input other instance of this class.

Returns
---

Returns nothing.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Disables the color override.

**virtual** Acad::ErrorStatus clearColorOverride();

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links

AcMapDMThematicTableItemAnnotation Class, AcMapDMThematicTableItemAnnotation Class
AcMapDMThematicTableItemAnnotation:: ClearLayerOverride Method
AcMapDMThematicTableItemAnnotation Class | AcMapDMThematicTableItemAnnotation Class

Disables the layer override.

**virtual** Acad::ErrorStatus ClearLayerOverride();

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links

AcMapDMThematicTableItemAnnotation Class,
AcMapDMThematicTableItemAnnotation Class
AcMapDMThematicTableItemAnnotation:: ClearLinetypeOverride Method
AcMapDMThematicTableItemAnnotation Class |
AcMapDMThematicTableItemAnnotation Class

Disables the linetype override.

**virtual** Acad::ErrorStatus ClearLinetypeOverride();

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Disables the lineweight override.

**virtual** Acad::ErrorStatus ClearLineweightOverride();

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Disables the rotation override.

```cpp
virtual Acad::ErrorStatus ClearRotationOverride();
```

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableItemAnnotation Class, AcMapDMThematicTableItemAnnotation Class
AcMapDMThematicTableItemAnnotation:: ClearScaleOverride Method
AcMapDMThematicTableItemAnnotation Class | AcMapDMThematicTableItemAnnotation Class

Disables the scale override.

**virtual** Acad::ErrorStatus ClearScaleOverride();

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links

AcMapDMThematicTableItemAnnotation Class,
AcMapDMThematicTableItemAnnotation Class
AcMapDMThematicTableItemAnnotation:: ColorOverride Method
AcMapDMThematicTableItemAnnotation Class |
AcMapDMThematicTableItemAnnotation Class

Retrieves the color override.

```cpp
virtual AcCmColor ColorOverride() const;
```

Returns the color override if successful, or AcDbObjectId::kNull otherwise.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the color override expression.

```cpp
virtual const ACHAR* ColorOverrideExpression() const;
```

Returns the color override expression if successful, or NULL otherwise.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableItemAnnotation Class,
AcMapDMThematicTableItemAnnotation Class
AcMapDMThematicTableItemAnnotation:: dwgInFields Method
AcMapDMThematicTableItemAnnotation Class |
AcMapDMThematicTableItemAnnotation Class

Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide.

**virtual** Acad::ErrorStatus dwgInFields(  
   AcDbDwgFiler* pFiler
);

Parameters  Description
pFiler  Input filer to use to read the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableItemAnnotation Class,
AcMapDMThematicTableItemAnnotation Class
AcMapDMThematicTableItemAnnotation:: dwgOutFields Method
AcMapDMThematicTableItemAnnotation Class |
AcMapDMThematicTableItemAnnotation Class

Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide.

**virtual** Acad::ErrorStatus dwgOutFields(
    AcDbDwgFiler* pFiler
) **const**;

Parameters | Description
--- | ---
pFiler | Input filer to use to write the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links

AcMapDMThematicTableItemAnnotation Class,
AcMapDMThematicTableItemAnnotation Class
AcMapDMThematicTableItemAnnotation:: dxfInFields Method
AcMapDMThematicTableItemAnnotation Class |
AcMapDMThematicTableItemAnnotation Class

Lets this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide.

```cpp
virtual Acad::ErrorStatus dxfInFields(
    AcDbDxfFiler* pFiler
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pFiler</td>
<td>Input filer to use to read the object's data.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Let's this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide.

```cpp
virtual Acad::ErrorStatus dxfOutFields(
    AcDbDxfFiler* pFiler
) const;
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pFiler</td>
<td>Input filer to use to write in the object's data.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns the implementation object.

```cpp
virtual AcMapDMImpThematicTableItemAnnotation* Implementation() const
```

Returns the implementation object, or NULL if unsuccessful.

Remarks

This function is for internal use only; do not call it directly.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableItemAnnotation Class,
AcMapDMThematicTableItemAnnotation Class
AcMapDMThematicTableItemAnnotation:: IsColorOverrideEnabled Method
AcMapDMThematicTableItemAnnotation Class |
AcMapDMThematicTableItemAnnotation Class

Indicates the enable state of the color override.

`virtual bool IsColorOverrideEnabled() const;`

Returns

Returns true if enabled, false if disabled.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Indicates the enable state of the layer override.

```cpp
virtual bool IsLayerOverrideEnabled() const;
```

Returns true if enabled, false if disabled.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Indicates the enable state of the linetype override.

```cpp
virtual bool IsLinetypeOverrideEnabled() const;
```

Returns true if enabled, false if disabled.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableItemAnnotation Class,
AcMapDMThematicTableItemAnnotation Class
AcMapDMThematicTableItemAnnotation::
IsLineweightOverrideEnabled Method
AcMapDMThematicTableItemAnnotation Class |
AcMapDMThematicTableItemAnnotation Class

Indicates the enable state of the lineweight override.

virtual bool IsLineweightOverrideEnabled() const;

Returns

Returns true if enabled, false if disabled.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableItemAnnotation Class, AcMapDMThematicTableItemAnnotation Class
AcMapDMThematicTableItemAnnotation:: IsRotationOverrideEnabled Method
AcMapDMThematicTableItemAnnotation Class | AcMapDMThematicTableItemAnnotation Class

Indicates the enable state of the rotation override.

virtual bool IsRotationOverrideEnabled() const;

Returns

Returns true if enabled, false if disabled.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Indicates the enable state of the scale override.

```cpp
virtual bool IsScaleOverrideEnabled() const;
```

Returns true if enabled, false if disabled.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the layer override.

```cpp
virtual AcDbObjectId LayerOverride() const;
```

Returns the layer override if successful, or AcDbObjectId::kNull otherwise.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the layer override expression.

```cpp
virtual const ACHAR* LayerOverrideExpression() const;
```

Returns the layer override expression if successful, or NULL otherwise.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the linetype override.

```cpp
virtual AcDbObjectId LinetypeOverride() const;
```

Returns the linetype override if successful, or `AcDbObjectId::kNull` otherwise.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the linetype override expression.

```cpp
virtual const ACHAR* LinetypeOverrideExpression() const;
```

Returns the linetype override expression if successful, or NULL otherwise.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableItemAnnotation Class,
AcMapDMThematicTableItemAnnotation Class
AcMapDMThematicTableItemAnnotation:: LineweightOverride Method
AcMapDMThematicTableItemAnnotation Class |
AcMapDMThematicTableItemAnnotation Class

Retrieves the lineweight override.

virtual AcDb::LineWeight LineweightOverride() const;

Returns

Returns the lineweight override.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieve the lineweight override expression.

```cpp
virtual const ACHAR* LineweightOverrideExpression() const;
```

Returns the lineweight override expression if successful, or NULL otherwise.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Retrieves the annotation location (insertion point) override expression.

```
virtual const ACHAR * LocationOverrideExpression() const;
```

Returns the annotation location override expression if successful, or NULL otherwise.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the rotation override.

```
virtual double RotationOverride() const;
```

Returns the rotation override.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableItemAnnotation Class,
AcMapDMThematicTableItemAnnotation Class
AcMapDMThematicTableItemAnnotation::
RotationOverrideExpression Method
AcMapDMThematicTableItemAnnotation Class |
AcMapDMThematicTableItemAnnotation Class

Retrieves the annotation rotation override expression.

virtual const ACHAR * RotationOverrideExpression() const;

Returns

Returns the annotation rotation override expression if successful, or NULL otherwise.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableItemAnnotation Class,
AcMapDMThematicTableItemAnnotation Class
AcMapDMThematicTableItemAnnotation:: ScaleOverride Method
AcMapDMThematicTableItemAnnotation Class |
AcMapDMThematicTableItemAnnotation Class

Retrieves the scale override.

**virtual double** ScaleOverride() **const**;

Returns

Returns the scale override.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableItemAnnotation Class, AcMapDMThematicTableItemAnnotation Class
AcMapDMThematicTableItemAnnotation:: ScaleOverrideExpression Method
AcMapDMThematicTableItemAnnotation Class | AcMapDMThematicTableItemAnnotation Class

Retrieves the annotation scale override expression.

**virtual const** ACHAR * ScaleOverrideExpression() **const**;

Returns

Returns the annotation scale override expression if successful, or NULL otherwise.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the color override value.

```
virtual Acad::ErrorStatus SetColorOverride(
    const AcCmColor colorOverride
);
```

Parameters                  Description
----------------------------------------------
colorOverride               Input color override value.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the layer override value.

```cpp
virtual Acad::ErrorStatus SetLayerOverride(
    const AcDbObjectId idLayerOverride
);
```

Parameters | Description
---|---
idLayerOverride | Input layer override AcDbObjectId.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the linetype override value.

```cpp
def virtual Acad::ErrorStatus SetLinetypeOverride(
    const AcDbObjectId idLinetypeOverride);
```

**Parameters**
- `idLinetypeOverride`: Input linetype override `AcDbObjectId`.

**Returns**
- Returns `Acad::eOk` if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.doc-o-matic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the lineweight override value.

`virtual Acad::ErrorStatus SetLineweightOverride(
    AcDb::LineWeight lineweightOverride
);`

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>lineweightOverride</td>
<td>Input lineweight.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the rotation override value.

```cpp
virtual Acad::ErrorStatus SetRotationOverride(
    double rotationOverride
);
```

**Parameters**

- `rotationOverride`: Input rotation override value.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the scale override value.

```cpp
virtual Acad::ErrorStatus SetScaleOverride(
    double scaleOverride
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>scaleOverride</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the template name for this annotation alteration.

```cpp
virtual Acad::ErrorStatus SetTemplateId(
   AcDbObjectId templateId)

Parameters
  templateId
    Description
    Input object ID of the template for this annotation alteration.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the annotation template Id for this alteration.

```cpp
virtual AcDbObjectId TemplateId() const;
```

Returns the object ID of the annotation template if successful, or NULL otherwise.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableItemBlock Class
AcMapDMThematicTableItemBlock Class

AcMapDMThematicTableItemBlock::~AcMapDMThematicTableItemBlock
Destructor
AcMapDMThematicTableItemBlock Class |
AcMapDMThematicTableItemBlock Class

Destroys an instance of this class.

virtual ~AcMapDMThematicTableItemBlock();
Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Constructs an instance of this class.

AcMapDMThematicTableItemBlock();
Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableItemBlock Class,
AcMapDMThematicTableItemBlock Class
AcMapDMThematicTableItemBlock::
AcMapDMThematicTableItemBlock Constructor
AcMapDMThematicTableItemBlock Class |
AcMapDMThematicTableItemBlock Class

Constructs an instance of this class by using another instance of this class.

AcMapDMThematicTableItemBlock(
    const AcMapDMThematicTableItemBlock& other
);

Parameters  Description
other        Input other instance of this class.

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
AcMapDMThematicTableItemBlock Class,
AcMapDMThematicTableItemBlock Class
AcMapDMThematicTableItemBlock:: Angle Method
AcMapDMThematicTableItemBlock Class
AcMapDMThematicTableItemBlock Class

Retrieves the angle.

**virtual int** Angle() const;

Returns

Returns the angle.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at **support@toolsfactory.com**.
Retrieves the block AcDbObjectId.

\[
\text{AcDbObjectId BlockId()} \text{ const;}
\]

Returns the block AcDbObjectId if successful, or AcDbObjectId::kNull otherwise.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links

AcMapDMThematicTableItemBlock Class,
AcMapDMThematicTableItemBlock Class

AcMapDMThematicTableItemBlock:: BlockName Method

AcMapDMThematicTableItemBlock Class |
AcMapDMThematicTableItemBlock Class

Retrieves the block name.

**virtual const** ACHAR* BlockName() **const**;

Returns

Returns the block name if successful, or NULL otherwise.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links

AcMapDMThematicTableItemBlock Class,
AcMapDMThematicTableItemBlock Class
AcMapDMThematicTableItemBlock:: dwgInFields Method
AcMapDMThematicTableItemBlock Class |
AcMapDMThematicTableItemBlock Class

Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide.

virtual Acad::ErrorStatus dwgInFields(
    AcDbDwgFiler* pFiler
);

Parameters Description
pFiler Input filer to use to read the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableItemBlock Class,
AcMapDMThematicTableItemBlock Class
AcMapDMThematicTableItemBlock:: dwgOutFields Method
AcMapDMThematicTableItemBlock Class |
AcMapDMThematicTableItemBlock Class

Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide.

**virtual Acad::ErrorStatus dwgOutFields(  
    AcDbDwgFiler* pFiler  
) const;**

Parameters  
Description
pFiler  
Input filer to use to write the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableItemBlock Class,
AcMapDMThematicTableItemBlock Class
AcMapDMThematicTableItemBlock:: dxfInFields Method
AcMapDMThematicTableItemBlock Class |
AcMapDMThematicTableItemBlock Class

Lets this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide.

```cpp
virtual Acad::ErrorStatus dxfInFields(
    AcDbDxfFiler* pFiler
);
```

Parameters

- **pFiler**
  - Description: Input filer to use to read the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Let this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide.

```cpp
virtual Acad::ErrorStatus dxfOutFields(
    AcDbDxfFiler* pFiler
) const;
```

**Parameters**

- **pFiler**
  - Description: Input filer to use to write in the object's data.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns the implementation object.

```cpp
virtual AcMapDMImpThematicTableItemBlock* Implementation() const;
```

Returns the implementation object, or NULL if unsuccessful.

Remarks

This function is for internal use only; do not call it directly.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links

AcMapDMThematicTableItemBlock Class,
AcMapDMThematicTableItemBlock Class
AcMapDMThematicTableItemBlock:: LayerId Method
AcMapDMThematicTableItemBlock Class |
AcMapDMThematicTableItemBlock Class

Retrieves the layer AcDbObjectId.

AcDbObjectId LayerId() const;
Returns

Returns the layer AcDbObjectId if successful, or AcDbObjectId::kNull otherwise.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the layer name.

```cpp
virtual const ACHAR* LayerName() const;
```

Returns the layer name if successful, or NULL otherwise.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableItemBlock Class,
AcMapDMThematicTableItemBlock Class
AcMapDMThematicTableItemBlock:: Scale Method
AcMapDMThematicTableItemBlock Class |
AcMapDMThematicTableItemBlock Class

Retrieves the scale.

`virtual double Scale() const;`

Returns

Returns scale.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the angle.

```
virtual Acad::ErrorStatus SetAngle(
    int angle
);
```

**Parameters**  
angle  
Input angle.

**Returns**  
Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the block by AcDbObjectId.

```
Acad::ErrorStatus SetBlockId(
    const AcDbObjectId idBlock
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>idBlock</td>
<td>Input block AcDbObjectId.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the layer by AcDbObjectId.

```
Acad::ErrorStatus SetLayerId(
    const AcDbObjectId idLayer);
```

Parameters
- Description
  - `idLayer`: Input layer AcDbObjectId.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the scale.

\begin{verbatim}
virtual Acad::ErrorStatus SetScale( 
   double scale 
);
\end{verbatim}

Parameters Description
scale Input scale.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableItemColor Class, AcMapDMThematicTableItemColor Class
AcMapDMThematicTableItemColor:: ~AcMapDMThematicTableItemColor Destructor
AcMapDMThematicTableItemColor Class | AcMapDMThematicTableItemColor Class

Destroys an instance of this class.

```cpp
virtual ~AcMapDMThematicTableItemColor();
```

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Constructs an instance of this class.

AcMapDMThematicTableItemColor();

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Constructs an instance of this class by using another instance of this class.

```cpp
AcMapDMThematicTableItemColor(
    const AcMapDMThematicTableItemColor& other
);
```

Parameters

- `other`: Input other instance of this class.

Returns

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapDMThematicTableItemColor Class, AcMapDMThematicTableItemColor Class
AcMapDMThematicTableItemColor:: Color Method
AcMapDMThematicTableItemColor Class | AcMapDMThematicTableItemColor Class

Retrieves the color.

virtual AcCmColor Color() const;

Returns

Returns the color.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableItemColor Class, AcMapDMThematicTableItemColor Class
AcMapDMThematicTableItemColor:: dwgInFields Method
AcMapDMThematicTableItemColor Class | AcMapDMThematicTableItemColor Class

Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide.

```cpp
virtual Acad::ErrorStatus dwgInFields(
    AcDbDwgFiler* pFiler
);
```

Parameters
pFiler Input filer to use to read the object's data.

Returns
Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableItemColor Class, AcMapDMThematicTableItemColor Class
AcMapDMThematicTableItemColor::dwgOutFields Method
AcMapDMThematicTableItemColor Class | AcMapDMThematicTableItemColor Class

Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide.

virtual Acad::ErrorStatus dwgOutFields(
    AcDbDwgFiler* pFiler
) const;

Parameters

pFiler

Description

Input filer to use to write the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links

AcMapDMThematicTableItemColor Class, AcMapDMThematicTableItemColor Class
AcMapDMThematicTableItemColor:: dxfInFields Method
AcMapDMThematicTableItemColor Class | AcMapDMThematicTableItemColor Class

Lets this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide.

```cpp
virtual Acad::ErrorStatus dxfInFields(
    AcDbDxfFiler* pFiler
);
```

Parameters  Description
pFiler       Input filer to use to read the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableItemColor Class, AcMapDMThematicTableItemColor Class
AcMapDMThematicTableItemColor:: dxfOutFields Method
AcMapDMThematicTableItemColor Class | AcMapDMThematicTableItemColor Class

Lets this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide.

```cpp
virtual Acad::ErrorStatus dxfOutFields(
    AcDbDxfFiler* pFiler
) const;
```

Parameters
- **pFiler**
  - Input filer to use to write in the object's data.

Returns
- Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableItemColor Class, AcMapDMThematicTableItemColor Class

AcMapDMThematicTableItemColor:: Implementation Method
AcMapDMThematicTableItemColor Class | AcMapDMThematicTableItemColor Class

Returns the implementation object.

**virtual** AcMapDMImpThematicTableItemColor* Implementation() const;

Returns

Returns the implementation object, or NULL if unsuccessful.

Remarks

This function is for internal use only; do not call it directly.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the color.

```cpp
virtual Acad::ErrorStatus SetColor(
    const AcCmColor pColor);
```

**Parameters**
- `pColor` Input color.

**Returns**
Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

```cpp
virtual ~AcMapDMThematicTableItemDataValue();
```

Returns

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Constructs an instance of this class.

`AcMapDMThematicTableItemDataValue();`

Returns

Returns nothing.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Constructs an instance of this class by using another instance of this class.

```cpp
AcMapDMThematicTableItemDataValue(
    const AcMapDMThematicTableItemDataValue& other);
```

Parameters

- **other**
  - Input other instance of this class.

Returns

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links

AcMapDMThematicTableItemDataValue Class,
AcMapDMThematicTableItemDataValue Class
AcMapDMThematicTableItemDataValue:: dwgInFields Method
AcMapDMThematicTableItemDataValue Class |
AcMapDMThematicTableItemDataValue Class

Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide.

virtual Acad::ErrorStatus dwgInFields(
    AcDbDwgFiler* pFiler
);

Parameters Description
pFiler Input filer to use to read the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableItemDataValue Class,
AcMapDMThematicTableItemDataValue Class

AcMapDMThematicTableItemDataValue:: dwgOutFields Method
AcMapDMThematicTableItemDataValue Class |  
AcMapDMThematicTableItemDataValue Class

Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide.

```cpp
virtual Acad::ErrorStatus dwgOutFields(
    AcDbDwgFiler* pFiler
) const;
```

Parameters

- **pFiler**
  - Description: Input filer to use to write the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableItemDataValue Class,
AcMapDMThematicTableItemDataValue Class
AcMapDMThematicTableItemDataValue::dxfInFields Method
AcMapDMThematicTableItemDataValue Class | AcMapDMThematicTableItemDataValue Class

Lets this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide.

virtual Acad::ErrorStatus dxfInFields(
    AcDbDxfFiler* pFiler
);

Parameters
pFiler Input filer to use to read the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links

AcMapDMThematicTableItemDataValue Class, 
AcMapDMThematicTableItemDataValue Class

AcMapDMThematicTableItemDataValue:: dxfOutFields Method

AcMapDMThematicTableItemDataValue Class | 
AcMapDMThematicTableItemDataValue Class

Lets this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide.

virtual Acad::ErrorStatus dxfOutFields(
    AcDbDxfFiler* pFiler
) const;

Parameters Description

pFiler Input filer to use to write in the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns the implementation object.

```cpp
virtual AcMapDMImpThematicTableItemDataValue* Implementation() const
```

Returns the implementation object, or NULL if unsuccessful.

Remarks

This function is for internal use only; do not call it directly.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableItemDataValue Class,
AcMapDMThematicTableItemDataValue Class
AcMapDMThematicTableItemDataValue:: SetValue Method
AcMapDMThematicTableItemDataValue Class |
AcMapDMThematicTableItemDataValue Class

Sets the value.

```cpp
virtual Acad::ErrorStatus SetValue(
    double value
);
```

Parameters Description
value Input value.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableItemDataValue Class,
AcMapDMThematicTableItemDataValue Class
AcMapDMThematicTableItemDataValue:: Text Method
AcMapDMThematicTableItemDataValue Class | AcMapDMThematicTableItemDataValue Class

Retrieves the text.

`virtual const ACHAR* Text() const;`
Returns

Returns the text if successful, or NULL otherwise.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the value.

```cpp
virtual double Value() const;
```

Returns

Returns the value if successful, or NULL otherwise.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

```cpp
virtual ~AcMapDMThematicTableItemHatch();
```

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Constructs an instance of this class.

AcMapDMThematicTableItemHatch();
Returns
Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Constructs an instance of this class by using another instance of this class.

```c++
AcMapDMThematicTableItemHatch(
    const AcMapDMThematicTableItemHatch& other
);
```

**Parameters**

- `other` Input other instance of this class.

**Returns**

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.doc-omatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links

AcMapDMThematicTableItemHatch Class, AcMapDMThematicTableItemHatch Class
AcMapDMThematicTableItemHatch:: Angle Method
AcMapDMThematicTableItemHatch Class | AcMapDMThematicTableItemHatch Class

Retrieves the angle.

virtual int Angle() const;

Returns

Returns the angle.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableItemHatch Class,
AcMapDMThematicTableItemHatch Class
AcMapDMThematicTableItemHatch:: Color Method
AcMapDMThematicTableItemHatch Class |
AcMapDMThematicTableItemHatch Class

Retrieves the color.

```cpp
virtual AcCmColor Color() const;
```

Returns

Returns the color.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableItemHatch Class,
AcMapDMThematicTableItemHatch Class
AcMapDMThematicTableItemHatch:: dwgInFields Method
AcMapDMThematicTableItemHatch Class |
AcMapDMThematicTableItemHatch Class

Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide.

```cpp
virtual Acad::ErrorStatus dwgInFields(
    AcDbDwgFiler* pFiler
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pFiler</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableItemHatch Class,
AcMapDMThematicTableItemHatch Class
AcMapDMThematicTableItemHatch:: dwgOutFields Method
AcMapDMThematicTableItemHatch Class |
AcMapDMThematicTableItemHatch Class

Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide.

virtual Acad::ErrorStatus dwgOutFields(
  AcDbDwgFiler* pFiler
) const;

Parameters  Description
pFiler  Input filer to use to write the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableItemHatch Class,
AcMapDMThematicTableItemHatch Class
AcMapDMThematicTableItemHatch:: dxfInFields Method
AcMapDMThematicTableItemHatch Class |
AcMapDMThematicTableItemHatch Class

Lets this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide.

**virtual** Acad::ErrorStatus dxfInFields(
    AcDbDxfFiler* pFiler
);

Parameters
---
**pFiler**
Input filer to use to read the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapDMThematicTableItemHatch Class,
AcMapDMThematicTableItemHatch Class
AcMapDMThematicTableItemHatch:: dxfOutFields Method
AcMapDMThematicTableItemHatch Class |
AcMapDMThematicTableItemHatch Class

Lets this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide.

```cpp
virtual Acad::ErrorStatus dxfOutFields(
    AcDbDxfFiler* pFiler
) const;
```

Parameters

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pFiler</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns the implementation object.

```
virtual AcMapDMImpThematicTableItemHatch* Implementation() const;
```

Returns the implementation object, or NULL if unsuccessful.

Remarks

This function is for internal use only; do not call it directly.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableItemHatch Class,
AcMapDMThematicTableItemHatch Class

AcMapDMThematicTableItemHatch:: LayerId Method
AcMapDMThematicTableItemHatch Class | AcMapDMThematicTableItemHatch Class

Retrieves the layer AcDbObjectId.

AcDbObjectId LayerId() const;

Returns

Returns the layer AcDbObjectId if successful, or AcDbObjectId::kNull otherwise.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the layer name.

```cpp
virtual const ACHAR* LayerName() const;
```

Returns the layer name if successful, or NULL otherwise.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableItemHatch Class,
AcMapDMThematicTableItemHatch Class
AcMapDMThematicTableItemHatch:: PatternName Method
AcMapDMThematicTableItemHatch Class |
AcMapDMThematicTableItemHatch Class

Retrieves the pattern name.

```cpp
virtual const ACHAR* PatternName() const;
```

Returns the pattern name if successful. or NULL otherwise.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableItemHatch Class,
AcMapDMThematicTableItemHatch Class
AcMapDMThematicTableItemHatch:: Scale Method
AcMapDMThematicTableItemHatch Class | 
AcMapDMThematicTableItemHatch Class

Retrieves the scale.

**virtual double Scale() const**;

Returns

Returns the scale.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the angle.

```cpp
virtual Acad::ErrorStatus SetAngle(
    int angle
);
```

Parameters

- **angle**
  - Input angle.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.toolsfactory.com/products/docomatic). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the color.

```cpp
virtual Acad::ErrorStatus SetColor(
    const AcCmColor color
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>color</td>
<td>Input color.</td>
</tr>
</tbody>
</table>

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the layer AcDbObjectId.

```
Acad::ErrorStatus SetLayerId(
    AcDbObjectId layerId
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>layerId</td>
</tr>
</tbody>
</table>

**Description**

Input layer AcDbObjectId.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the scale.

```cpp
virtual Acad::ErrorStatus SetScale(
    double scale
);
```

Parameters  

- **scale**: Input scale.

Returns  

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableItemLegendText Class,
AcMapDMThematicTableItemLegendText Class
AcMapDMThematicTableItemLegendText::~:
~AcMapDMThematicTableItemLegendText Destructor
AcMapDMThematicTableItemLegendText Class |
AcMapDMThematicTableItemLegendText Class

Destroys an instance of this class.

virtual ~AcMapDMThematicTableItemLegendText();

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links

AcMapDMThematicTableItemLegendText Class, 
AcMapDMThematicTableItemLegendText Class
AcMapDMThematicTableItemLegendText::
AcMapDMThematicTableItemLegendText Constructor
AcMapDMThematicTableItemLegendText Class | 
AcMapDMThematicTableItemLegendText Class

Constructs an instance of this class.

AcMapDMThematicTableItemLegendText();

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Constructs an instance of this class by using another instance of this class.

```
AcMapDMThematicTableItemLegendText(
    const AcMapDMThematicTableItemLegendText& other
);
```

**Parameters**
- `other`: Input other instance of this class.

**Returns**
- Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableItemLegendText Class,
AcMapDMThematicTableItemLegendText Class
AcMapDMThematicTableItemLegendText:: Color Method
AcMapDMThematicTableItemLegendText Class |
AcMapDMThematicTableItemLegendText Class

Retrieves the color of the cell.

```
virtual AcCmColor Color() const;
```

Returns

Returns the color.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableItemLegendText Class,
AcMapDMThematicTableItemLegendText Class
AcMapDMThematicTableItemLegendText:: dwgInFields Method
AcMapDMThematicTableItemLegendText Class |
AcMapDMThematicTableItemLegendText Class

Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide.

```cpp
virtual Acad::ErrorStatus dwgInFields
(  
   AcDbDwgFiler* pFiler
);
```

Parameters
- **pFiler**
  - Input filer to use to read the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableItemLegendText Class,
AcMapDMThematicTableItemLegendText Class
AcMapDMThematicTableItemLegendText:: dwgOutFields Method
AcMapDMThematicTableItemLegendText Class | 
AcMapDMThematicTableItemLegendText Class

Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide.

```cpp
virtual Acad::ErrorStatus dwgOutFields(AcDbDwgFiler* pFiler)
```  
Parameters Description
pFiler Input filer to use to write the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Let this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide.

```cpp
virtual Acad::ErrorStatus dxfInFields(AcDbDxfFiler* pFiler);
```

**Parameters**

- **pFiler**: Input filer to use to read the object's data.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.doc-o-matic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableItemLegendText Class,
AcMapDMThematicTableItemLegendText Class
AcMapDMThematicTableItemLegendText:: dxfOutFields Method
AcMapDMThematicTableItemLegendText Class |
AcMapDMThematicTableItemLegendText Class

Lets this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide.

```cpp
virtual Acad::ErrorStatus dxfOutFields(
    AcDbDxfFiler* pFiler
) const;
```

Parameters Description
pFiler Input filer to use to write in the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableItemLegendText Class,
AcMapDMThematicTableItemLegendText Class
AcMapDMThematicTableItemLegendText:: Height Method
AcMapDMThematicTableItemLegendText Class | 
AcMapDMThematicTableItemLegendText Class

Retrieves the height of the cell.

virtual double Height() const;

Returns

Returns the height.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns the implementation object.

`virtual AcMapDMImpThematicTableItemLegendText* Implementation() const` Returns

Returns the implementation object, or NULL if unsuccessful.

Remarks

This function is for internal use only; do not call it directly.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableItemLegendText Class,
AcMapDMThematicTableItemLegendText Class
AcMapDMThematicTableItemLegendText:: LayerId Method
AcMapDMThematicTableItemLegendText Class |
AcMapDMThematicTableItemLegendText Class

Retrieves the layer AcDbObjectId of the cell.

AcDbObjectId LayerId() const;
Returns

Returns the layer AcDbObjectId if successful, or AcDbObjectId::kNull otherwise.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableItemLegendText Class,
AcMapDMThematicTableItemLegendText Class
AcMapDMThematicTableItemLegendText:: LayerName Method
AcMapDMThematicTableItemLegendText Class |
AcMapDMThematicTableItemLegendText Class

Retrieves the layer name of the cell.

```cpp
virtual const ACHAR* LayerName() const;
```

Returns

Returns the layer name if successful, or NULL otherwise.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the color of the cell.

```cpp
virtual Acad::ErrorStatus SetColor(
    const AcCmColor color
);
```

**Parameters**
- `color` 
  - **Description**: Input color.

**Returns**
- Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapDMThematicTableItemLegendText Class, AcMapDMThematicTableItemLegendText Class
AcMapDMThematicTableItemLegendText:: SetHeight Method
AcMapDMThematicTableItemLegendText Class | AcMapDMThematicTableItemLegendText Class

Sets the height of the cell.

\texttt{virtual Acad::ErrorStatus SetHeight(}
\hspace{1em} \texttt{double height)}

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>height</td>
<td>Input height.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the layer AcDbObjectId of the cell.

Acad::ErrorStatus SetLayerId(
    const AcDbObjectId layerId
);

Parameters

layerId          Description
Input layer AcDbObjectId.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the style AcDbObjectId of the cell.

```
Acad::ErrorStatus SetStyleId(
    const AcDbObjectId styleId
);
```

**Parameters**
- **styleId**: Input style AcDbObjectId.

**Returns**
- Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the style AcDbObjectId of the cell.

```cpp
AcDbObjectId StyleId() const;
```

Returns the style AcDbObjectId if successful, or AcDbObjectId::kNull otherwise.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the style name of the cell.

```cpp
virtual const ACHAR* StyleName() const;
```

Returns the style name if successful, or NULL otherwise.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the text of the cell.

```cpp
virtual const ACHAR* Text() const;
```

Returns the text if successful, or NULL otherwise.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

```cpp
virtual ~AcMapDMThematicTableItemLinestyle();
```

Returns nothing.

Created with a commercial version of [Doc-O-Matic](#). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Constructs an instance of this class.

```
AcMapDMThematicTableItemLineStyle();
```

Returns

```
Returns nothing.
```

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Constructs an instance of this class by using another instance of this class.

```cpp
AcMapDMThematicTableItemLinestyle(
    const AcMapDMThematicTableItemLinestyle& other
);
```

Parameters

- `other` Input other instance of this class.

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableItemLinestyle Class,
AcMapDMThematicTableItemLinestyle Class
AcMapDMThematicTableItemLinestyle:: dwgInFields Method
AcMapDMThematicTableItemLinestyle Class |
AcMapDMThematicTableItemLinestyle Class

Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide.

virtual Acad::ErrorStatus dwgInFields(
    AcDbDwgFiler* pFiler
);

Parameters	Description
pFiler	Input filer to use to read the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableItemLinestyle Class,
AcMapDMThematicTableItemLinestyle Class
AcMapDMThematicTableItemLinestyle:: dwgOutFields Method
AcMapDMThematicTableItemLinestyle Class |
AcMapDMThematicTableItemLinestyle Class

Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide.

```cpp
virtual Acad::ErrorStatus dwgOutFields(
    AcDbDwgFiler* pFiler
) const;
```

Parameters Description
---
pFiler Input filer to use to write the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Let's this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide.

```c++
virtual Acad::ErrorStatus dxfInFields(
    AcDbDxfFiler* pFiler
);
```

**Parameters**

- **pFiler**
  - Input filer to use to read the object's data.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableItemLinestyle Class, AcMapDMThematicTableItemLinestyle Class
AcMapDMThematicTableItemLinestyle:: dxfOutFields Method
AcMapDMThematicTableItemLinestyle Class | AcMapDMThematicTableItemLinestyle Class

 Lets this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide.

```cpp
virtual Acad::ErrorStatus dxfOutFields(
    AcDbDxfFiler* pFiler
) const;
```

Parameters Description
---
pFiler Input filer to use to write in the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableItemLinestyle Class,
AcMapDMThematicTableItemLinestyle Class
AcMapDMThematicTableItemLinestyle:: Implementation Method
AcMapDMThematicTableItemLinestyle Class |
AcMapDMThematicTableItemLinestyle Class

Returns the implementation object.

virtual AcMapDMImpThematicTableItemLinestyle* Implementation() const
Returns

Returns the implementation object, or NULL if unsuccessful.

Remarks

This function is for internal use only; do not call it directly.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieve the layer `AcDbObjectId`.

```cpp
AcDbObjectId LayerId() const;
```

Returns the layer `AcDbObjectId` if successful, or `AcDbObjectId::kNull` otherwise.

Created with a commercial version of [Doc-O-Matic](https://toolsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the layer name.

```cpp
virtual const ACHAR* LayerName() const;
```

Returns the layer name if successful, or NULL otherwise.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableItemLinestyle Class,
AcMapDMThematicTableItemLinestyle Class
AcMapDMThematicTableItemLinestyle:: LinetypeId Method
AcMapDMThematicTableItemLinestyle Class | 
AcMapDMThematicTableItemLinestyle Class

Retrieves the linetype AcDbObjectId.

AcDbObjectId LinetypeId() const;
Returns

Returns the AcDbObjectId name if successful, or AcDbObjectId::kNull otherwise.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieve the linetype name.

```cpp
definition

virtual const ACHAR* LinetypeName() const;
```

Returns the linetype name if successful, or NULL otherwise.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the layer AcDbObjectId.

```cpp
Acad::ErrorStatus SetLayerId(
    const AcDbObjectId idLayer
);
```

Parameters  Description
idLayer      Input layer name.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the linetype AcDbObjectId.

```cpp
Acad::ErrorStatus SetLinetypeId(
    const AcDbObjectId idLinetype
);
```

Parameters | Description
--- | ---
idLinetype | Input linetype AcDbObjectId.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the width.

```cpp
virtual Acad::ErrorStatus SetWidth(
    double width
);
```

Parameters

- **width**: Input width.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Retrieves the width.

```cpp
virtual double Width() const;
```

Returns the width.

Created with a commercial version of `Doc-O-Matic`. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at `support@toolsfactory.com`.
Destroys an instance of this class.

```cpp
virtual ~AcMapDMThematicTableItemLinetype();
```

Returns nothing.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapDMThematicTableItemLinetype Class, AcMapDMThematicTableItemLinetype Class
AcMapDMThematicTableItemLinetype::
AcMapDMThematicTableItemLinetype Constructor
AcMapDMThematicTableItemLinetype Class | AcMapDMThematicTableItemLinetype Class

Constructs an instance of this class.

AcMapDMThematicTableItemLinetype();
Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Constructs an instance of this class by using another instance of this class.

`AcMapDMThematicTableItemLinetype(const AcMapDMThematicTableItemLinetype& other);`

Parameters
- `other`: Input other instance of this class.

Returns
- Returns nothing.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links  
AcMapDMThematicTableItemLinetype Class,  
AcMapDMThematicTableItemLinetype Class

AcMapDMThematicTableItemLinetype:: dwgInFields Method  
AcMapDMThematicTableItemLinetype Class |  
AcMapDMThematicTableItemLinetype Class

Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide.

```cpp
virtual Acad::ErrorStatus dwgInFields(
    AcDbDwgFiler* pFiler
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pFiler</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links

AcMapDMThematicTableItemLinetype Class,
AcMapDMThematicTableItemLinetype Class

AcMapDMThematicTableItemLinetype:: dwgOutFields Method

AcMapDMThematicTableItemLinetype Class |
AcMapDMThematicTableItemLinetype Class

Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide.

```cpp
virtual Acad::ErrorStatus dwgOutFields(
    AcDbDwgFiler* pFiler
) const;
```

Parameters

- `pFiler` Input filer to use to write the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Let this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide.

```cpp
virtual Acad::ErrorStatus dxfInFields(
    AcDbDxfFiler* pFiler
);
```

### Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pFiler</td>
<td>Input filer to use to read the object's data.</td>
</tr>
</tbody>
</table>

### Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.doc-omatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
AcMapDMThematicTableItemLinetype Class
AcMapDMThematicTableItemLinetype Class
AcMapDMThematicTableItemLinetype:: dxfOutFields Method
AcMapDMThematicTableItemLinetype Class | AcMapDMThematicTableItemLinetype Class

Let this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide.

```cpp
virtual Acad::ErrorStatus dxfOutFields(const AcDbDxfFiler* pFiler)
```

**Parameters**

- **pFiler** Input filer to use to write in the object's data.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns the implementation object.

```cpp
virtual AcMapDMImpThematicTableItemLinetype* Implementation() const;
```

Returns the implementation object, or NULL if unsuccessful.

Remarks

This function is for internal use only; do not call it directly.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the linetype.

```cpp
virtual const ACHAR * Linetype() const;
```

Returns the linetype name if successful, or NULL otherwise.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableItemLinetype Class,
AcMapDMThematicTableItemLinetype Class
AcMapDMThematicTableItemLinetype:: LinetypeId Method
AcMapDMThematicTableItemLinetype Class |
AcMapDMThematicTableItemLinetype Class

Retrieves the linetype AcDbObjectId.

AcDbObjectId LinetypeId() const;

Returns

Returns the linetype AcDbObjectId if successful, or AcDbObjectId::kNull otherwise.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the linetype AcDbObjectId.

```
Acad::ErrorStatus SetLinetypeId(
    const AcDbObjectId idLinetype
);
```

Parameters

- `idLinetype` Input linetype AcDbObjectId.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

```cpp
virtual ~AcMapDMThematicTableItemLineweight();
```

Returns nothing.

Created with a commercial version of [Doc-O-Matic](http://example.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Constructs an instance of this class.

AcMapDMThematicTableItemLineweight();

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Constructs an instance of this class by using another instance of this class.

```cpp
AcMapDMThematicTableItemLineweight(const AcMapDMThematicTableItemLineweight& other);
```

**Parameters**
- `other` Input other instance of this class.

**Returns**

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableItemLineweight Class,
AcMapDMThematicTableItemLineweight Class
AcMapDMThematicTableItemLineweight:: dwgInFields Method
AcMapDMThematicTableItemLineweight Class |
AcMapDMThematicTableItemLineweight Class

Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide.

**virtual** Acad::ErrorStatus dwgInFields(
    AcDbDwgFiler* pFiler
);

Parameters
pFiler Input filer to use to read the object's data.

Returns
Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapDMThematicTableItemLineweight Class,
AcMapDMThematicTableItemLineweight Class
AcMapDMThematicTableItemLineweight:: dwgOutFields Method
AcMapDMThematicTableItemLineweight Class |
AcMapDMThematicTableItemLineweight Class

Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide.

```cpp
virtual Acad::ErrorStatus dwgOutFields(
   AcDbDwgFiler* pFiler
) const;
```

Parameters Description
---
pFiler Input filer to use to write the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapDMThematicTableItemLineweight Class,
AcMapDMThematicTableItemLineweight Class
AcMapDMThematicTableItemLineweight:: dxfInFields Method
AcMapDMThematicTableItemLineweight Class |
AcMapDMThematicTableItemLineweight Class

Lets this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide.

**virtual** Acad::ErrorStatus dxfInFields(
   AcDbDxfFiler* pFiler
);

Parameters Description
pFiler Input filer to use to read the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapDMThematicTableItemLineweight Class,
AcMapDMThematicTableItemLineweight Class
AcMapDMThematicTableItemLineweight:: dxfOutFields Method
AcMapDMThematicTableItemLineweight Class |
AcMapDMThematicTableItemLineweight Class

Lets this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide.

virtual Acad::ErrorStatus dxfOutFields(
    AcDbDxfFiler* pFiler
) const;

Parameters                  Description
pFiler                      Input filer to use to write in the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableItemLineweight Class,
AcMapDMThematicTableItemLineweight Class
AcMapDMThematicTableItemLineweight:: Implementation Method
AcMapDMThematicTableItemLineweight Class | 
AcMapDMThematicTableItemLineweight Class

Returns the implementation object.

virtual AcMapDMImpThematicTableItemLineweight* Implementation() const

Returns the implementation object, or NULL if unsuccessful.

Remarks

This function is for internal use only; do not call it directly.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the lineweight.

```cpp
virtual AcDb::LineWeight Lineweight() const;
```

Returns the lineweight.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the lineweight.

```cpp
virtual Acad::ErrorStatus SetLineweight(
    AcDb::LineWeight lineweight
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>lineweight</td>
<td>Input lineweight.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

```cpp
virtual ~AcMapDMThematicTableItemPlotstyle();
```

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Constructs an instance of this class.

AcMapDMThematicTableItemPlotstyle();

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Constructs an instance of this class by using another instance of this class.

```
AcMapDMThematicTableItemPlotstyle(
    const AcMapDMThematicTableItemPlotstyle& other
);
```

**Parameters**

- `other`  
  Input other instance of this class.

**Returns**

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableItemPlotstyle Class,
AcMapDMThematicTableItemPlotstyle Class
AcMapDMThematicTableItemPlotstyle:: dwgInFields Method
AcMapDMThematicTableItemPlotstyle Class |
AcMapDMThematicTableItemPlotstyle Class

Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide.

virtual Acad::ErrorStatus dwgInFields(
   AcDbDwgFiler* pFiler
);
Parameters Description
pFiler Input filer to use to read the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableItemPlotstyle Class,
AcMapDMThematicTableItemPlotstyle Class
AcMapDMThematicTableItemPlotstyle:: dwgOutFields Method
AcMapDMThematicTableItemPlotstyle Class |
AcMapDMThematicTableItemPlotstyle Class

Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide.

```cpp
virtual Acad::ErrorStatus dwgOutFields(
    AcDbDwgFiler* pFiler
) const;
```

Parameters Description
pFiler Input filer to use to write the object's data.

Returns
Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Lets this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide.

```cpp
virtual Acad::ErrorStatus dxfInFields(
   AcDbDxfFiler* pFiler
);
```

**Parameters**

- `pFiler` 
  Input filer to use to read the object's data.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableItemPlotstyle Class,
AcMapDMThematicTableItemPlotstyle Class
AcMapDMThematicTableItemPlotstyle:: dxfOutFields Method
AcMapDMThematicTableItemPlotstyle Class |
AcMapDMThematicTableItemPlotstyle Class

Lets this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide.

virtual Acad::ErrorStatus dxfOutFields(
   AcDbDxfFiler* pFiler
) const;

Parameters Description
pFiler Input filer to use to write in the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns the implementation object.

```cpp
virtual AcMapDMImpThematicTableItemPlotstyle* Implementation() const
```

Returns the implementation object, or NULL if unsuccessful.

Remarks

This function is for internal use only; do not call it directly.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the plotstyle name.

```cpp
virtual const ACHAR* PlotstyleName() const;
```

Returns the plotstyle name if successful, or NULL otherwise.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

```cpp
virtual ~AcMapDMThematicTableItemText();
```

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableItemText Class, AcMapDMThematicTableItemText Class

AcMapDMThematicTableItemText::
AcMapDMThematicTableItemText Constructor
AcMapDMThematicTableItemText Class | AcMapDMThematicTableItemText Class

Constructs an instance of this class.

AcMapDMThematicTableItemText();

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Constructs an instance of this class by using another instance of this class.

```cpp
AcMapDMThematicTableItemTextText( // const AcMapDMThematicTableItemTextText& other
);
```

Parameters

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>other</td>
</tr>
</tbody>
</table>

Input other instance of this class.

Returns

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.modularencapsulation.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapDMThematicTableItemText Class, AcMapDMThematicTableItemText Class
AcMapDMThematicTableItemText:: Angle Method
AcMapDMThematicTableItemText Class | AcMapDMThematicTableItemText Class

Retrieves the angle.

virtual int Angle() const;

Returns

Returns the angle.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableItemText Class, AcMapDMThematicTableItemText Class
AcMapDMThematicTableItemText:: Color Method
AcMapDMThematicTableItemText Class | AcMapDMThematicTableItemText Class

Retrieves the color.

```cpp
virtual AcCmColor Color() const;
```

Returns

Returns the color.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableItemText Class, AcMapDMThematicTableItemText Class
AcMapDMThematicTableItemText::dwgInFields Method
AcMapDMThematicTableItemText Class | AcMapDMThematicTableItemText Class

Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide.

```cpp
template<>
virtual Acad::ErrorStatus dwgInFields(
   AcDbDwgFiler* pFiler
);
```

Parameters  | Description
------------|-------------
pFiler       | Input filer to use to read the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableItemText Class, AcMapDMThematicTableItemText Class
AcMapDMThematicTableItemText:: dwgOutFields Method
AcMapDMThematicTableItemText Class | AcMapDMThematicTableItemText Class

Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide.

```cpp
virtual Acad::ErrorStatus dwgOutFields(
    AcDbDwgFiler* pFiler
) const;
```

Parameters

| pFiler | Input filer to use to write the object's data. |

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableItemText Class, AcMapDMThematicTableItemText Class
AcMapDMThematicTableItemText:: dxfInFields Method
AcMapDMThematicTableItemText Class | AcMapDMThematicTableItemText Class

Lets this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide.

```cpp
virtual Acad::ErrorStatus dxfInFields(
   AcDbDxfFiler* pFiler
);
```

Parameters Description
pFiler Input filer to use to read the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableItemText Class, AcMapDMThematicTableItemText Class
AcMapDMThematicTableItemText:: dxfOutFields Method
AcMapDMThematicTableItemText Class | AcMapDMThematicTableItemText Class

Lets this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide.

virtual Acad::ErrorStatus dxfOutFields(
   AcDbDxfFiler* pFiler
) const;

Parameters Description
pFiler Input filer to use to write in the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the height.

\textbf{virtual double} \textit{Height()} const;

Returns

Returns the height.

Created with a commercial version of \textit{Doc-O-Matic}. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns the implementation object.

```cpp
virtual AcMapDMImpThematicTableItemText* Implementation() const;
```

Returns the implementation object, or NULL if unsuccessful.

Remarks

This function is for internal use only; do not call it directly.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the insertion point.

```cpp
virtual AcMapThematicTableItemInsertionPoint InsertPoint() const;
```

Returns an `AcMapThematicTableItemInsertionPoint` value.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableItemText Class, AcMapDMThematicTableItemText Class
AcMapDMThematicTableItemText:: Justification Method
AcMapDMThematicTableItemText Class | AcMapDMThematicTableItemText Class

Retrieves the justification.

virtual AcMapThematicTableItemJustificationType Justification() const

Returns

Returns an AcMapThematicTableItemJustificationType value.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the layer AcDbObjectId.

```
AcDbObjectId LayerId() const;
```

Returns the layer AcDbObjectId if successful, or AcDbObjectId::kNull otherwise.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableItemText Class, AcMapDMThematicTableItemText Class
AcMapDMThematicTableItemText:: LayerName Method
AcMapDMThematicTableItemText Class | AcMapDMThematicTableItemText Class

Retrieves the layer name.

virtual const ACHAR* LayerName() const;

Returns

Returns the layer name if successful, or NULL otherwise.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the angle.

```cpp
virtual Acad::ErrorStatus SetAngle(
    int angle
);
```

Parameters

- **angle**: Input angle.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at **support@toolsfactory.com**.
Sets the color.

```cpp
virtual Acad::ErrorStatus SetColor(
    const AcCmColor color
);
```

Parameters

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>color</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the height.

```cpp
virtual Acad::ErrorStatus SetHeight(
    double height
);
```

**Parameters**
- **height**: Input height.

**Returns**
- Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the insertion point.

```cpp
virtual Acad::ErrorStatus SetInsertPoint(
    AcMapThematicTableItemInsertionPoint insertPoint
);
```

**Parameters**
- `insertPoint`: Input `AcMapThematicTableItemInsertionPoint` value.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the justification type.

```cpp
virtual Acad::ErrorStatus SetJustification(
    AcMapDMThematicTableItemJustificationType justification);
```

**Parameters**

- `justification`: Input `AcMapDMThematicTableItemJustificationType` value.

**Returns**

Returns `Acad::eOk` if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the layer AcDbObjectId.

```cpp
Acad::ErrorStatus SetLayerId(const AcDbObjectId idLayer);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>idLayer</td>
<td>Input layer AcDbObjectId.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the style AcDbObjectId.

```cpp
Acad::ErrorStatus SetStyleId(
    const AcDbObjectId idStyle
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>idStyle</td>
<td>Input style AcDbObjectId.</td>
</tr>
</tbody>
</table>

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableItemText Class, AcMapDMThematicTableItemText Class
AcMapDMThematicTableItemText:: StyleId Method
AcMapDMThematicTableItemText Class | AcMapDMThematicTableItemText Class

Retrieves the style AcDbObjectId.

AcDbObjectId StyleId() const;

Returns

Returns the style AcDbObjectId if successful, or AcDbObjectId::kNull otherwise.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableItemText Class, AcMapDMThematicTableItemText Class
AcMapDMThematicTableItemText:: StyleName Method
AcMapDMThematicTableItemText Class | AcMapDMThematicTableItemText Class

Retrieves the style name.

virtual const ACHAR* StyleName() const;

Returns

Returns the style name if successful, or NULL otherwise.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableItemText Class, AcMapDMThematicTableItemText
Class
AcMapDMThematicTableItemText:: Text Method
AcMapDMThematicTableItemText Class | AcMapDMThematicTableItemText
Class

Retrieves the text.

```cpp
virtual const ACHAR* Text() const;
```

Returns

Returns the text if successful, or NULL otherwise.

Created with a commercial version of Doc-O-Matic. In order to make this
message disappear you need to register this software. If you have problems
registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableRow Class, AcMapDMThematicTableRow Class
AcMapDMThematicTableRow:: mpAnnotation Data Member
AcMapDMThematicTableRow Class | AcMapDMThematicTableRow Class

Annotation value type.

AcMapDMThematicTableRowItemAnnotation * mpAnnotation;
Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableRow Class, AcMapDMThematicTableRow Class
AcMapDMThematicTableRow:: mpBlock Data Member
AcMapDMThematicTableRow Class | AcMapDMThematicTableRow Class

Block value type.

AcMapDMThematicTableItemBlock * mpBlock;
Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Color value type.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableRow Class, AcMapDMThematicTableRow Class
AcMapDMThematicTableRow:: mpDataValue Data Member
AcMapDMThematicTableRow Class | AcMapDMThematicTableRow Class

Text/value-pair value type.

AcMapDMThematicTableItemDataValue * mpDataValue;
Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableRow Class, AcMapDMThematicTableRow Class
AcMapDMThematicTableRow:: mpHatch Data Member
AcMapDMThematicTableRow Class | AcMapDMThematicTableRow Class

Hatch value type.

AcMapDMThematicTableItemHatch * mpHatch;
Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableRow Class,  AcMapDMThematicTableRow Class
AcMapDMThematicTableRow:: mpLegendText Data Member
AcMapDMThematicTableRow Class |  AcMapDMThematicTableRow Class

Legend-text value type.

AcMapDMThematicTableRowItemLegendText * mpLegendText;
Created with a commercial version of  Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableRow Class, AcMapDMThematicTableRow Class
AcMapDMThematicTableRow:: mpLinestyle Data Member
AcMapDMThematicTableRow Class | AcMapDMThematicTableRow Class

Linestyle value type.

AcMapDMThematicTableRowItemLinestyle * mpLinestyle;
Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableRow Class, AcMapDMThematicTableRow Class
AcMapDMThematicTableRow:: mpLinetype Data Member
AcMapDMThematicTableRow Class | AcMapDMThematicTableRow Class

Linetype value type.

AcMapDMThematicTableItemLinetype * mpLinetype;
Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Lineweighting value type.

AcMapDMThematicTableItemLineweight * mLineweight;

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableRow Class, AcMapDMThematicTableRow Class
AcMapDMThematicTableRow:: mplPlotstyle Data Member
AcMapDMThematicTableRow Class | AcMapDMThematicTableRow Class

Plotstyle value type.

AcMapDMThematicTableItemPlotstyle * mplPlotstyle;
Created with a commercial version of Doc-O-Matic. In order to make this
message disappear you need to register this software. If you have problems
registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMThematicTableRow Class, AcMapDMThematicTableRow Class
AcMapDMThematicTableRow:: mpText Data Member
AcMapDMThematicTableRow Class | AcMapDMThematicTableRow Class

Text value type.

AcMapDMThematicTableItemText * mpText;
Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

```cpp
virtual ~AcMapDMThematicTableRow();
```

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Constructs an instance of this class.

AcMapDMThematicTableRow();

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Closes all members.

```cpp
Acad::ErrorStatus closeRow();
```

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Initializes the member variables.

```cpp
void Init();
```

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

```
virtual ~AcMapDMTopoElement();
```

Returns

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links

AcMapDMTopoElement Class, AcMapDMTopoElement Class
AcMapDMTopoElement:: AcMapDMTopoElement Constructor
AcMapDMTopoElement Class | AcMapDMTopoElement Class

Constructs an instance of this class.

AcMapDMTopoElement();
Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Runs the topological query against the current drawing and acquires both selected entities and topology objects.

```cpp
virtual Acad::ErrorStatus AcquireEntities();
```

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Clones objects from the source drawings.

```cpp
virtual bool ClonesObjectsFromExternalSource() const;
```

Returns

Always returns false.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Erasers queried objects.

**virtual** Acad::ErrorStatus DismissEntities();

Returns Academic::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](#). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapDMTopoElement Class, AcMapDMTopoElement Class
AcMapDMTopoElement::dwgInFields Method
AcMapDMTopoElement Class | AcMapDMTopoElement Class

Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dwgInFields(
    AcDbDwgFiler* pFiler
);
```

Parameters

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pFiler</td>
</tr>
</tbody>
</table>

Input filer to use to read the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Let this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dwgOutFields(
    AcDbDwgFiler* pFiler
) const;
```

**Parameters**
- `pFiler`: Input filer to use to write the object's data.

**Returns**
- Returns `Acad::eOk` if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapDMTopoElement Class, AcMapDMTopoElement Class
AcMapDMTopoElement:: dxfInFields Method
AcMapDMTopoElement Class | AcMapDMTopoElement Class

Lets this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dxfInFields(
    AcDbDxfFiler* pFiler
);
```

Parameters

- **pFiler**: Input filer to use to read the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMTopoElement Class, AcMapDMTopoElement Class
AcMapDMTopoElement:: dxfOutFields Method
AcMapDMTopoElement Class | AcMapDMTopoElement Class

Lets this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```
virtual Acad::ErrorStatus dxfOutFields(
    AcDbDxfFiler* pFiler
) const;
```

Parameters | Description
--- | ---
pFiler | Input filer to use to write in the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the query definition's data-source descriptor.

```cpp
virtual Acad::ErrorStatus GetAcquisitionCriteria(
    AcMapDMDataSourceDescriptor* pDataSourceDsc
) const;
```

**Parameters**

- `pDataSourceDsc` Output pointer to the data-source descriptor.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](http://www.toolsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Invoked when an AutoCAD Map project is initialized.

```cpp
virtual bool OnMapProjectInitialized(
    AcMapProject* pMapProject);
```

Parameters

- `pMapProject`: Input AutoCAD Map project.

Returns

Returns true if successful.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the query definition's data-source descriptor.

```cpp
virtual Acad::ErrorStatus SetAcquisitionCriteria(
    const AcMapDMDataSourceDescriptor* pDataSourceDsc);
```

**Parameters**

- `pDataSourceDsc`  
  Description: Input pointer to a data-source descriptor.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

**Remarks**

This function determines which entities become part of this element.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

`~AcMapDMTopoElementDataSourceDescriptor();`

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Constructs an instance of this class.

AcMapDMTopoElementDataSourceDescriptor();

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

\textbf{virtual} \texttt{\sim AcMapDMTopoQueryDataSourceDescriptor()};

Returns

Returns nothing.

Created with a commercial version of \texttt{Doc-O-Matic}. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at \texttt{support@toolsfactory.com}.  

Links

\texttt{AcMapDMTopoQueryDataSourceDescriptor Class, AcMapDMTopoQueryDataSourceDescriptor Class}

\texttt{AcMapDMTopoQueryDataSourceDescriptor::\texttt{\sim AcMapDMTopoQueryDataSourceDescriptor Destructor}}

\texttt{AcMapDMTopoQueryDataSourceDescriptor Class | AcMapDMTopoQueryDataSourceDescriptor Class}
Constructs an instance of this class.

AcMapDMTopoQueryDataSourceDescriptor();
Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the query definition.

```cpp
virtual Acad::ErrorStatus GetQuery(
    struct resbuf* & pRb
) const;
```

**Parameters**
- **pRb**
  - Description: Output resbuf object that contains the query.

**Returns**
- Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the query definition.

```cpp
virtual Acad::ErrorStatus SetQuery(const struct resbuf* pRb);
```

**Parameters**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pRb</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Destroys an instance of this class.

```cpp
virtual ~AcMapDMTopoQueryElement();
```

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
AcMapDMTopoQueryElement Class

Constructs an instance of this class.

AcMapDMTopoQueryElement();

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Runs the topological query and acquires both selected entities and topology objects.

```cpp
virtual Acad::ErrorStatus AcquireEntities();
```

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Clones objects from the source drawings.

```c++
virtual bool ClonesObjectsFromExternalSource() const;
```

Returns

Always returns true.

Created with a commercial version of [Doc-O-Matic](http://www.docsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Erases entities that are part of this query element.

```cpp
virtual Acad::ErrorStatus DismissEntities();
```

Returns Acad::eOk if successful; otherwise, returns a different error code.

Remarks

See also AcquireEntities().

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Dismisses the current stylization.

```cpp
virtual Acad::ErrorStatus DismissStylization(
    bool bDismissEntities
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bDismissEntities</td>
<td>Input true to unstylize the entities.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMTopoQueryElement Class, AcMapDMTopoQueryElement Class

AcMapDMTopoQueryElement:: dwgInFields Method
AcMapDMTopoQueryElement Class | AcMapDMTopoQueryElement Class

Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

virtual Acad::ErrorStatus dwgInFields(
    AcDbDwgFiler* pFiler
);

Parameters Description
pFiler Input filer to use to read the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMTopoQueryElement Class, AcMapDMTopoQueryElement Class
AcMapDMTopoQueryElement:: dwgOutFields Method
AcMapDMTopoQueryElement Class | AcMapDMTopoQueryElement Class

Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dwgOutFields(
    AcDbDwgFiler* pFiler
) const;
```

Parameters Description
---

pFiler Input filer to use to write the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMTopoQueryElement Class, AcMapDMTopoQueryElement Class
AcMapDMTopoQueryElement:: dxfInFields Method
AcMapDMTopoQueryElement Class | AcMapDMTopoQueryElement Class

Lets this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dxfInFields(
    AcDbDxfFiler* pFiler
);
```

Parameters Description
---
pFiler Input filer to use to read the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Lets this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide. This overloaded function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dxfOutFields(
    AcDbDxfFiler* pFiler
) const;
```

**Parameters**
- **pFiler**: Input filer to use to write in the object's data.

**Returns**
- Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Called during database destruction to trigger the deletion of topology-managed cookies.

```cpp
void EmptyCookieJar();
```

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Enables or disables a style reference.

```cpp
virtual Acad::ErrorStatus EnableStyle(
    AcDbObjectId styleRefId,
    bool bNewEnableStatus
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>styleRefId</td>
<td>Input style reference ID.</td>
</tr>
<tr>
<td>bNewEnableStatus</td>
<td>Input new enable status.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieve the query definition's data-source descriptor.

```cpp
virtual Acad::ErrorStatus GetAcquisitionCriteria(
    AcMapDMDataSourceDescriptor* pDataSourceDsc
) const;
```

**Parameters**

<table>
<thead>
<tr>
<th>pDataSourceDsc</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pDataSourceDsc</td>
<td>Output pointer to the data-source descriptor.</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Invoked when an AutoCAD Map project is initialized.

**virtual bool** OnMapProjectInitialized(
    AcMapProject* pMapProject
);

- **Parameters**
  - pMapProject: Input AutoCAD Map project.

- **Returns**
  - Returns true if successful.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Removes a style reference from this element.

```cpp
template<>
virtual Acad::ErrorStatus RemoveStyle(
    AcDbObjectId styleRefId);
```

### Parameters
- **styleRefId**
  - **Description:** Input style reference ID.

### Returns
- Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the query definition's data-source descriptor.

```cpp
virtual Acad::ErrorStatus SetAcquisitionCriteria(
    const AcMapDMDataSourceDescriptor* pDataSourceDsc);
```

Parameters  | Description
--- | ---
pDataSourceDsc  | Input pointer to a data-source descriptor.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Remarks

This function determines which entities become part of this element. Current-drawing topologies are ignored.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapDMTopoQueryElement Class, AcMapDMTopoQueryElement Class

AcMapDMTopoQueryElement:: SetVisible Method
AcMapDMTopoQueryElement Class | AcMapDMTopoQueryElement Class

Makes the entities that are part of this element visible or invisible.

```cpp
virtual Acad::ErrorStatus SetVisible(
    bool bNewVal,
    double dScale
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bNewVal</td>
<td>Input true to make the entities visible, or false to make them invisible.</td>
</tr>
<tr>
<td>dScale</td>
<td>Input scale at which to make entities visible.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Stylizes the current or updated selection at the current scale.

```cpp
virtual Acad::ErrorStatus UpdateStylization(
    bool bAcquireEntities
);  
```

**Parameters**

- **bAcquireEntities**
  - Input true to acquire entities as part of update operation.

**Returns**

- Returns Acad::eOk if successful. Returns Acad::eAmbiguousOutput if bAcquireEntities is true but the previous selection has not been dismissed.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
This is m_ulEntitiesExported, a member of class CExportResults.

```
unsigned long m_ulEntitiesExported;
```

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
This is `m_ulEntitiesSkippedCouldn'tTransform`, a member of class `CExportResults`.

```c
unsigned long m_ulEntitiesSkippedCouldn'tTransform;
```

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
This is CExportResults, a member of class CExportResults.

CExportResults();
Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
This is `m_ulEntitiesImported`, a member of class `CImportResults`.

```cpp
unsigned long m_ulEntitiesImported;
```

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
This is \texttt{m\_ulEntitiesSkippedCouldntTransform}, a member of class \texttt{CImportResults}.

\texttt{unsigned long \_\_u\_\_m\_ulEntitiesSkippedCouldntTransform;}  
Created with a commercial version of \texttt{Doc-O-Matic}. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at \texttt{support@toolsfactory.com}.
This is m_bEvaluateHatchFailed, a member of class CImportResults.

```cpp
bool m_bEvaluateHatchFailed;
```

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
This is `m_ulEntitiesWithColorCloseToBackground`, a member of class `CImportResults`.

`unsigned long m_ulEntitiesWithColorCloseToBackground;`

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
This is m_ulEntitiesClassified, a member of class CImportResults.

`unsigned long m_ulEntitiesClassified;`

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
This is `m_ulEntitiesNotClassifiable`, a member of class `CImportResults`.

```cpp
unsigned long m_ulEntitiesNotClassifiable;
```

Created with a commercial version of `Doc-O-Matic`. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at `support@toolsfactory.com`.
This is `m_ulOutOfRangeNotFixed`, a member of class `CImportResults`.

```c
unsigned long m_ulOutOfRangeNotFixed;
```

Created with a commercial version of [Doc-O-Matic](https://toolsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
This is CImportResults, a member of class CImportResults.

CImportResults();
Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

\texttt{virtual} \ AcMapIEColumn();

Returns

Returns nothing.

Created with a commercial version of \texttt{Doc-O-Matic}. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at \texttt{support@toolsfactory.com}. 

\begin{itemize}
\item \texttt{AcMapIEColumn Class}
\item \texttt{AcMapIEColumn Class}
\end{itemize}
Returns the name of the feature class property that is mapped to this column.

```cpp
template virtual const ACHAR* ColumnClassMapping() const = 0;
```

Returns the feature class property string. The format of this string is `category:property`, where both the `category` and `property` substrings are extended symbol names that can contain special characters. Furthermore, `category` can take the form `OD:table_name`, where `table_name` is the name of an object data table. If you are developing an international application, note that the substring "OD:" is localizable.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Returns the name of the column in the AutoCAD Map table that this incoming column is mapped to.

```cpp
virtual const ACHAR* ColumnDataMapping(
    AcMapIE::ImportDataMapping& colMappingType
) const = 0;
```

**Parameters**

- `colMappingType` Output ImportDataMapping type of column mapping. This value is provided as a convenience; the type was set previously in the input layer in the AcMapIEInputLayer class.

**Returns**

Returns the name of the mapped column.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns the (fixed) name of this incoming column.

```cpp
virtual const ACHAR* ColumnName() const = 0;
```

Returns the name of this column.

Remarks

The allowable characters are: a-z A-Z 0-9 . { } _ -. 

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Destroys an instance of this class.

```cpp
virtual ~AcMapIEColumnIterator();
```

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Determines whether the iterator has reached the end of the collection.

\texttt{virtual bool Done() = 0;}

Returns

Returns true if the iterator has reached the end of the collection; otherwise, returns false.

Created with a commercial version of \texttt{Doc-O-Matic}. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at \texttt{support@toolsfactory.com}. 
Retrieves the current element in the iteration.

```cpp
virtual bool Get(
    AcMapIEColumn*& pColumn
) = 0;
```

**Parameters**

- pColumn: Output pointer to the current AcMapIEColumn.

**Returns**

Returns true if the current element is valid, or false if the iteration has no more elements to retrieve.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapIEColumnIterator Class, AcMapIEColumnIterator Class
AcMapIEColumnIterator:: Rewind Method
AcMapIEColumnIterator Class | AcMapIEColumnIterator Class

Moves to the first element in the iteration.

**virtual void** Rewind() = 0;

Returns

Returns nothing.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Advances to the next element in the iteration.

**virtual void** Step() = 0;

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Adds a reactor to this exporter. You must derive a reactor from the virtual base class `AcMapIEExportReactor`. Reactors are triggered through only the API; an end-user cannot trigger a reactor by using the AutoCAD Map user interface. Reactors are live until they are removed explicitly with `RemoveReactor()` or until AutoCAD Map exits. An exporter is a singleton; all callers add reactors to the same exporter instance. If an exporter has multiple reactors, they are called in the order added.

```c++
virtual AcMapIE::ErrCode AddReactor(
    AcMapIEExportReactor * pReactor
) = 0;
```

Parameters

- **pReactor**
  - Input `AcMapIEExportReactor` export reactor to add.

Returns

Returns `AcMapIE::ErrCode kErr_OK` if successful. Returns `AcMapIE::ErrCode kErr_Fail` if the process failed.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Determines whether closed polylines are exported as polygons.

```cpp
virtual bool ClosedPolylinesAsPolygons() const = 0;
```

Returns true if closed polylines are exported as polygons, or false if they are exported as polylines.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Counts the number of objects to export in the current drawing.

```cpp
virtual AcMapIE::ErrCode CountObjects(
    int& numPoints,
    int& numLines,
    int& numPolygons,
    int& numText,
    int& numTotal
) = 0;
```

Parameters | Description
--- | ---
numPoints | Output number of points to export.
numLines | Output number of lines to export.
numPolygons | Output number of polygons to export.
numText | Output number of text entities to export.
numTotal | Output total number of entities to export.

Returns

Returns AcMapIE::ErrCode kErr_OK if successful. Returns AcMapIE::ErrCode kErr_OutOfMemory if system resources are insufficient. Returns AcMapIE::ErrCode kErr_Fail if the process failed for some other reason.

Remarks

The output values depend on the Layer filter, Feature class filter, and selection set.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Retrieve the discretization angle. The discretization angle setting is stored in the file *MapExport.ini*, and labeled *SegmentationDegrees*.

```cpp
virtual bool DiscretizationAngle(
    double& dDegrees
) const = 0;
```

**Parameters**

- **dDegrees**: Output discretization angle, in decimal degrees.

**Returns**

Returns true if the exporter was initialized; otherwise, returns false.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the current driver options. Driver options can be set in the file
`MapExport.ini`, by calling `InvokeDriverOptionsDialog()`, or by loading a profile.
Use this function to first retrieve the current driver options before adding driver
options with `SetDriverOptions()`.

```cpp
virtual AcMapI::ErrCode DriverOptions(
    AcMapINameValueIterator* & nameValueIter
) = 0;
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>nameValueIter</td>
<td>Output driver options, as <code>AcMapINameValueIterator</code> name-value pairs. The</td>
</tr>
<tr>
<td></td>
<td>caller is responsible for deleting this object.</td>
</tr>
</tbody>
</table>

Returns

Returns `AcMapI::ErrCode kErr_OK` if successful. Returns `AcMapI::ErrCode kErr_Fail` if the process failed.

Created with a commercial version of `Doc-O-Matic`. In order to make this
message disappear you need to register this software. If you have problems
registering this software please contact us at `support@toolsfactory.com`.
Invokes the driver-specific Driver Options dialog box. Not all drivers have dialog boxes, so this function may finish immediately, with the returned error code indicating the status of the dialog box. Any setting changes that an AutoCAD Map user makes in this dialog box are reflected in subsequent calls to DriverOptions(). You can set driver options programmatically by using SetDriverOptions(). Any driver options in the file MapExport.ini are read and available in the set of driver options returned by DriverOptions().

```
virtual AcMapIE::ErrCode InvokeDriverOptionsDialog() = 0;
```

Returns

Returns AcMapIE::ErrCode kErr_OK if successful. Returns AcMapIE::ErrCode kErr_DialogNotAvailable if no driver options dialog box exists for this format. Returns AcMapIE::ErrCode kErr_Fail if the dialog box cannot be created or if the process failed for some other reason.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Performs the export.

```cpp
virtual AcMapIE::ErrCode Export(
    AcMapIE::CExportResults& results,
    bool bUseProgressDialog = false
) = 0;
```

**Parameters**

- `results`: Output `CExportResults` export results.
- `bUseProgressDialog`: Input true to display AutoCAD Map's progress dialog box during the export, or false to suppress this dialog box.

**Returns**

Returns `AcMapIE::ErrCode` `kErr_OK` if successful. Returns `AcMapIE::ErrCode` `kErr_OutOfMemory` if system resources are insufficient. Returns `AcMapIE::ErrCode` `kErr_Fail` if the process failed for some other reason.

**Remarks**

This operation is time-consuming for a large number of entities.

Created with a commercial version of [Doc-O-Matic](https://example.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Determines whether a selection set is ignored during export.

```cpp
virtual bool ExportAll() const = 0;
```

Returns false if a selection set is used; returns true if a selection set and all filtering is to be ignored.

Remarks

The default is to not export all but to use a selection set.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the source-column and output-column data mappings for export. The output collection will be empty if you have not first loaded an export profile file with LoadEPF().

```cpp
virtual AcMapIE::ErrCode ExportDataMappings(
    AcMapIEExpressionTargetIterator* &expTargetIter
) = 0;
```

Parameters  
<table>
<thead>
<tr>
<th>expTargetIter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AcMapIEExpressionTargetIterator</td>
<td>expression-target pairs. The caller is responsible for deleting this object.</td>
</tr>
</tbody>
</table>

Returns  
Returns AcMapIE::ErrCode kErr_OK if successful. Returns AcMapIE::ErrCode kErr_Fail if the process failed.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Retrieves the feature classes as a list of comma-separated patterns.

\texttt{virtual const ACHAR* FeatureClassFilter() const = \theta;}

Returns a list of feature classes, as a string.

Remarks

The patterns follow the rules for acutWcMatch. The comma character is not permitted in extended symbol names, so the list is unambiguous.

Created with a commercial version of \texttt{Doc-O-Matic}. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at \texttt{support@toolsfactory.com}. 

Links

\texttt{AcMapIEExporter Class, AcMapIEExporter Class}

\texttt{AcMapIEExporter:: FeatureClassFilter Method}

\texttt{AcMapIEExporter Class | AcMapIEExporter Class}
Returns the name of the format set during initialization. The format name is set with Init().

```
virtual const ACHAR* FormatName() const = 0;
```

Returns the format name.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the layers as a list of comma-separated patterns.

```cpp
virtual const ACHAR* LayerFilter() const = 0;
```

Returns a list of layers, as a string.

Remarks

The patterns follow the rules for acutWcMatch. The comma character is not permitted in extended symbol names, so the list is unambiguous.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the settings for mapping layers in the drawing to DGN levels.

```cpp
virtual AcMapIE::ErrCode LayerLevelMapping(
    bool& bUse,
    AcMapIENamValueIterator* & mapping
) = 0;

Parameters

bUse
Output true if layer mapping is enabled; otherwise, false.

mapping
Output mapping, as AcMapIENamValueIterator name-value pairs. In each name-value pair, an AutoCAD Map layer name is the name component and its corresponding level number is the value component. The caller is responsible for deleting this object.

Returns

Returns AcMapIE::ErrCode kErr_OK if successful. Returns AcMapIE::ErrCode kErr_Fail if the process failed.

Remarks

DGN supports 64 levels, numbered 0 to 63. Level 0 is reserved, so the default mapping starts with level 1.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Removes a reactor from this exporter.

**virtual** `AcMapIE::ErrCode RemoveReactor(`

`AcMapIEExportReactor * pReactor`

`) = 0;`

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pReactor</td>
<td>Input <code>AcMapIEExportReactor</code> export reactor to remove.</td>
</tr>
</tbody>
</table>

Returns AcMapIE::ErrCode kErr_OK if successful. Returns AcMapIE::ErrCode kErr_Invalid if the reactor does not already exist. Returns AcMapIE::ErrCode kErr_Fail if the process failed for some other reason.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Retrieves the selection set.

```cpp
virtual AcMapIE::ErrCode SelectionSet(
    ads_name& selset
) const = 0;
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>selset</td>
<td>Output name of the ADS (AutoCAD Development System) selection set. Do not free this object.</td>
</tr>
</tbody>
</table>

Returns

Returns AcMapIE::ErrCode kErr_OK if successful. Returns AcMapIE::ErrCode kErr_Fail if the process failed.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Changes the setting for exporting closed polylines as polygons.

```cpp
virtual AcMapIE::ErrCode SetClosedPolylinesAsPolygons(
    bool bPolygon
) = 0;
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bPolygon</td>
<td>Input true to export closed polylines as polygons; otherwise, false to export them as polylines.</td>
</tr>
</tbody>
</table>

Returns

Returns AcMapIE::ErrCode kErr_OK if successful. Returns AcMapIE::ErrCode kErr_Fail if the process failed.

Created with a commercial version of [Doc-O-Matic](https://www.docsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the discretization angle. The discretization angle setting is stored in the file MapExport.ini, and labeled SegmentationDegrees.

```cpp
virtual bool SetDiscretizationAngle(
    double dDegrees
) = 0;
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dDegrees</td>
<td>Input discretization angle to set, expressed in decimal degrees.</td>
</tr>
</tbody>
</table>

Returns

Returns true if the exporter was initialized; otherwise returns false.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the driver options. Typically, you should retrieve the current driver options from DriverOptions() and make changes as needed by using this function.

```cpp
virtual AcMapIE::ErrCode SetDriverOptions(
    AcMapIENamespaceValueCollection* & nameValueIter
) = 0;
```

**Parameters**
- `nameValueIter`  
  Input driver options, as `AcMapIENamespaceValueCollection` name-value pairs.

**Returns**

Returns `AcMapIE::ErrCode kErr_OK` if successful. Returns `AcMapIE::ErrCode kErr_OutOfMemory` if system resources are insufficient. Returns `AcMapIE::ErrCode kErr_Fail` if the process failed for some other reason.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Changes the setting for using a selection set during export.

```cpp
virtual AcMapIE::ErrCode SetExportAll(
    bool bTrue
) = 0;
```

**Parameters**

- **bTrue**
  - Description: Input true to export all entities in a drawing ignoring the selection set and filters; otherwise, false.

**Returns**

Returns AcMapIE::ErrCode kErr_OK if successful. Returns AcMapIE::ErrCode kErr_Fail if the process failed.

**Remarks**

A default selection set (containing the entire drawing) is used if no selection set is set explicitly.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at **support@toolsfactory.com**.
Sets the source-column and output-column data mappings for export.

```cpp
virtual AcMapIE::ErrCode SetExportDataMappings(
    AcMapIEExpressionTargetIterator* & expTargetIter
) = 0;
```

**Parameters**

- `expTargetIter` Input export data mappings, as `AcMapIEExpressionTargetIterator` expression-target pairs.

**Returns**

Returns `AcMapIE::ErrCode kErr_OK` if successful. Returns `AcMapIE::ErrCode kErr_ConflictWithKey` if a target column has the same name as the unique key column; see `UseUniqueKeyField()`. Returns `AcMapIE::ErrCode kErr_Fail` if the process failed for some other reason.

**Remarks**

The valid characters for column names are: a-z A-Z 0-9 . { } _ -. (Including accented and multibyte characters.) The dot and brace characters have special meaning and should be avoided except in special circumstances.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Changes the settings for mapping layers in the drawing to DGN levels.

```cpp
virtual AcMapIE::ErrCode SetLayerLevelMapping(
    bool bUse,
    const AcMapIENamespaceNameValueIterator* &mapping
) = 0;
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bUse</td>
<td>Input true to enable layer mapping; otherwise, false.</td>
</tr>
<tr>
<td>mapping</td>
<td>Input mapping to set, as name-value pairs. In each name-value pair, specify an AutoCAD Map layer name in the name component, and its corresponding level number as a string in the value component.</td>
</tr>
</tbody>
</table>

**Returns**

Returns AcMapIE::ErrCode kErr_OK if successful. Returns AcMapIE::ErrCode kErr_Fail if the process failed.

Created with a commercial version of [Doc-O-Matic](https://www.doc-o-matic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the ADS (AutoCAD Development System) selection set.

```cpp
virtual AcMapIE::ErrCode SetSelectionSet(  
    ads_name& other
) = 0;
```

Parameters

- `other`  
  Description: Input selection set.

Returns

Returns `AcMapIE::ErrCode kErr_OK` if successful. Returns `AcMapIE::ErrCode kErr_Fail` if the process failed.

Remarks

The exporter is responsible for freeing the selection set.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapIEExporter Class, AcMapIEExporter Class
AcMapIEExporter:: TargetCoordSys Method
AcMapIEExporter Class | AcMapIEExporter Class

Retrieves the target coordinate system for the exported data.

```cpp
virtual const ACHAR* TargetCoordSys() const = 0;
```

Returns

Returns the name of the target coordinate system, as the system's Mentor abbreviation.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

```cpp
virtual ~AcMapIEExportReactor();
```

Returns

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Constructs an instance of this class.

AcMapIEExportReactor();

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Invoked after an entity is exported.

```cpp
virtual void RecordExported(
    AcDbEntity* pEnt
);
```

**Parameters**

- **pEnt**  
  Description: Input entity.

**Returns**

Returns nothing.

**Remarks**

You can use this function to track exported entities.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Invoked before an entity is exported.

```cpp
virtual bool RecordReadyForExport(
    AcDbEntity* pEnt);
```

**Parameters**

- `pEnt`  
  Input entity.

**Returns**

Returns true if the specified entity should be exported; otherwise, returns false.

**Remarks**

This entity is not a const, so you can modify it as needed. The return value indicates whether an entity is ready to export. If this function returns false, the entity is not exported and no further reactors in the chain are called. If this function returns true for all reactors in the chain, the entity is exported.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Destroys an instance of this class.

```cpp
virtual ~AcMapIEExpressionTargetIterator();
```

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapIEExpressionTargetIterator Class, AcMapIEExpressionTargetIterator Class
AcMapIEExpressionTargetIterator:: Clear Method
AcMapIEExpressionTargetIterator Class | AcMapIEExpressionTargetIterator Class

Removes all elements from the collection.

**virtual void** Clear() = 0;

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determined whether the iterator has reached the end of the collection.

```cpp
virtual bool Done() = 0;
```

Returns

Returns true if the iterator has reached the end of the collection; otherwise, returns false.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapIEExpressionTargetIterator Class, AcMapIEExpressionTargetIterator Class
AcMapIEExpressionTargetIterator:: Rewind Method
AcMapIEExpressionTargetIterator Class | AcMapIEExpressionTargetIterator Class

Moves to the first element in the iteration.

**virtual void** Rewind() = 0;
Returns

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Advances to the next element in the iteration.

```
virtual void Step() = 0;
```

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

`virtual ~AcMapIEFormat();`

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves a brief description of this format.

```cpp
virtual const ACHAR* Description() const = 0;
```

Returns the description of this format.

Remarks

This description is suitable for the Files Of Type drop-down box in AutoCAD Map's Import or Export dialog box.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the filename extension that AutoCAD Map expects to be used for this format.

```cpp
virtual const ACHAR* Extension() const = 0;
```

Returns the filename extension used for this format, as a wildcard file specification such as "*CATD.DDF" or "*.dgn".

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns the unique name which AutoCAD Map uses to refer to this format. This format name is not translated and is used in the AutoCAD Map configuration files `MapImport.ini`, `MapExport.ini`, and `MapForeignFileProperties.ini`.

```cpp
virtual const ACHAR* FormatName() const = 0;
```

Returns the unique name of this format.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links

AcMapIEFormat Class, AcMapIEFormat Class
AcMapIEFormat:: HasDriverOptionsDialog Method
AcMapIEFormat Class | AcMapIEFormat Class

Determines whether a Driver Options dialog box exists for this format.

```cpp
virtual bool HasDriverOptionsDialog() const = 0;
```

Returns true if a Driver Options dialog box exists for this format; otherwise, returns false.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns the storage type of this format.

```cpp
virtual AcMapIE::StorageType StorageType() const = 0;
```

Returns the `StorageType` storage type of this format. For export formats, this value can be any format except `kStorage_FileMultiSelect`. For import formats, this value can be `kStorage_FileAllEntityTypes`, `kStorage_FileMultiSelect`, `kStorage_FolderNoPrefix`, or `kStorage_Database`.

Created with a commercial version of [Doc-O-Matic](https://example.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapIEFormatIterator Class, AcMapIEFormatIterator Class
AcMapIEFormatIterator:: Done Method
AcMapIEFormatIterator Class | AcMapIEFormatIterator Class

Determines whether the iterator has reached the end of the collection.

**virtual bool** Done() = \(\theta\);

Returns

Returns true if the iterator has reached the end of the collection; otherwise, returns false.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at **support@toolsfactory.com**.
Retrieves the current element in the iteration.

```cpp
virtual bool Get(
    AcMapIEFormat*& pFormat
) = 0;
```

**Parameters**

- `pFormat`: Output pointer to the current AcMapIEFormat. The caller is responsible for deleting this object.

**Returns**

Returns true if the current element is valid, or false if the iteration has no more elements to retrieve.

Created with a commercial version of [Doc-O-Matic](https://www.toolsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapIEFormatIterator Class, AcMapIEFormatIterator Class
AcMapIEFormatIterator:: Rewind Method
AcMapIEFormatIterator Class | AcMapIEFormatIterator Class

Moves to the first element in the iteration.

**virtual void** Rewind() = 0;

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Advances to the next element in the iteration.

**virtual void** `Step()` = 0;

Returns nothing.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Adds a reactor to this importer. You must derive a reactor from the virtual base class AcMapIEImportReactor. Reactors are triggered through only the API; an end-user cannot trigger a reactor by using the AutoCAD Map user interface. Reactors are live until they are removed explicitly with RemoveReactor() or until AutoCAD Map exits. An importer is a singleton; all callers add reactors to the same importer instance. If an importer has multiple reactors, they are called in the order added.

```cpp
virtual AcMapIE::ErrCode AddReactor(
    AcMapIEImportReactor * pReactor
) = 0;
```

**Parameters**
- `pReactor` Input `AcMapIEImportReactor` import reactor to add.

**Returns**
- Returns `AcMapIE::ErrCode kErr_OK` if successful. Returns `AcMapIE::ErrCode kErr_Fail` if the process failed.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapIEImporter Class, AcMapIEImporter Class
AcMapIEImporter:: AuditClassifiedAfterImport Method
AcMapIEImporter Class | AcMapIEImporter Class

Determines whether classified properties are audited after import.

**virtual bool** AuditClassifiedAfterImport() **const** = 0;

Returns

Returns true if classified-property auditing is turned on, or false if it is turned off.

Remarks

An audit determines whether each classified-property value is out of range and, if so, sets it to its default value.

Created with a commercial version of [Doc-O-Matic](#). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Retrieves the current driver options. Driver options can be set in the file `MapImport.ini`, by calling `InvokeDriverOptionsDialog()`, or by loading a profile. Use this function to first retrieve the current driver options before adding driver options with `SetDriverOptions()`.

```cpp
virtual AcMapIE::ErrCode DriverOptions(
    AcMapIENameValueIterator* & pDriverOptions
) = 0;
```

Parameters

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Output driver options, as <code>AcMapIENameValueIterator</code> name-value pairs. The caller is responsible for deleting this object.</td>
</tr>
</tbody>
</table>

Returns

Returns `AcMapIE::ErrCode kErr_OK` if successful. Returns `AcMapIE::ErrCode kErr_Fail` if the process failed.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Invokes the driver-specific Driver Options dialog box. Not all drivers have dialog boxes, so this function may finish immediately, with the returned error code indicating the status of the dialog box. Any setting changes that an AutoCAD Map user makes in this dialog box are reflected in subsequent calls to DriverOptions(). You can set driver options programmatically by using SetDriverOptions(). Any driver options in the file MapImport.ini are read and available in the set of driver options returned by DriverOptions().

```cpp
virtual AcMapIE::ErrCode InvokeDriverOptionsDialog(
    bool& bSchemaHasChanged
) = 0;
```

**Parameters**
- **bSchemaHasChanged**
  - Output true if the schema has changed; otherwise, false.

**Returns**
- Returns AcMapIE::ErrCode kErr_OK if successful. Returns AcMapIE::ErrCode kErr_DialogNotAvailable if no driver options dialog box exists for this format. Returns AcMapIE::ErrCode kErr_Fail if the dialog box cannot be created or if the process failed for some other reason.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Performs the import.

```cpp
virtual AcMapIE::ErrCode Import(
    AcMapIE::CImportResults& results,
    bool bUseProgressDialog = false
) = 0;
```

**Parameters**

- `results`: Output `CImportResults` - import results.
- `bUseProgressDialog`: Input true to display AutoCAD Map's progress dialog box during the import, or false to suppress this dialog box.

**Returns**

- Returns `AcMapIE::ErrCode kErr_OK` if successful. Returns `AcMapIE::ErrCode kErr_OutOfMemory` if system resources are insufficient. Returns `AcMapIE::ErrCode kErr_Fail` if the process failed for some other reason.

**Remarks**

This operation is time-consuming for a large number of entities.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determines whether polygons are imported as closed polylines or as polygons.

```cpp
virtual bool ImportPolygonsAsClosedPolylines() const = 0;
```

Returns true if polygons are imported as closed polylines, or false if they are imported as polygons.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves an iterator over the input layers. An input layer refers to a single MIF/MID or Shape file, a level or geometry type in a DGN file, or a geometry type in a Coverage or E00 file. A layer consists of a layer name and a schema for attribute data. You are responsible for determining how this data is imported; for more information, see the AcMapIEInputLayer and AcMapIEColumn classes.

```cpp
virtual AcMapIE::ErrCode InputLayerIterator(
    AcMapIEInputLayerIterator* & pInputLayers
) = 0;
```

Parameters

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Output AcMapIEInputLayerIterator iterator over the input layers. The caller is responsible for deleting this object.</td>
</tr>
</tbody>
</table>

Returns

Returns AcMapIE::ErrCode kErr_OK if successful. Returns AcMapIE::ErrCode kErr_Fail if no data was found. Returns AcMapIE::ErrCode kErr_Fail if the process failed for some other reason.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determines which location window is used to filter incoming entities.

```cpp
virtual void LocationWindowAndOptions(
    double& dXMin,
    double& dXMax,
    double& dYMin,
    double& dYMax,
    AcMapIE::LocationOption& locOption
) const = 0;
```

**Parameters**  
- **dXMin**: Output minimum X value.  
- **dXMax**: Output maximum X value.  
- **dYMin**: Output minimum Y value.  
- **dYMax**: Output maximum Y value.  
- **locOption**: Output LocationOption location name.

**Returns**  
Returns nothing.

**Remarks**

If the value returned in `locOption` is `AcMapIE::kDontUse`, then the window is invalid and its minimum and maximum points are set to 0.0; otherwise, the coordinates are set to the WCS (World Coordinate System) values for the screen size (if `locOption` is `AcMapIE::kUseScreenBoundary), or to the coordinates of the user-defined location window (if `locOption` is `AcMapIE::kUseLocationWindow`.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Removes a reactor from this importer.

```cpp
def virtual AcMapIE::ErrCode RemoveReactor(
    AcMapIEImportReactor * pReactor
) = 0;
```

**Parameters**

- **pReactor**
  - **Input:** `AcMapIEImportReactor`
  - import reactor to remove.

**Returns**

Returns `AcMapIE::ErrCode kErr_OK` if successful. Returns `AcMapIE::ErrCode kErr_Invalid` if the reactor does not already exist. Returns `AcMapIE::ErrCode kErr_Fail` if the process failed for some other reason.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Enables or disables classified-property audits after import.

```cpp
virtual void SetAuditClassifiedAfterImport(
    bool bAudit
) = 0;
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bAudit</td>
<td>Input true to turn auditing on, or false to turn it off.</td>
</tr>
</tbody>
</table>

**Returns**

Returns nothing.

**Remarks**

An audit determines whether each classified-property value is out of range and, if so, sets it to its default value.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the driver options. Typically, you should retrieve the current driver options from DriverOptions() and make changes as needed by using this function.

```cpp
virtual AcMapIE::ErrCode SetDriverOptions(
    AcMapIENamespaceValueIterator* &pNameValuePair,
    bool &bSchemaHasChanged
) = 0;
```

### Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pNameValuePair</td>
<td>Input driver options, as AcMapIENamespaceValueIterator name-value pairs.</td>
</tr>
<tr>
<td>bSchemaHasChanged</td>
<td>Output true if the schema has changed; otherwise, false.</td>
</tr>
</tbody>
</table>

### Returns

Returns AcMapIE::ErrCode kErr_OK if successful. Returns AcMapIE::ErrCode kErr_OutOfMemory if system resources are insufficient. Returns AcMapIE::ErrCode kErr_Fail if the process failed for some other reason.

Created with a commercial version of [Doc-O-Matic](https://www.doceasy.com/products/doc-o-matic). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the manner in which polygons are imported: as closed polylines or as polygons.

```cpp
virtual void SetImportPolygonsAsClosedPolylines(
    bool bLines
) = 0;
```

Parameters

- **bLines**: Input true to import polygons as closed polylines, or false to import them as polygons.

Returns

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links

AcMapIEImporter Class, AcMapIEImporter Class
AcMapIEImporter:: SetLocationWindowAndOptions Method
AcMapIEImporter Class | AcMapIEImporter Class

Sets the location window used to filter incoming entities.

```
virtual AcMapIE::ErrCode SetLocationWindowAndOptions(
    double dXMin,
    double dXMax,
    double dYMin,
    double dYMax,
    AcMapIE::LocationOption locOption
) = 0;
```

Parameters Description
---
dXMin Input minimum X value.
dXMax Input maximum X value.
dYMin Input minimum Y value.
dYMax Input maximum Y value.
locOption Input LocationOption location name.

Returns

Returns AcMapIE::ErrCode kErr_OK if successful. Returns AcMapIE::ErrCode kErr_InvalidWindow if window is invalid. Returns AcMapIE::ErrCode kErr_BadParams if the location option is invalid. Returns AcMapIE::ErrCode kErr_Fail if the process failed for some other reason.

Remarks

If locOption is set to AcMapIE::kDontUse, then all window-location values are discarded. This function fails if dXMin > dXMax or dYMin > dYMax.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

virtual ~AcMapIEImportReactor();

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapIEImportReactor Class, AcMapIEImportReactor Class
AcMapIEImportReactor:: AcMapIEImportReactor Constructor
AcMapIEImportReactor Class | AcMapIEImportReactor Class

Constructs an instance of this class.

AcMapIEImportReactor();

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Invoked after an entity is imported.

```cpp
virtual void RecordImported(
    AcDbObjectId oId
);
```

Parameters

- `oId`  
  Input ID of the imported entity.

Returns

Returns nothing.

Remarks

You can use this function to track imported entities.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Invoked before an entity is imported.

```cpp
virtual bool RecordReadyForImport(
    AcDbEntity* pEnt
);
```

**Parameters**
- `pEnt`: Input entity.

**Returns**
- Returns true if the specified entity should be imported; otherwise, returns false.

**Remarks**

The entity is not a const, so you can modify it as needed. The return value indicates whether an entity is ready to import. If this function returns false, the entity is discarded and no more reactors in the chain are called. If this function returns true for all reactors in the chain, the entity is added to the database. Only after an entity is added to the database can the entity can be classified or class properties be added to it.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

```cpp
virtual ~AcMapIEInputLayer();
```

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapIEInputLayer Class, AcMapIEInputLayer Class
AcMapIEInputLayer:: ColumnIterator Method
AcMapIEInputLayer Class | AcMapIEInputLayer Class

Retrieves an iterator over a collection of AcMapIEColumn instances.

virtual AcMapIEColumnIterator* ColumnIterator() const = 0;

Returns

Returns the column iterator. The caller is responsible for deleting this object.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determine whether this import layer is on (and will be imported).

**virtual bool** `ImportFromInputLayerOn()` **const** $= \theta$;

Returns

Returns true if this input layer is on (this layer will be imported); otherwise, returns false (this layer will not be imported).

Remarks

The default is on. In the AutoCAD Map user interface, this setting corresponds to the check box in the Input Layer column in the Import dialog box properties table.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software, please contact us at **support@toolsfactory.com**.
Retrieves the name of this input layer.

```cpp
virtual const ACHAR* Name() const = 0;
```

Returns the name of this input layer. Do not delete this name.

Remarks

This name identifies an input layer uniquely and does not change. In the AutoCAD Map user interface, this setting corresponds to the Input Layer column in the Import dialog box properties table.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Retrieves AutoCAD Map's best guess for the coordinate system that the data to be imported uses.

```c
virtual const ACHAR* OriginalCoordSys() const = 0;
```

Returns

Returns the name of coordinate system, as a Mentor abbreviation.

Remarks

This function is not yet implemented in AutoCAD Map and always returns an empty string.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Turns this input layer on (to be imported) or off (not to be imported).

```cpp
virtual void SetImportFromInputLayerOn(bool bOn)
```

**Parameters**

**Description**

- `bOn`:
  - Input true to turn this input layer on, or false to turn it off.

**Returns**

Returns nothing.

**Remarks**

In the AutoCAD Map user interface, this setting corresponds to the check box in the Input Layer column in the Import dialog box properties table.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets whether block attribute values are brought in from incoming attribute data.

```cpp
virtual AcMapIE::ErrCode SetUseForBlockAttributes(
    bool bGetAttributeFromField
) = 0;
```

**Parameters**
- `bGetAttributeFromField` Input true to import attribute values; otherwise, false.

**Returns**
- Returns `AcMapIE::ErrCode kErr_OK` if successful. Returns `AcMapIE::ErrCode kErr_Invalid` if `bGetAttributeFromField` is true and the drawing has no blocks. Returns `AcMapIE::ErrCode kErr_Fail` if the process failed for some other reason.

**Remarks**

In the AutoCAD Map user interface, this setting corresponds to the Get Attribute Values From Fields check box in the Point Mapping dialog box. Attributes are brought in when the incoming field name matches the attribute name.

Created with a commercial version of [Doc-O-Matic](https://www.toolsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the coordinate system that incoming data is transformed from.

```cpp
virtual const ACHAR* TargetCoordSys() const = 0;
```

Returns the name of the coordinate system, as a Mentor abbreviation.

Remarks

In the AutoCAD Map user interface, this setting corresponds to the Coordinate System column in the Import dialog box properties table.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determines whether block attribute values are brought in from incoming attribute data.

```
virtual bool UseForBlockAttributes() const = 0;
```

Returns true if attribute values are imported; otherwise, returns false.

Remarks

In the AutoCAD Map user interface, this setting corresponds to the Get Attribute Values From Fields check box in the Point Mapping dialog box.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

virtual ~AcMapIEInputLayerIterator();
Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determines whether the iterator has reached the end of the collection.

```cpp
virtual bool Done() = 0;
```

Returns true if the iterator has reached the end of the collection; otherwise, returns false.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the current element in the iteration.

```cpp
virtual bool Get(
    AcMapIEInputLayer*& pInputLayer
) = 0;
```

**Parameters**
- `pInputLayer` Output pointer to the current AcMapIEInputLayer.

**Returns**

Returns true if the current element is valid, or false if the iteration has no more elements to retrieve.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links

AcMapIEInputLayerIterator Class, AcMapIEInputLayerIterator Class

AcMapIEInputLayerIterator:: Rewind Method

AcMapIEInputLayerIterator Class | AcMapIEInputLayerIterator Class

Moves to the first element in the iteration.

**virtual void** Rewind() = 0;

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Advances to the next element in the iteration.

```cpp
virtual void Step() = 0;
```

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
AcMapIENamValueIterator Class, AcMapIENamValueIterator Class
AcMapIENamValueIterator:: ~AcMapIENamValueIterator Destructor
AcMapIENamValueIterator Class | AcMapIENamValueIterator Class

Destroys an instance of this class.

**virtual ~AcMapIENamValueIterator();**

Returns

Returns nothing.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Removes all elements from the collection.

```cpp
virtual void Clear() = 0;
```

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determines whether the iterator has reached the end of the collection.

```cpp
virtual bool Done() = 0;
```

Returns true if the iterator has reached the end of the collection; otherwise, returns false.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Moves to the first element in the iteration.

```cpp
virtual void Rewind() = 0;
```

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Advances to the next element in the iteration.

```cpp
virtual void Step() = 0;
```

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
This enum represents the list of defined Map Book errors.

```c
enum E MbStatus {
    kMapMbOk = 0,
    kInvalidTemplateName = 1,
    kInvalidLayoutName,
    kTileViewportNotFound,
    kKeyViewportNotFound,
    kLegendViewportNotFound,
    kAdjacentArrowNotFound,
    kTitleBlockNotFound,
    kMultipleTileViewports,
    kMultipleKeyViewports,
    kMultipleLegendViewports,
    kNoDMLegend,
    kInvalidTitleBlockRedefinition,
    kInvalidAdjacentArrowBlockRedefinition,
    kDrawingHasNotBeenSaved = 50,
    kDrawingHasBeenModified,
    kSourceMapNotFound,
    kSourceMapScaleNotFound,
    kSourceMapCannotReset,
    kInvalidSheetSetName = 100,
    kSheetSetNotFound,
    kSheetSetOutOfSynch,
    kTileGenGridRowsOrColsAreZero = 200,
    kTileGenGridVariableTileSizeNotSupported,
    kTileGenGridEmptyExtents,
    kTileGenIncorrectTileSettings,
    kTileNameGenSettingsNull = 300,
    kTileNameGenExpressionFailed,
    kTileNameGenSchemeUnknown,
    kTileNameGenSettingsValueFail,
    kTileNameGenSetTileNameFail,
    kTileNameGenNotGridTiling,
    kTileRelationshipBuilder = 400,
    kTileRelBlderTileAdjacencyFail,
    kTileRelBlderInvalidTable,
    kTileRelBlderUnknownRegion,
    kTileRelBlderRegionFail,
    kXMLFiler = 500,
    kXMLFilerUnrecognizedSettings,
};
```
kTileGeneratorSettings = 600,
kTileValueNotInitialized,
kTileNameGeneratorSettings = 700,
kTileNameGenBadAlphaSequence
};
File

AcMapMbError.h

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
KMapMbClassErrors represents a type of errors associated with Map Book application.

```cpp
enum ErrorClass {
    kMapMbClassErrors = 0xB9104053
};
```

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

```cpp
~AcMapMbError(
    void
);
```

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Constructs an instance of this class.

```cpp
AcMapMbError()
void
); 
Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Checks if the status code indicates an error.

```cpp
static bool IsError(
    EMbStatus es
);
```

**Parameters**

<table>
<thead>
<tr>
<th>es</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Input error status</td>
</tr>
</tbody>
</table>

**Returns**

Returns true in the case if es indicates an error, false otherwise.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Checks if the status code indicates success.

```cpp
static bool IsSuccess(
    EMbStatus es
);
```

Parameters

- `es` : Input error status

Returns

Returns true in the case if `es` indicates success, false otherwise.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Checks if the status code indicates a warning.

```cpp
static bool IsWarning(
    EMbStatus es
);
```

**Parameters**
- `es` Input error status

**Returns**

Returns true in the case if `es` indicates a warning, false otherwise.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Loads an error message associated with the error code from resource file.

```c
static const ACHAR* Message(EMbStatus es);
```

**Parameters**

- `es` : Input error status

**Returns**

Returns a pointer to a static buffer with the message loaded.

**Remarks**

If there is no specific message, "Unexpected error" is returned. This method is using internal static buffer to load the resource string; it is recommended to either use the returned string immediately or create a local copy.

Created with a commercial version of [Doc-O-Matic](#). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbError Class, AcMapMbError Class
AcMapMbError:: PushErrorMessage Method
AcMapMbError Class | AcMapMbError Class

Pushes an error entry to the error stack.

static void PushErrorMessage(
    EMbStatus es
);

Parameters
es Description
Input error status

Returns
Nothing.

Remarks
The entry includes kMapMbClassErrors as error type, es as an error code and the error message associated with es if any.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Bit flags indicating to a reactor what kind of modification is taking place.

```c
enum EModificationType {
    kMapBookGeneral = 0,
    kMapBookTileSetModified = 1,
    kMapBookRegenerated = 2,
    kMapBookModifyUndone = 4,
    kMapBookTileNameChanged = 8,
    kMapBookTileAdjacencyModified = 16
};
```

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

```
virtual ~AcMapMbMapBook();
```

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Constructs an instance of this class.

```
AcMapMbMapBook();
```

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbMapBook Class, AcMapMbMapBook Class
AcMapMbMapBook:: dwgInFields Method
AcMapMbMapBook Class | AcMapMbMapBook Class

Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dwgInFields(
    AcDbDwgFiler* pFiler
);
```

**Parameters**

- **pFiler**: Input filer to use to read the object's data.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Let's this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

```
virtual Acad::ErrorStatus dwgOutFields(
    AcDbDwgFiler* pFiler
) const;
```

**Parameters**

- **pFiler**

  Input filer to use to write the object's data.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Let this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dxfInFields(
    AcDbDxfFiler* pFiler
);
```

**Parameters**
- **Description**
- **pFiler**
  - Input filer to use to read the object's data.

**Returns**
- Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapMbMapBook Class, AcMapMbMapBook Class
AcMapMbMapBook:: dxfOutFields Method
AcMapMbMapBook Class | AcMapMbMapBook Class

Lets this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dxfOutFields(
    AcDbDxfFiler* pFiler
) const;
```

Parameters Description
pFiler Input filer to use to write the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrives the north direction.

```cpp
Acad::ErrorStatus GetNorthDirection(
    double& dAngle
) const;
```

**Parameters**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Output the angle between the vertical and the north direction in degrees; the possible values can be in the interval [0, 360).</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

**Remarks**

This is a Map Book property stored with the Sheet or Sheet Subset; if a Sheet Set file has been deleted, it resets the value from the projection or to 0.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Gets the Map Sheet Layout Settings used to generate the Map Book.

```cpp
Acad::ErrorStatus GetSettings(
    const AcMapMbMapSheetLayoutSettings* &pSettings)
) const;
```

**Parameters**

- `pSettings`: Output the Map Sheet Layout Settings settings used to generate the Map Book.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Gets the Tile Generator settings used to generate the Map Book.

```
Acad::ErrorStatus GetSettings(
    const AcMapMbTileGeneratorSettings* & pSettings
) const;
```

**Parameters**
- `pSettings`: Output the Tile Generator settings used to generate the Map Book.

**Returns**
- Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Gets the Tile Name Generator settings used to generate the Map Book.

```cpp
Acad::ErrorStatus GetSettings(
    const AcMapMbTileNameGeneratorSettings*& pSettings
) const;
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pSettings</td>
<td>Output the Tile Name Generator settings used to generate the Map Book.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrives the number of tiles.

```cpp
Acad::ErrorStatus GetTileCount(
    unsigned int& usTileCount
) const;
```

**Parameters**

- `usTileCount`: Output tile number.

**Returns**

- Returns Acad::eOk if successful; otherwise, returns a different error code.

**Remarks**

This is a Map Book property stored with the Sheet or Sheet Subset as TileCount custom property; if a Sheet Set file has been deleted, it retrieves the actual number of tiles in a tile set.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbMapBook Class, AcMapMbMapBook Class
AcMapMbMapBook:: Implementation Method
AcMapMbMapBook Class | AcMapMbMapBook Class

Returns the implementation object.

**virtual** AcMapMbImpMapBook* Implementation();

Returns the implementation, or NULL if unsuccessful.

Remarks

This function is for internal use only; do not call it directly.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the name of this Map Book.

```cpp
const ACHAR* Name() const;
```

Returns

Returns the MapBook Name. The function returns NULL if the name is empty. The returned name should not be deleted.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the north direction.

```cpp
Acad::ErrorStatus SetNorthDirection(
    double dAngle
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dAngle</td>
<td>Input the angle between the vertical and the north</td>
</tr>
<tr>
<td></td>
<td>direction in degrees; the allowed values can be from</td>
</tr>
<tr>
<td></td>
<td>0 up to but not equal to 360.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; Acad::eInvalidInput if the value is not in the interval [0, 360) otherwise, returns a different error code.

Remarks

This is a Map Book property stored with the Sheet or Sheet Subset as NorthDirection custom property; this way it can be seen and even edited in Sheet Set Manager. 0 indicates aligning with the Y axis pointing up, increasing in the clockwise direction.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Retrieves the file name of the Map Sheet Set associated with this MapBook.

```cpp
const ACHAR* SheetSetFileName() const;
```

Returns Map Sheet Set file name.

Created with a commercial version of [Doc-O-Matic](http://www/toolsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Retrieves the Subset name within the Map Sheet Set if any.

```cpp
const ACHAR* SheetSetSubsetName() const;
```

Returns

Output Subset name within the Map Sheet Set. The Subset name includes the full path where a slash "/" is used as a delimiter, for example: "Root/Nested_Subset/Map_Book_Subset". If a Map Book is stored as a complete Sheet Set, NULL is returned.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Shows/hides grid objects (polylines) associated with the Tile Set.

```cpp
Acad::ErrorStatus ShowGridObjects(
    bool bToBeShown
)
const;
```

**Parameters**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Output the desired visibility status, if false object will be hidden.</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; returns Acad::eNotApplicable in the case of grid objects being created manually, otherwise, returns a different error code.

**Remarks**

This functionality is applicable only to grid objects generated automatically; polylines created manually are not affected. By default, grid objects become visible when a Map Book becomes current.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Invoked from within close() before the close actually occurs. The default implementation of this function returns Acad::eOk. See also subClose() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus subClose();
```

Returns

Returns Acad::eOk if successful.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Invoked from within erase() before the erase actually occurs. See also subErase() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus subErase(
    Adesk::Boolean erasing
);
```

**Parameters**

- **erasing**
  - Description: Input erasing argument that was passed to the erase() function call that triggered this subErase() call.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbMapBook Class, AcMapMbMapBook Class
AcMapMbMapBook:: TileSet Method
AcMapMbMapBook Class | AcMapMbMapBook Class

Gets the Tile Set of the Map Book.

AcMapMbTileSet* TileSet() const;

Returns

Returns the Map Book Tile Set. Returns NULL in the case of failure.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Grants control of deep clone operations to the object. In the default implementation, the object is cloned and appended to the owner object pOwnerObject. See also wblockClone() in the AutoCAD ObjectARX Developer’s Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus wblockClone(
    AcRxObject* pOwnerObject,
    AcDbObject*& pClonedObject,
    AcDbIdMapping& idMap,
    Adesk::Boolean isPrimary = true
) const;
```

Parameters

- pOwnerObject: Input object to append the clones to.
- pClonedObject: Output the cloned object, or NULL if not cloned.
- idMap: Input current ID map.
- isPrimary: Input true if this object is primary, or false if it is owned.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.
Links
AcMapMbMapBookManager Class, AcMapMbMapBookManager Class
AcMapMbMapBookManager:: TILEID Nested Type
AcMapMbMapBookManager Class | AcMapMbMapBookManager Class

Defines a type for ID's of map book tiles.

typedef unsigned long TILEID;
Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Defines a direction for selecting adjacent tiles.

```cpp
enum EAdjacentTile {
    kAdjacentTop = 0,
    kAdjacentTopRight,
    kAdjacentRight,
    kAdjacentBottomRight,
    kAdjacentBottom,
    kAdjacentBottomLeft,
    kAdjacentLeft,
    kAdjacentTopLeft
};
```

File

AcMapMbMapBookManager.h

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Add a map book creation reactor to monitor map book creation process such as end of tiling and sheet creation.

```cpp
static bool AddMapBookCreationReactor(
    AcDbDatabase* pDatabase,
    AcMapMbMapBookCreationReactor* pReactor
);
```

**Parameters**

- **pDatabase**: Input database.
- **pReactor**: Creates a map book creation reactor.

**Returns**

Returns true if successful.

**Remarks**

This reactor is not currently exposed through the public API and is used for internal purposes only.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links

AcMapMbMapBookManager Class, AcMapMbMapBookManager Class
AcMapMbMapBookManager:: AddMapBookReactor Method
AcMapMbMapBookManager Class | AcMapMbMapBookManager Class

Adds a map-book-management reactor to monitor map book management
related activities.

```cpp
static bool AddMapBookReactor(
    AcDbDatabase* pDatabase,
    AcMapMbMapBookReactor* pReactor
);
```

Parameters Description
pDatabase Input database.
pReactor Input reactor.

Returns
Returns true if successful.

Created with a commercial version of Doc-O-Matic. In order to make this
message disappear you need to register this software. If you have problems
registering this software please contact us at support@toolsfactory.com.
Retrieves the current Map Book.

\[
\text{static Acad::ErrorStatus GetCurrent(}
\text{   AcDbObjectId& Id,}
\text{   AcDbDatabase* pDatabase)}
\]

Parameters

- **Id**: Output ID of the current Map Book. Null Id corresponds no current Map Book state.

- **pDatabase**: Input pointer to a drawing database.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves an Id of a Map Book associated with the given layout if any.

```cpp
static Acad::ErrorStatus GetMapBookForLayout(
    Acad::ObjectId& mapBookId,
    const Acad::ObjectId& layoutId
);
```

### Parameters

- **mapBookId**: Output Id of the Map Book. It is assigned a value only in the case the specified layout is a Map Book layout.
- **layoutId**: Input Id of the AutoCAD Layout object.

### Returns

Returns Acad::eOk if this is a Map Book layout, Acad::eKeyNotFound if this is not a Map Book layout; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Determines whether a project has map book data.

```cpp
static bool HasMapBookData(
    AcDbDatabase * pDatabase
);
```

Parameters  Description
pDatabase    Input database.

Returns

Returns true if the project has map book data.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns the implementation object.

```c
static AcMapMbImpMapBookManager* mbMapBookManagerImp();
```

Returns the implementation, or NULL if unsuccessful.

Remarks

This function is for internal use only; do not call it directly.

Created with a commercial version of [Doc-O-Matic](https://www.doc-omatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbMapBookManager Class, AcMapMbMapBookManager Class
AcMapMbMapBookManager:: NewIterator Method
AcMapMbMapBookManager Class | AcMapMbMapBookManager Class

Returns a new map book dictionary iterator.

```c++
static AcDbDictionaryIterator* NewIterator(
    AcDbDatabase* pDatabase
);
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pDatabase</td>
<td>Input pointer to a drawing database. If the input is NULL this function returns NULL.</td>
</tr>
</tbody>
</table>

Returns

Returns iterator if Map Book dictionary exists; otherwise, returns NULL.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Removes the map book with the specified ID.

\[
\text{static Acad::ErrorStatus Remove(} \\
\quad \text{const AcDbObjectId} & \text{Id})
\]

Parameters

- **Id**
  - Input ID of the map book to remove.

Returns

- Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Removes a map book creation reactor.

```cpp
static bool RemoveMapBookCreationReactor(
    AcDbDatabase* pDatabase,
    AcMapMbMapBookCreationReactor* pReactor
);
```

**Parameters**

- `pDatabase` Input database.
- `pReactor` Input reactor.

**Returns**

Returns true if successful.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Removes a map book reactor from a database.

```cpp
static bool RemoveMapBookReactor(
    AcDbDatabase* pDatabase,
    AcMapMbMapBookReactor* pReactor
);
```

Parameters

- **pDatabase**
  - Input database.
- **pReactor**
  - Returns

Returns

Returns true if successful.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the current Map Book.

```
static Acad::ErrorStatus SetCurrent(
    const AcDbObjectId& Id,
    AcDbDatabase* pDatabase = NULL)
```

**Parameters**
- **Id**: Input ID of the Map Book to set current. Null Id corresponds to current Map Book state. Id should belong to the specified database. Even if a source Display Map or Map scale are missing or a system cannot find the Sheet Set set file associated with the specified Map Book the operation succeeds, however an appropriate error code is returned in these cases.
- **pDatabase**: Input pointer to a drawing database

**Returns**
- Returns Acad::eOk if successful; returns Acad::eKeyNotFound if DM Map or scale is missing returns Acad::eFileNotFound if proper sheet set not found; returns Acad::eInProcessOfCommitting in the case new Map Book required switching Display Map but there are uncomitted edits; otherwise, returns a different error code.

**Remarks**

In the case a Map Book has a Display Map as a source, resets the proper current Display Map. In the case there are uncommitted edits, setting current Display Map and respectively current Map Book fails.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the Display Management environment such as DM Map and Scale associated with the Map Book.

```cpp
static Acad::ErrorStatus SetMapBookDMEnvironment(
    const AcMapMbTileGeneratorSettings* pTileGeneratorSettings);
```

### Parameters

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>pTileGeneratorSettings</strong>  Input settings object for MapBookSource,</td>
</tr>
<tr>
<td>DMMapName, and DmScaleFactor</td>
</tr>
</tbody>
</table>

### Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

### Remarks

Uses the tile generator settings object for MapBookSource DMMapName, and DmScaleFactor to set the appropriate environment. These may not be consistent, in which case an error message may be returned and/or messages may be pushed onto the AutoCAD Map Error Stack. The user should check this stack after each call.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the environment associated with the Map Book.

```cpp
static Acad::ErrorStatus SetMapBookEnvironment(
    const AcMapMbMapBook* pMapBook
);
```

**Parameters**
- `pMapBook` Input pointer to an existing MapBook.

**Returns**
- Returns Acad::eOk if successful; otherwise, returns a different error code.

**Remarks**

This includes opening the related Sheet Set, setting current DM Map and scale etc.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Invoked when a new map book is appended or after a reactor is attached to a project that has map book(s) defined.

```cpp
virtual void MapBookAppended(
    AcMapMbMapBook* pMapBook
);
```

Parameters | Description
--- | ---
pMapBook | Input pointer to the affected MapBook.

Returns

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Invoked when a map book is detached or erase operation is undone.

```cpp
virtual void MapBookErased(
    AcMapMbMapBook* pMapBook,
    bool bErased
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pMapBook</td>
<td>Input pointer to the affected MapBook.</td>
</tr>
<tr>
<td>bErased</td>
<td>Input parameter indicating the kind of operation. Set to true if the MapBook has been erased, and false if the MapBook erase has been reversed.</td>
</tr>
</tbody>
</table>

Returns

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapMbMapBookReactor Class,  AcMapMbMapBookReactor Class

AcMapMbMapBookReactor::~ MapBookModified Method
AcMapMbMapBookReactor Class  |  AcMapMbMapBookReactor Class

Invoked when a map book is detached or erase operation is undone.

```cpp
virtual void MapBookModified(
    AcMapMbMapBook* pMapBook,
    unsigned long usFlag
);
```

Parameters | Description
--- | ---
pMapBook | Input pointer to the affected MapBook.
usFlag | Input bit set where each bit indicates the specific change. The bits are defined by EModificationType enum in AcMapMbMapBook.h.

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Invoked when a current Map Book gets set or at the very begining of the Map session.

```cpp
virtual void MapBookSetCurrent(
    const AcDbObjectId& oldMapBookId,
    const AcDbObjectId& newMapBookId
);
```

Parameters

- `oldMapBookId` Input old current Map Book Id.
- `newMapBookId` Input new current Map Book Id.

Returns

- Returns nothing.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Invoked when a map book tile is modified.

```cpp
virtual void MapBookTileModified(
    const AcMapMbTile* pTile,
    unsigned long usFlag
);
```

**Parameters**

- **pTile**
  Input pointer to the affected Tile.

- **usFlag**
  Input bit set where each bit indicates the specific change. The bits are defined by EModificationType enum in AcMapMbMapBook.h.

**Returns**

Returns nothing.

**Remarks**

Notifications occur when the Tile Name is modified, when Adjacency is modified and when the tile is disabled.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Invoked when a map book tile is about to be erased.

```cpp
virtual void MapBookTileWillBeErased(
        const AcMapMbTile* pTile)
```

**Parameters**

- **pTile**
  - Description: Input pointer to the affected Tile.

**Returns**

Returns nothing.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Invoked when a map book tree node is modified.

```cpp
virtual void MapBookTreeNodeModified(
    const AcMapMbTileSet::AcMapTreeNode* pTile,
    unsigned long usFlag
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>usFlag</td>
</tr>
<tr>
<td>pNode</td>
</tr>
</tbody>
</table>

**Returns**

Returns nothing.

**Remarks**

Notifications occur when the tree node name is modified.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Invoked when a map book is about to be erased or erase operation is about to be undone.

```cpp
virtual void MapBookWillBeErased(
    AcMapMbMapBook* pMapBook,
    bool bErased
);
```

**Parameters**

- `pMapBook` Input pointer to the affected MapBook.
- `bErased` Input parameter indicating the kind of operation. Set to true if the MapBook is being erased, and false if the MapBook erase is being reversed.

**Returns**

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Enumerates supported types of a Map Key.

```cpp
enum EMapKeyType {
    kMapKeyNone,
    kMapKeyCurrentDrawing,
    kMapKeyExternalDrawing,
    kMapKeyLinkedDrawing
};
```

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Enumerates supported types of legend

```cpp
enum EMapLegendType {
    kMapLegendNone,      
    kMapLegendCurrentMap,
    kMapLegendBox
};
```

File

AcMapMbMapSheetLayoutSettings.h

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

```
virtual ~AcMapMbMapSheetLayoutSettings();
```

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapMbMapSheetLayoutSettings Class, AcMapMbMapSheetLayoutSettings Class
AcMapMbMapSheetLayoutSettings::
AcMapMbMapSheetLayoutSettings Constructor
AcMapMbMapSheetLayoutSettings Class | AcMapMbMapSheetLayoutSettings Class

Constructs an instance of this class.

AcMapMbMapSheetLayoutSettings();
Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbMapSheetLayoutSettings Class, AcMapMbMapSheetLayoutSettings Class
AcMapMbMapSheetLayoutSettings::: AdjacentArrowBlockName Method
AcMapMbMapSheetLayoutSettings Class | AcMapMbMapSheetLayoutSettings Class

Retrieves an adjacent arrow block name.

```cpp
const ACHAR* AdjacentArrowBlockName() const;
```

Returns

Returns an adjacent arrow block name or NULL if has not been specified.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Allocates an instance of Map Sheet Template class.

```
virtual AcMapMbMapSheetTemplate* AllocateMapSheetTemplate() const;
```

Returns pointer to the allocated `AcMapMbMapSheetTemplate` object.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Clones the object. See also clone() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual AcRxObject* clone() const;
```

Returns a pointer to the clone, or NULL if cloning was unsuccessful.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links

AcMapMbMapSheetLayoutSettings Class, AcMapMbMapSheetLayoutSettings Class

AcMapMbMapSheetLayoutSettings:: copyFrom Method

AcMapMbMapSheetLayoutSettings Class | AcMapMbMapSheetLayoutSettings Class

Copies the contents of an object into the messaged object, if feasible. See also copyFrom() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

```
virtual Acad::ErrorStatus copyFrom(
    const AcRxObject* other
);
```

Parameters

- `other` Input pointer to the object to copy from.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbMapSheetLayoutSettings Class, AcMapMbMapSheetLayoutSettings Class
AcMapMbMapSheetLayoutSettings:: dwgInFields Method
AcMapMbMapSheetLayoutSettings Class | AcMapMbMapSheetLayoutSettings Class

Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

```
virtual Acad::ErrorStatus dwgInFields(
    AcDbDwgFiler* pFiler
);
```

Parameters Description
pFiler Input filer to use to read the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbMapSheetLayoutSettings Class, AcMapMbMapSheetLayoutSettings Class
AcMapMbMapSheetLayoutSettings:: dwgOutFields Method
AcMapMbMapSheetLayoutSettings Class | AcMapMbMapSheetLayoutSettings Class

Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

virtual Acad::ErrorStatus dwgOutFields(
    AcDbDwgFiler* pFiler
) const;

Parameters Description
pFiler Input filer to use to write the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links

AcMapMbMapSheetLayoutSettings Class, AcMapMbMapSheetLayoutSettings Class
AcMapMbMapSheetLayoutSettings:: dxfInFields Method
AcMapMbMapSheetLayoutSettings Class | AcMapMbMapSheetLayoutSettings Class

Lets this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dxfInFields(
    AcDbDxfFiler* pFiler
);
```

Parameters | Description
--- | ---
pFiler | Input filer to use to read the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbMapSheetLayoutSettings Class, AcMapMbMapSheetLayoutSettings Class
AcMapMbMapSheetLayoutSettings:: dxfOutFields Method
AcMapMbMapSheetLayoutSettings Class | AcMapMbMapSheetLayoutSettings Class

Lets this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

**virtual** Acad::ErrorStatus dxfOutFields(
    AcDbDxfFiler* pFiler
) const;

Parameters Description
pFiler Input filer to use to write the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://toolsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves a Legend box.

```cpp
Acad::ErrorStatus GetLegendBox(
    AcDbExtents& extents
) const;
```

**Parameters**

- `extents` Output bounding box.

**Returns**

- Returns Acad::eOk if successful; Acad::eInvalidExtents if the legend box is invalid or has not been initialized, otherwise, returns a different error code.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns the implementation object.

\texttt{virtual AcMapMbImpMapSheetLayoutSettings* Implementation();}

Returns the implementation, or NULL if unsuccessful.

Remarks

This function is for internal use only; do not call it directly.

Created with a commercial version of \texttt{Doc-O-Matic}. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at \texttt{support@toolsfactory.com}.
Retrieves the flag to include a adjacent arrows to a Map Sheet layout.

```cpp
bool IncludeAdjacentArrows() const;
```

Returns true if a Map Sheet should include Adjacent Arrows; otherwise, returns false.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbMapSheetLayoutSettings Class, AcMapMbMapSheetLayoutSettings Class
AcMapMbMapSheetLayoutSettings:: IncludeTitleBlock Method
AcMapMbMapSheetLayoutSettings Class | AcMapMbMapSheetLayoutSettings Class

Retrieves the flag to include a title block to a Map Sheet layout.

\texttt{bool IncludeTitleBlock() const;}

Returns

Returns true if a Map Sheet should include a Title block; otherwise, returns false.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbMapSheetLayoutSettings Class, AcMapMbMapSheetLayoutSettings Class
AcMapMbMapSheetLayoutSettings:: LegendType Method
AcMapMbMapSheetLayoutSettings Class | AcMapMbMapSheetLayoutSettings Class

Retrieves a Legend type.

EMapLegendType LegendType() const;

Returns

Returns Legend Type.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves a drawing file name for the External Drawing Map Key.

```c
const ACHAR* MapKeyExternalFileName() const;
```

Returns file name. The name specifies a complete path name with the .DWT or .DWG extension. Returns the empty string if this path has not been set.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves a pattern to select layers visible within a current drawing Map Key.

```c
const ACHAR* MapKeyLayerPattern() const;
```

Returns pattern to select layers visible within a current drawing Map Key or NULL if has not been specified.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links  
AcMapMbMapSheetLayoutSettings Class, AcMapMbMapSheetLayoutSettings Class  
AcMapMbMapSheetLayoutSettings:: MapKeyType Method  
AcMapMbMapSheetLayoutSettings Class | AcMapMbMapSheetLayoutSettings Class  

Retrieves a Map Key type.

EMapKeyType MapKeyType() const;

Returns  

Returns Map Key Type. Returns kMapKeyNone as default.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the flag to include adjacent arrows to a Map Sheet layout.

```
Acad::ErrorStatus SetIncludeAdjacentArrows(
    bool bIsOn
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bIsOn</td>
</tr>
</tbody>
</table>

**Description**

- **bIsOn**
  - Input whether to include adjacent arrows. If true adjacent arrows get included in a layout; otherwise Map Sheet layout will not include adjacent arrows. The name of the adjacent arrow block gets set via `SetAdjacentArrowBlockName()` method.

**Returns**

- Returns `Acad::eOk` if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the flag to include a title block to a Map Sheet layout.

```cpp
Acad::ErrorStatus SetIncludeTitleBlock(
    bool bIsOn
);
```

### Parameters

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input whether to include a title block. If true a title block gets included in a layout; otherwise Map Sheet layout will not include Title Block object. The name of the title block has to be set via SetTitleBlockName() method in order for SetIncludeTitleBlock() to succeed.</td>
</tr>
</tbody>
</table>

### Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets a Legend box.

```cpp
Acad::ErrorStatus SetLegendBox(
    const AcDbExtents& extents
);
```

Parameters

- `extents` Input bounding box.

Returns

Returns Acad::eOk if successful; Acad::eInvalidExtents if the legend box is invalid, otherwise, returns a different error code.

Remarks

In the case of kMapLegendBox type the legend viewport is zoomed to the specified area. Use this area to locate custom legend objects.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets a Legend type.

```
Acad::ErrorStatus SetLegendType(
    EMapLegendType legendType
);
```

**Parameters**

- `legendType`: Input type of legend. In order to specify a `kMapLegendBox`, a legend box has to be specified via the `SetlegendBox()` method.

**Returns**

Returns `Acad::eOk` if successful; `Acad::eInvalidExtents` if the legend box has not being specified yet, otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.doc-o-matic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets a Map Key type.

```cpp
Acad::ErrorStatus SetMapKeyType(
    EMapKeyType kMapKeyType
);
```

**Parameters**
- `kMapKeyType`: Input type of Map Key.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Retrieves a template file name.

```cpp
const ACHAR* TemplateFileName() const;
```

Returns file name. The name specifies a complete path name with the .DWT or .DWG extension. If this has not been set, the function returns an empty string.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves a template layout name.

```cpp
const ACHAR* TemplateLayoutName() const;
```

Returns layout name. If this has not been set, the function returns an empty string.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves a title block name.

```cpp
const ACHAR* TitleBlockName() const;
```

Returns a title block name or NULL if has not been specified.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@tooltfactory.com.
Defines template elements.

```cpp
text
enum ETemplateElementType {
    kMapView,
    kMapKey,
    kMapLegend,
    kAdjacentArrow,
    kTitleBlock
};
```

File

AcMapMbMapSheetTemplate.h

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

```
virtual ~AcMapMbMapSheetTemplate();
```

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Constructs an instance of this class.

AcMapMbMapSheetTemplate();

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbMapSheetTemplate Class, AcMapMbMapSheetTemplate Class
AcMapMbMapSheetTemplate:: Close Method
AcMapMbMapSheetTemplate Class | AcMapMbMapSheetTemplate Class

Closes the template drawing file.

Acad::ErrorStatus Close();

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the drawing database object associated with the Map Sheet template.

```
AcDbDatabase* DwgDatabase() const;
```

Returns a pointer to a valid drawing database object if successful; otherwise, returns NULL.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Gets the adjacent arrow Id for the specified direction from the layout specified by the settings object.

```cpp
virtual Acad::ErrorStatus GetAdjacentArrow(
    AcDbObjectId& adjacentArrowId,
    AcMapMbMapBookManager::EAdjacentTile direction)
) const;
```

Parameters

- adjacentArrowId: Output adjacent arrow template element.
- direction: Input direction enum indicating which adjacent tile.

Returns

Returns Acad::eOk if successful; Acad::eKeyNotFound in the case the tile does not have adjacent tile in the specified direction, otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Gets the adjacent arrow Ids for layout specified by the settings object.

```c++
virtual Acad::ErrorStatus GetAdjacentArrows(
    AcDbObjectIdArray& aIds
) const;
```

### Parameters

- **aIds**
  - Description: Output Ids of the arrow insert objects in the layout.

### Returns

- Returns Acad::eOk if successful; Acad::eKeyNotFound in the case the layout does not have adjacent arrow placeholder, otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Gets adjacency direction from the adjacent arrow placeholder.

```cpp
static Acad::ErrorStatus GetArrowDirection(
    AcMapMbMapBookManager::EAdjacentTile& adjDirection,
    const AcDbObjectId& objectId
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>adjDirection</td>
<td>Output adjacency direction. Input Id of the adjacent arrow placeholder that is an insert with the attribute. The value of the attribute should be one of the following nonlocalizable strings &quot;TOP&quot; &quot;TOPRIGHT&quot; &quot;RIGHT&quot; &quot;BOTTOMRIGHT&quot; &quot;BOTTOM&quot; &quot;BOTTOMLEFT&quot; &quot;LEFT&quot; &quot;TOPLEFT&quot;</td>
</tr>
<tr>
<td>objectId</td>
<td>Input Id of the adjacent arrow placeholder that is an insert with the attribute. The value of the attribute should be one of the following nonlocalizable strings &quot;TOP&quot; &quot;TOPRIGHT&quot; &quot;RIGHT&quot; &quot;BOTTOMRIGHT&quot; &quot;BOTTOM&quot; &quot;BOTTOMLEFT&quot; &quot;LEFT&quot; &quot;TOPLEFT&quot;</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; Acad::eWrongObjectType in the case the Id is not Id of an adjacent arrow placeholder; Acad::eKeyNotFound if attribute value does not correspond to any of the values above; otherwise, returns a different error code.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the Id of the specified template element.

```cpp
virtual Acad::ErrorStatus GetElement(
    AcDbObjectId& elementId,
    ETemplateElementType elementType
) const;
```

**Parameters**

- `elementId` (output) Element Id.
- `elementType` (input) Element type.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbMapSheetTemplate Class, AcMapMbMapSheetTemplate Class
AcMapMbMapSheetTemplate::GetElement Method
AcMapMbMapSheetTemplate Class | AcMapMbMapSheetTemplate Class

Returns the object id of the object in the specified layout that is marked as an
element of the specified type.

Returns Acad::eOk if successful; Returns Acad::eAmbiguousOutput if there is
more than one element of the specified type. Returns Acad::eKeyNotFound if
there are no elements of the specified type; otherwise, returns a different error
code.

```cpp
static Acad::ErrorStatus GetElement(
    AcDbObjectId& elementId,
    const AcDbObjectId& layoutId,
    ETemplateElementType elementType
);
```

Parameters | Description
--- | ---
elementId | Output Object id of entity previously marked
layoutId | Input Object id of the layout
elementType | Input Enumerated type identifying the element type

Remarks
If there is more than one element of the specified type, the function fails.

Created with a commercial version of Doc-O-Matic. In order to make this
message disappear you need to register this software. If you have problems
registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbMapSheetTemplate Class, AcMapMbMapSheetTemplate Class
AcMapMbMapSheetTemplate::GetElement Method
AcMapMbMapSheetTemplate Class | AcMapMbMapSheetTemplate Class

Returns an array of object ids of the objects in the specified layout that are marked as an element of the specified type.

Returns Acad::eOk if successful; Returns Acad::eKeyNotFound if there are no entities of the specified type; otherwise, returns a different error code.

static Acad::ErrorStatus GetElement(
    AcDbObjectIdArray& aIds,
    const AcDbObjectId& layoutId,
    ETemplateElementType elementType
);

Parameters
Description
aIds
Output Object ids of entity previously marked
layoutId
Input Object id of the layout
elementType
Input Enumerated type identifying the element type

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Gets height and width of tile from the Map viewport.

```cpp
virtual Acad::ErrorStatus GetTileSize(
    double& dWidth,
    double& dHeight
) const;
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dWidth</td>
<td>Output viewport width</td>
</tr>
<tr>
<td>dHeight</td>
<td>Output viewport height</td>
</tr>
</tbody>
</table>

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns the implementation object.

```c
AcMapMbImpMapSheetTemplate* Implementation();
```

Returns the implementation, or NULL if unsuccessful.

Remarks

This function is for internal use only; do not call it directly.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Checks if the object specified by the Id is an element of the specified type.

```cpp
static bool IsElement(
    const AcDbObjectId& Id,
    ETemplateElementType elementType
);
```

**Parameters**

- Id: Input Id of the object.
- elementType: Input element type.

**Returns**

Returns true if the object is an element, false otherwise.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbMapSheetTemplate Class, AcMapMbMapSheetTemplate Class
AcMapMbMapSheetTemplate:: LayoutId Method
AcMapMbMapSheetTemplate Class | AcMapMbMapSheetTemplate Class

Retrieves current template layout Id.

```cpp
Acad::ErrorStatus LayoutId(
    AcDbObjectId& layoutId
) const;
```

**Parameters**

- `layoutId` Output Id of the current template layout specified via the settings.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Marks the selected template object with XData so that it is considered as a template element of the specified type.

Returns Acad::eOk if successful; otherwise, returns a different error code.

```cpp
static Acad::ErrorStatus MarkElement(
    const AcDbObjectId& elementId,
    ETemplateElementType elementType
);
```

**Parameters**

- `elementId`: Input Object id of entity to mark
- `elementType`: Input Enumerated type identifying the element type

**Remarks**

This is a utility that is used to produce Map Book template.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Opens the template drawing file for reading.

```cpp
virtual Acad::ErrorStatus Open(
    const AcMapMbMapSheetLayoutSettings* kpSettings);
```

**Parameters**

<table>
<thead>
<tr>
<th>Description</th>
<th>kpSettings</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input Map Layot template settings.</td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

**Remarks**

The name of the file and the other settings are defined via the `AcMapMbMapSheetLayoutSettings` parameter. In order to change the settings of this object an application has to call Close() method an call Open() with the new settings.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Remove marker XData from the selected template object so that it is considered as a template element any more.

Returns Acad::eOk if successful; otherwise, returns a different error code.

```cpp
static Acad::ErrorStatus UnMarkElement(
    const AcDbObjectId& elementId
);
```

**Parameters**
- `elementId`: Input Object id of entity to clear map book xData

**Remarks**
This is a utility that is used to produce Map Book template.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

```
virtual ~AcMapMbTile();
```

Returns

Returns nothing.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbTile Class, AcMapMbTile Class
AcMapMbTile:: AcMapMbTile Constructor
AcMapMbTile Class | AcMapMbTile Class

Constructs an instance of this class.

AcMapMbTile();

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Constructs an instance of this class.

AcMapMbTile(
    AcMapMbMapBookManager::TILEID tileId,
    const AcDbExtents& extents,
    double dViewWidth,
    double dViewHeight,
    bool bIsEnabled,
    const AcDbObjectId& refObjectId,
    AcMapMbTileSet * pTileSet
);

Parameters

Description

tileId
Input tile id for this tile. Tile Ids should be unique within a drawing.

extents
Input size of the tile in model space units. This value does factor in the scale.

dViewWidth
Input width of the view port in model space units. This value does not include the scale factor.

dViewHeight
Input height of the view port in model space units. This value does not include the scale factor.

refObjectId
Input Object Id of the polygon defining the tile border.

pTileSet
Input tile set pointer to the tile set this tile belongs to.

bEnabled
Input boolean to indicate whether this tile should be enabled or disabled.

Returns

Returns nothing.

Remarks

This constructor should be used by any third party tile generator.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems
registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbTile Class, AcMapMbTile Class
AcMapMbTile:: AcMapMbTile Constructor
AcMapMbTile Class | AcMapMbTile Class

Copies an instance of this class.

```
AcMapMbTile(const AcMapMbTile& other);
```

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbTile Class, AcMapMbTile Class
AcMapMbTile:: AddHyperlink Method
AcMapMbTile Class | AcMapMbTile Class

Assignes a hyperlink to this tile to the specified adjacent arrow object.

Acad::ErrorStatus AddHyperlink(
  const AcDbObjectId& arrowId,
  AcMapMbMapBookManager::EAdjacentTile tileDir
);

Parameters

arrowId
  Input Id of the arrow object; basically it can be any entity - this method does not check entity type.

kTileDir
  Input direction.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns the center of a tile.

**virtual const** AcGePoint3d& Center() **const**;

Returns the center of the tile. If the center has not been set, then the point 0,0,0 will be returned.

Remarks

Z values are not significant for this release, but may have some function in the future.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Gets the adjacent tile of the specified direction for the tile.

```cpp
Acad::ErrorStatus GetAdjacentTile(
    AcMapMbMapBookManager::TILEID& ulTileId,
    AcMapMbMapBookManager::EAdjacentTile tileDir
) const;
```

**Parameters**
- **ulTileId**
  - Output Id of the adjacent tile.
- **kTileDir**
  - Input direction.

**Returns**

Returns Acad::eOk if successful; Acad::eNotApplicable in the case the tile does not have adjacent tile in the specified direction, otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Gets the adjacent tile of the specified direction for the tile.

```
Acad::ErrorStatus GetAdjacentTile(
    AcMapMbTile*& pTile,
    AcMapMbMapBookManager::EAdjacentTile tileDir
) const;
```

**Parameters**
- `pTile`: Output the adjacent tile.
- `kTileDir`: Input direction.

**Returns**
- Returns Acad::eOk if successful; Returns Acad::eNotApplicable in case the tile does not have adjacent tile in the specified direction, otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns the layout object id at the index.

```cpp
virtual AcDbObjectId GetLayoutIdAt(
    int nIndex
) const;
```

Parameters

- `nIndex` Input index of the layout we want.

Returns

Returns the object Id of the if successful; otherwise, returns NullId.

Remarks

Since it is possible for the same tile to appear in several layouts, this will allow the user access to the relevant layouts.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns the size of the tile.

```
virtual Acad::ErrorStatus GetTileSize(
    double& dWidth,
    double& dHeight
) const;
```

Parameters

- **dWidth**: Output width of the tile.
- **dHeight**: Output height of the tile.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Remarks

This also typically indicates the size of the Reference object, however reference objects may have sizes different from the extents. For instance, if the reference objects are used to set the names of the tiles in a mapbook, they may not be the same size, or even rectangular polylines, but the tiles will be calculated to be the correct size based on the template and map scale. For a grid tiling where the sizes of the tiles are calculated, but there may be an overlap specified, this is the size of the grid elements and does not include the overlap. Get and Set ViewPortSize includes the overlap.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Gets the size of the tile as viewed.

```cpp
virtual Acad::ErrorStatus GetViewPortSize(
    double& dWidth,
    double& dHeight
) const;
```

**Parameters**

- `dWidth`: Output width of the viewed size of the tile.
- `dHeight`: Output height of the viewed size of the tile.

**Returns**

Returns `Acad::eOk` if successful; otherwise, returns a different error code.

**Remarks**

This includes the boundary of the tile plus the overlap.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns the implementation object.

\[
\text{AcMapMbImpTile}^* \text{ Implementation() const;}
\]

Returns the implementation, or NULL if unsuccessful.

Remarks

This function is for internal use only; do not call it directly.

Created with a commercial version of \textit{Doc-O-Matic}. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at \texttt{support@toolsfactory.com}. 

Links

\texttt{AcMapMbTile Class, AcMapMbTile Class}

\texttt{AcMapMbTile:: Implementation Method}

\texttt{AcMapMbTile Class | AcMapMbTile Class}
Gets whether the tile is enabled.

```cpp
bool IsEnabled() const;
```

Returns

Returns true if enabled; otherwise, returns false.

Remarks

Disabled tiles typically do not have sheets created for them, but may be a graphical place holder.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns the name of the tile.

```cpp
const ACHAR * Name() const;
```

Returns a const pointer to the string defining the name.

Remarks

Names are set when the tile set is created. Names may be changed.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
This is a notification that every tile except the erased one receives.

`virtual void OnTileErased(`

`AcMapMbMapBookManager::TILEID ulErasedTileId`)

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ulTileId</td>
<td>Input Id of the erased tile.</td>
</tr>
</tbody>
</table>

Returns nothing

Remarks

An object should not subscribe for this notification - it will be sent all the time. The default implementation corrects adjacent information if the erased tile is adjacent to this tile.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns the reference object associated with the tile.

```cpp
virtual AcDbObjectId RefObject() const;
```

Returns AcDbObjectId of the reference object, returns NullId if no reference object has been set.

Remarks

This object will typically be a rectangular closed polyline matching the extents.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the adjacent tile of the specified direction for the tile.

```cpp
Acad::ErrorStatus SetAdjacentTile(
    AcMapMbMapBookManager::TILEID ulTileId,
    AcMapMbMapBookManager::EAdjacentTile tileDir
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ulTileId</td>
<td>Input Id of the adjacent tile; if Id is equal to the Id of this tile, the adjacency in the specified direction will be removed.</td>
</tr>
<tr>
<td>kTileDir</td>
<td>Input direction.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; Returns Acad::eNotApplicable in case the tile does not have adjacent tile in the specified direction, otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns the unique handle of the Sheet in the Sheet Set.

```cpp
const ACHAR* SheetHandle() const;
```

Returns Tile handle

Remarks

This handle string is what IAcSmObjectId::GetHandle() returns.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapMbTile Class, AcMapMbTile Class
AcMapMbTile:: StreamIn Method
AcMapMbTile Class | AcMapMbTile Class

Reads data from a dwg file.

```cpp
virtual Acad::ErrorStatus StreamIn(
    AcDbDwgFiler* pFiler
);
```

**Parameters**
- **pFiler**: Input filer. The filer should be open and ready to read this tile data from.

**Returns**
Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Reads data from a dxf file.

```
virtual Acad::ErrorStatus StreamIn(
    AcDbDxfFiler* pFiler
);
```

Parameters

- **pFiler**: Input filer. The filer should be open and ready to read this tile data from.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Loads data into a dwg file.

```cpp
virtual Acad::ErrorStatus StreamOut(
    AcDbDwgFiler* pFiler
) const;
```

**Parameters**

<table>
<thead>
<tr>
<th>pFiler</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>input filer. The filer should be open and ready to write this tile data.</td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Loads data into a dxf file.

```cpp
virtual Acad::ErrorStatus StreamOut(
    AcDbDxfFiler* pFiler
) const;
```

**Parameters**

<table>
<thead>
<tr>
<th>pFiler</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input filer. The filer should be open and ready to write this tile data.</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbTile Class, AcMapMbTile Class
AcMapMbTile:: TileId Method
AcMapMbTile Class | AcMapMbTile Class

Returns the unique tile Id for this tile.

AcMapMbMapBookManager::TILEID TileId() const;

Returns Tile Id

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbTile Class, AcMapMbTile Class
AcMapMbTile:: TileSet Method
AcMapMbTile Class | AcMapMbTile Class

Returns the pointer to the Tile Set it belongs to.

AcMapMbTileSet* TileSet() const;
Returns

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbTileGenerator Class, AcMapMbTileGenerator Class

AcMapMbTileGenerator::: ~AcMapMbTileGenerator Destructor
AcMapMbTileGenerator Class | AcMapMbTileGenerator Class

Destroys an instance of this class.

virtual ~AcMapMbTileGenerator();

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Constructs an instance of this class.

AcMapMbTileGenerator();

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Creates the tile set from the settings.

```cpp
virtual Acad::ErrorStatus GenerateTiles(
    AcMapMbTileSet& tileSet,
    const AcMapMbTileGeneratorSettings& tSettings,
    double dWidth,
    double dHeight,
    AcDbDatabase* pDb
) = 0;
```

**Parameters**

- **tileSet**  
  Output Tile set class.
- **tSettings**  
  Input Tile settings class.
- **dWidth**  
  Input width of tiles in model units.
- **dHeight**  
  Input height of tiles in model units.
- **pDb**  
  Input Database where Tiles, Tile set, and boundary polylines if any will be created. Typically indicates the current drawing.

**Returns**

Returns Acad::eOk if successful; otherwise, returns another value Returns Acad::eInvalidInput if the settings are incorrect for the Generator type. Returns Acad::eAmbiguousInput if global extents are empty or other data problems. Returns Acad::eCreateFailed for Manual tile generator and no reference objects were passed in. Returns Acad::eOutOfMemory for allocate memory or other serious errors.

**Remarks**

In addition to the error returns listed below, messages may be added to the Error stack where such messages would be useful to a user in deciding what input settings would be more useful.

Created with a commercial version of [Doc-O-Matic](#). In order to make this message disappear you need to register this software. If you have problems
registering this software please contact us at support@toolsfactory.com.
This enum defines the different sources for a MapBook.

```c
enum EMapBookSource {
    kMBSourceUnknown = 0,
    kMBSourceDMMMap,
    kMBSourceModelSpace,
    kMBImportOldPlot
};
```

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
This enum defines the different tiling schemes we have available.

```cpp
enum ETilingScheme {
    kTilingSchemeUnknown = 0,
    kTilingSchemeManual,
    kTilingSchemeGridArea,
    kTilingSchemeGridRowCol
};
```

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

```cpp
virtual ~AcMapMbTileGeneratorSettings();
```

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Constructs an instance of this class.

AcMapMbTileGeneratorSettings();

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns an enum which specifies the active tiling scheme.

`ETilingScheme ActiveTilingScheme() const;`

Returns tiling scheme or `kTilingSchemeUnknown` if not set.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
This method is used to allocate a tile generator that can be used by the settings object.

**virtual AcMapMbTileGenerator** * AllocateTileGenerator() **const**;

Returns a pointer to a Tile Generator object or NULL if unsuccessful.

Remarks

The caller is responsible for deleting the object.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Copies the contents of an object into the messaged object, if feasible. See also copyFrom() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus copyFrom(
    const AcRxObject* other
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Description</th>
<th>other</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input pointer to the object to copy from.</td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](http://example.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns the boolean value whether to disable empty tiles.

```cpp
bool DisableEmptyTiles() const;
```

Returns whether to disable empty tiles

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
The pointer points to memory holding the dm map name. It is recommended that if you are going to keep this pointer for any length of time you make a copy of it as the contents can be changed at any time. The user should not attempt to delete this memory. This string can be empty to indicate model space but will never be NULL.

```cpp
const ACHAR* DMMapName() const;
```

Returns Dm Map Name or empty string to indicate Model space.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns the scale factor of the DM Map for the Map Book.

```cpp
double DmScaleFactor() const;
```

Returns scale factor or DBL_MAX if not set.

Remarks

This is an identifier only to identify which scale range of a DM map to use for this book.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Let this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dwgInFields(
    AcDbDwgFiler* pFiler
);
```

**Parameters**

- **pFiler**: Input filer to use to read the object's data.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapMbTileGeneratorSettings Class, AcMapMbTileGeneratorSettings Class
AcMapMbTileGeneratorSettings:: dwgOutFields Method
AcMapMbTileGeneratorSettings Class | AcMapMbTileGeneratorSettings Class

Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

```c++
virtual Acad::ErrorStatus dwgOutFields(
    AcDbDwgFiler* pFiler
) const;
```

Parameters
- **pFiler**: Input filer to use to write the object's data.

Returns
- Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbTileGeneratorSettings Class, AcMapMbTileGeneratorSettings Class
AcMapMbTileGeneratorSettings:: dxfInFields Method
AcMapMbTileGeneratorSettings Class | AcMapMbTileGeneratorSettings Class

Lets this object read its data. See also dxfInFields() in the AutoCAD ObjectARX
Developer's Guide. This overridden function is called by the system as needed; it
is unlikely that you will need to call it directly.

virtual Acad::ErrorStatus dxfInFields(
    AcDbDxfFiler* pFiler
);
Parameters       Description
pFiler            Input filer to use to read the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this
message disappear you need to register this software. If you have problems
registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbTileGeneratorSettings Class, AcMapMbTileGeneratorSettings Class
AcMapMbTileGeneratorSettings:: dxfOutFields Method
AcMapMbTileGeneratorSettings Class | AcMapMbTileGeneratorSettings Class

Lets this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dxfOutFields(
    AcDbDxfFiler* pFiler
) const;
```

Parameters Description
pFiler Input filer to use to write the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns the selection set of entities intersected.

```
Acad::ErrorStatus GetIntersectEntitySet(
    AcDbObjectIdArray& SelSet
) const;
```

Parameters

- **SelSet**
  - Output id array of selection set.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbTileGeneratorSettings Class, AcMapMbTileGeneratorSettings Class
AcMapMbTileGeneratorSettings:: Implementation Method
AcMapMbTileGeneratorSettings Class | AcMapMbTileGeneratorSettings Class

Returns the implementation object.

**virtual** AcMapMbImpTileGeneratorSettings* Implementation();

Returns the implementation, or NULL if unsuccessful.

Remarks

This function is for internal use only; do not call it directly.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns the boolean value whether to use a particular set of entities as an intersection set when determining which tiles are empty.

```cpp
bool IntersectEntities() const;
```

Returns whether to intersect for entities.

Remarks

If disable empty tiles is false, then the IntersectEntities and IntersectEntitiesSet properties have no effect. If disable empty tiles is true, and intersect entities is true and the Intersect Entities set is non-empty, then this set is used as and intersection set to determine empty tiles. Otherwise the entire drawing is used as the intersection set.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Methods used to store and retrieve the data.

Returns an enum which specifies the Source type.

**EMapBookSource** `MapBookSource()` **const**;

Returns Source Type or `kMBSourceUnknown` if not set.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapMbTileGeneratorSettings Class, AcMapMbTileGeneratorSettings Class
AcMapMbTileGeneratorSettings::OverlapPercent Method
AcMapMbTileGeneratorSettings Class | AcMapMbTileGeneratorSettings Class

Returns the Overlap percent.

```cpp
double OverlapPercent() const;
```

Returns overlap percent.

Remarks

Note that an overlap of 100 indicates that the tile is twice the size of an overlap of 0. In a rectangular grid the tiles will overlap to the midpoint of the adjacent tiles. This convention is used in the User Interface as well.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns the scale factor for the Map Book.

```cpp
double ScaleFactor() const;
```

Returns scale factor.

Remarks

A 1: prefix is assumed so if the scale is 1:200 then this value will return 200. This value is applied to the physical size of the layout main tile viewport to determine tile size.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets an enum which specifies the active tiling scheme.

```cpp
Acad::ErrorStatus setActiveTilingScheme(
    ETilingScheme tilingScheme
);
```

**Parameters**

<table>
<thead>
<tr>
<th>tilingScheme</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input tiling scheme.</td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the boolean value whether to disable empty tiles.

```cpp
Acad::ErrorStatus SetDisableEmptyTiles(
    bool bDisableEmptyTiles
);
```

**Parameters**
- `bDisableEmptyTiles`: Input whether to disable empty tiles.

**Returns**
- Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the scale of the DM Map for the Map Book.

\[
\text{Acad::ErrorStatus SetDmScaleFactor(}
\text{    const double dDmScaleFactor}
\text{);} \\
\text{Parameters} \\
\text{dDmScaleFactor Input Display Manager scale factor.} \\
\text{Returns} \\
\text{Returns Acad::eOk if successful; otherwise, returns a different error code.} \\
\text{Remarks} \\
\text{His is an identifier only to identify which scale range of a DM map to use for this book.} \\
\text{Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.}
Sets the boolean value whether to use the intersection set.

```cpp
Acad::ErrorStatus SetIntersectEntities(
    const bool bIntersectEntities
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bIntersectEntities</td>
<td>Input Boolean whether to use the intersection set.</td>
</tr>
</tbody>
</table>

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the selection set of entities for determining empty tiles.

```cpp
Acad::ErrorStatus SetIntersectEntitySet(
    const AcDbObjectIdArray& SelSet
);
```

**Parameters**
- **SelSet**: Input id array of Selection set.

**Returns**
- Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets an enum which specifies the Source type.

```
Acad::ErrorStatus SetMapBookSource(
    const EMapBookSource sourceType
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sourceType</td>
<td>Input source type.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the Overlap data member.

```cpp
Acad::ErrorStatus SetOverlapPercent(
    const double dOverlapPercent
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dOverlapPercent</td>
<td>Input percent of overlap.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the scale data member.

```cpp
Acad::ErrorStatus SetScaleFactor(
    const double dScaleFactor);
```

**Parameters**
- **dScaleFactor**: Input scale factor.

**Returns**
- Returns Acad::eOk if successful; otherwise, returns a different error code.

**Remarks**

A 1: prefix is assumed so if the scale is 1:200 then this value should be 200. This value is applied to the physical size of the layout main tile viewport to determine tile size.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns a pointer to memory holding the tile layer name.

```cpp
virtual const ACHAR * TileLayerName() const;
```

Returns tile layer name or an empty string if none has been set.

Remarks

It is recommended that if you are going to keep this pointer for any length of time you make a copy of it as the contents can be changed at any time. The user should not attempt to delete the memory returned.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

```cpp
virtual ~AcMapMbTileGeneratorSettingsArea();
```

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Constructs an instance of this class.

AcMapMbTileGeneratorSettingsArea();

Returns nothing.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
This method is used to allocate a tile generator that can be used by the settings object.

```cpp
virtual AcMapMbTileGenerator* AllocateTileGenerator() const;
```

Returns a pointer to a Tile Generator object or NULL if unsuccessful.

Remarks

The caller is responsible for deleting the object.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns the first corner point of the area to tile.

**virtual const** AcGePoint2d& AreaFirstPoint() **const**;

Returns the first corner point. If this value has not been set, the origin (0, 0) is returned.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Return the second or opposite corner point of the area to tile.

`virtual const AcGePoint2d& AreaLastPoint() const;`

Returns the second or opposite corner point. If this value has not been set, the origin (0, 0) is returned.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Clones the object. See also clone() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual AcRxObject* clone() const;
```

Returns a pointer to the clone, or NULL if cloning was unsuccessful.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbTileGeneratorSettingsArea Class,
AcMapMbTileGeneratorSettingsArea Class
AcMapMbTileGeneratorSettingsArea:: copyFrom Method
AcMapMbTileGeneratorSettingsArea Class |
AcMapMbTileGeneratorSettingsArea Class

Copies the contents of an object into the messaged object, if feasible. See also copyFrom() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus copyFrom(
    const AcRxObject* other
);
```

Parameters

- other

Description

Input pointer to the object to copy from.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbTileGeneratorSettingsArea Class,
AcMapMbTileGeneratorSettingsArea Class
AcMapMbTileGeneratorSettingsArea:: dwgInFields Method
AcMapMbTileGeneratorSettingsArea Class |
AcMapMbTileGeneratorSettingsArea Class

Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
template virtual Acad::ErrorStatus dwgInFields(
    AcDbDwgFiler* pFiler
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pFiler</td>
<td>Input filer to use to read the object's data.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Let's this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dwgOutFields(  
    AcDbDwgFiler* pFiler
) const;
```

**Parameters**

- **pFiler**
  - Input filer to use to write the object's data.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapMbTileGeneratorSettingsArea Class,
AcMapMbTileGeneratorSettingsArea Class
AcMapMbTileGeneratorSettingsArea:: dxfInFields Method
AcMapMbTileGeneratorSettingsArea Class |
AcMapMbTileGeneratorSettingsArea Class

Lets this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dxfInFields(  
    AcDbDxfFiler* pFiler
);
```

Parameters

- **pFiler**
  - Description: Input filer to use to read the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Lets this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dxfOutFields(
    AcDbDxfFiler* pFiler
) const;
```

**Parameters**

- *pFiler*: Input filer to use to write the object's data.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Returns the extent points of the area to tile.

```cpp
virtual Acad::ErrorStatus GetAreaPoints(
    AcGePoint2d& firstPoint,
    AcGePoint2d& lastPoint
) const;
```

**Parameters**

- `firstPoint` Output, first corner point
- `lastPoint` Output, second or opposite corner point

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Gets the extents of the area to tile.

```cpp
virtual Acad::ErrorStatus GetExtents(
    double& dXExtMax,
    double& dYExtMax,
    double& dXExtMin,
    double& dYExtMin
) const;
```

**Parameters**
- `dXExtMax`  
  Output, Holds x value of EXTMAX of area to tile
- `dYExtMax`  
  Output, Holds y value of EXTMAX of area to tile
- `dXExtMin`  
  Output, Holds x value of EXTMIN of area to tile
- `dYExtMin`  
  Output, Holds y value of EXTMIN of area to tile

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://toolsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the first corner point of the area to tile.

```cpp
virtual Acad::ErrorStatus SetAreaFirstPoint(
    const AcGePoint2d& firstPoint
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>firstPoint</td>
<td>Input, first corner point.</td>
</tr>
</tbody>
</table>

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the second or opposite corner point of the area to tile.

```cpp
virtual Acad::ErrorStatus SetAreaLastPoint(
    const AcGePoint2d& lastPoint
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>lastPoint</td>
<td>Input, second or opposite corner</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the start point of the area to tile.

```cpp
virtual Acad::ErrorStatus SetAreaPoints(
    const AcGePoint2d& firstPoint,
    const AcGePoint2d& lastPoint);
```

**Parameters**

- `firstPoint` Input, first corner point
- `lastPoint` Input, second or opposite corner point

**Returns**

Returns `Acad::eOk` if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.toolsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the extents of the area to tile.

```cpp
virtual Acad::ErrorStatus SetExtents(
    const double dXExtMax,
    const double dYExtMax,
    const double dXExtMin,
    const double dYExtMin
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dXExtMax</td>
<td>Input, Holds x value of EXTMAX of area to tile</td>
</tr>
<tr>
<td>dYExtMax</td>
<td>Input, Holds y value of EXTMAX of area to tile</td>
</tr>
<tr>
<td>dXExtMin</td>
<td>Input, Holds x value of EXTMIN of area to tile</td>
</tr>
<tr>
<td>dYExtMin</td>
<td>Input, Holds y value of EXTMIN of area to tile</td>
</tr>
</tbody>
</table>

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.

Links
AcMapMbTileGeneratorSettingsArea Class,
AcMapMbTileGeneratorSettingsArea Class
AcMapMbTileGeneratorSettingsArea:: SetExtents Method
AcMapMbTileGeneratorSettingsArea Class | AcMapMbTileGeneratorSettingsArea Class
Destroys an instance of this class.

```cpp
virtual ~AcMapMbTileGeneratorSettingsGrid();
```

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Constructs an instance of this class.

AcMapMbTileGeneratorSettingsGrid();

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
This method is used to allocate a tile generator that can be used by the settings object.

```cpp
virtual AcMapMbTileGenerator* AllocateTileGenerator() const;
```

Returns

Returns a pointer to a Tile Generator object or NULL if unsuccessful.

Remarks

The caller is responsible for deleting the object.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Return the start point of the area to tile.

```cpp
virtual const AcGePoint2d& AreaStartPoint() const;
```

Returns the start point.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Clones the object. See also clone() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

```
virtual AcRxObject* clone() const;
```

Returns a pointer to the clone, or NULL if cloning was unsuccessful.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Copies the contents of an object into the messaged object, if feasible. See also `copyFrom()` in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus copyFrom(
    const AcRxObject* other
);
```

**Parameters**

- `other`

**Description**

Input pointer to the object to copy from.

**Returns**

Returns `Acad::eOk` if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapMbTileGeneratorSettingsGrid Class
AcMapMbTileGeneratorSettingsGrid Class
AcMapMbTileGeneratorSettingsGrid::dwgInFields Method
AcMapMbTileGeneratorSettingsGrid Class |
AcMapMbTileGeneratorSettingsGrid Class

Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dwgInFields(
    AcDbDwgFiler* pFiler
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pFiler</td>
<td>Input filer to use to read the object's data.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Let's this object write its data. See also `dwgOutFields()` in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dwgOutFields(
    AcDbDwgFiler* pFiler
) const;
```

**Parameters**

- **pFiler**
  - Input filer to use to write the object's data.

**Returns**

Returns ` Acad::eOk` if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbTileGeneratorSettingsGrid Class, AcMapMbTileGeneratorSettingsGrid Class
AcMapMbTileGeneratorSettingsGrid:: dxfInFields Method
AcMapMbTileGeneratorSettingsGrid Class | AcMapMbTileGeneratorSettingsGrid Class

Lets this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dxfInFields(
    AcDbDxfFiler* pFiler
);
```

Parameters

<table>
<thead>
<tr>
<th>pFiler</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input filer to use to read the object's data.</td>
<td></td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Let's this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dxfOutFields(
    AcDbDxfFiler* pFiler
) const;
```

**Parameters**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pFiler</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](http://toolsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links

AcMapMbTileGeneratorSettingsGrid Class
AcMapMbTileGeneratorSettingsGrid Class
AcMapMbTileGeneratorSettingsGrid:: NumberOfColumns Method
AcMapMbTileGeneratorSettingsGrid Class |
AcMapMbTileGeneratorSettingsGrid Class

Returns the integer value indicating number of columns.

```cpp
int NumberOfColumns() const;
```

Returns

Returns the number of Columns. 0 probably indicates that the value has not been set.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns the integer value indicating number of rows.

```cpp
int NumberOfRows() const;
```

Returns

Returns the number of Rows. 0 probably indicates that the value has not been set.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Set the start point of the area to tile.

```cpp
virtual Acad::ErrorStatus SetAreaStartPoint(
    const AcGePoint2d& firstPoint);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>firstPoint</td>
<td>Lower left point (minimum) to indicate where the area to tile begins.</td>
</tr>
</tbody>
</table>

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the integer value indicating number of columns.

```
Acad::ErrorStatus SetNumberOfColumns(
    const unsigned int nNumberOfColumns
);
```

Parameters

- nNumberOfColumns: value to store

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the integer value indicating number of rows.

```
Acad::ErrorStatus SetNumberOfRows(
    const unsigned int nNumberOfRows
);
```

**Parameters**
- **nNumberOfRows**: value to store

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.doctomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Destroys an instance of this class.

\texttt{\sim AcMapMbTileGeneratorSettingsManual()};

Returns

Returns nothing.

Created with a commercial version of \texttt{Doc-O-Matic}. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at \texttt{support@toolsfactory.com}.
AcMapMbTileGeneratorSettingsManual::

Constructs an instance of this class.

AcMapMbTileGeneratorSettingsManual();

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
This method is used to allocate a tile generator that can be used by the settings object.

```cpp
virtual AcMapMbTileGenerator* AllocateTileGenerator() const;
```

Returns a pointer to a Tile Generator object or NULL if unsuccessful.

Remarks

The caller is responsible for deleting the object.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Clones the object. See also clone() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual AcRxObject* clone() const;
```

Returns a pointer to the clone, or NULL if cloning was unsuccessful.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Copies the contents of an object into the messaged object, if feasible. See also copyFrom() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus copyFrom(
    const AcRxObject* other
);
```

Parameters

- `other`: Input pointer to the object to copy from.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Let this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dwgInFields(
    AcDbDwgFiler* pFiler
);
```

**Parameters**

- **pFiler**: Input filer to use to read the object's data.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](#). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapMbTileGeneratorSettingsManual Class,
AcMapMbTileGeneratorSettingsManual Class
AcMapMbTileGeneratorSettingsManual:: dwgOutFields Method
AcMapMbTileGeneratorSettingsManual Class |
AcMapMbTileGeneratorSettingsManual Class

Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

virtual Acad::ErrorStatus dwgOutFields(
    AcDbDwgFiler* pFiler
) const;

Parameters  Description
pFiler  Input filer to use to write the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Let this object read its data. See also dxflnFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dxflnFields(
    AcDbDxfFiler* pFiler
);
```

**Parameters**

- `pFiler`: Input filer to use to read the object’s data.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.doc-o-matic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
virtual Acad::ErrorStatus dxfOutFields(
    AcDbDxfFiler* pFiler
) const;

Parameters Description
pFiler Input filer to use to write the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Returns the object ids of entities selected for tile boundaries.

```cpp
Acad::ErrorStatus GetSelectedTileBoundaries(
    AcDbObjectIdArray& SelSet
) const;
```

Parameters

- **SelSet**: Output ids of the tile boundaries.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the selection set of entities selected.

```cpp
Acad::ErrorStatus SetSelectedTileBoundaries(
    const AcDbObjectIdArray& SelSet
);
```

**Parameters**
- **SelSet**: Input Selection set value to store

**Returns**
- Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
This is overriden from the base class since layer name is not appropriate for manual selection.

```
virtual const ACHAR * TileLayerName() const;
```

Returns

Returns the empty string.

Remarks

This function returns the empty string.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

```cpp
virtual ~AcMapMbTileNameGenerator();
```

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.doc-omatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Constructs an instance of this class.

AcMapMbTileNameGenerator();

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Iterates through tile set and sets the names of individual tiles.

```cpp
virtual Acad::ErrorStatus GenerateTileNames(
    const AcMapMbTileNameGeneratorSettings& tSettings,
    AcMapMbTileSet& TileSet
) = 0;
```

**Parameters**

- `tSettings`  
  Input Tile Name settings class.
- `tileSet`  
  Output Tile set class.

**Returns**

Returns `Acad::eOk` if successful; otherwise, returns another value. Returns `Acad::eInvalidInput` if the settings are incorrect for the Generator type.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

```cpp
virtual ~AcMapMbTileNameGeneratorData();
```

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbTileNameGeneratorData Class, AcMapMbTileNameGeneratorData Class
AcMapMbTileNameGeneratorData::
AcMapMbTileNameGeneratorData Constructor
AcMapMbTileNameGeneratorData Class | AcMapMbTileNameGeneratorData Class

Constructs an instance of this class.

AcMapMbTileNameGeneratorData();
Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Iterates through tile set and sets the names of individual tiles.

```cpp
virtual Acad::ErrorStatus GenerateTileNames(
    const AcMapMbTileNameGeneratorSettings& tSettings,
    AcMapMbTileSet& TileSet
);```

**Parameters**

- `tSettings`  
  Input Tile Name settings class.

- `tileSet`  
  Output Tile set class.

**Returns**

- Returns Acad::eOk if successful; otherwise, returns another value
- Returns Acad::eInvalidInput if the settings are incorrect for the Generator type.

Created with a commercial version of [Doc-O-Matic](https://example.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Destroys an instance of this class.

**virtual** ~AcMapMbTileNameGeneratorGrid();

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbTileNameGeneratorGrid Class, AcMapMbTileNameGeneratorGrid Class
AcMap MbTileNameGeneratorGrid::
AcMapMbTileNameGeneratorGrid Constructor
AcMapMbTileNameGeneratorGrid Class | AcMapMbTileNameGeneratorGrid Class

Constructs an instance of this class.

AcMapMbTileNameGeneratorGrid();
Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Iterates through tile set and sets the names of individual tiles.

```cpp
virtual Acad::ErrorStatus GenerateTileNames(
    const AcMapMbTileNameGeneratorSettings& tSettings,
    AcMapMbTileSet& TileSet
);
```

**Parameters**
- `tSettings`  
  Input Tile Name settings class.
- `tileSet`  
  Output Tile set class.

**Returns**
Returns Acad::eOk if successful; otherwise, returns another value Returns Acad::eInvalidInput if the settings are incorrect for the Generator type.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
This is record `AcMapMbTileNameGeneratorIndexer::EIndexOrientation`.

```
enum EIndexOrientation {
  kOrientUnknown = 0,
  kOrientRow,
  kOrientColumn
};
```

File

`AcMapMbTileNameGeneratorSettings.h`

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapMbTileNameGeneratorIndexer Class,
AcMapMbTileNameGeneratorIndexer Class
AcMapMbTileNameGeneratorIndexer:: ~AcMapMbTileNameGeneratorIndexer Destructor
AcMapMbTileNameGeneratorIndexer Class |
AcMapMbTileNameGeneratorIndexer Class

Destroys an instance of this class.

~AcMapMbTileNameGeneratorIndexer();

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
AcMapMbTileNameGeneratorIndexer::
AcMapMbTileNameGeneratorIndexer Constructor

Constructs an instance of this class.

```cpp
AcMapMbTileNameGeneratorIndexer(
    EIndexOrientation Orient = kOrientUnknown,
    bool bIsInverted = false,
    bool bIsNumeric = true,
    long lIndex = 1,
    unsigned short uIncrement = 1
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Orient</td>
<td>Input orientation. Tells the user what kind of index this is.</td>
</tr>
<tr>
<td>bIsInverted</td>
<td>Input whether the index starts at the front (top) or back</td>
</tr>
<tr>
<td></td>
<td>(bottom)</td>
</tr>
<tr>
<td>bIsNumeric</td>
<td>Input whether the index is numeric or alphabetic.</td>
</tr>
<tr>
<td>lIndex</td>
<td>Input starting index value.</td>
</tr>
<tr>
<td>uIncrement</td>
<td>Input increment between values of the index.</td>
</tr>
</tbody>
</table>

Returns

Returns nothing.

Created with a commercial version of [Doc-O-Matic](#). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the increment value for the index.

```cpp
unsigned short Increment() const;
```

Returns

Returns the increment value.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbTileNameGeneratorIndexer Class,
AcMapMbTileNameGeneratorIndexer Class
AcMapMbTileNameGeneratorIndexer:: Index Method
AcMapMbTileNameGeneratorIndexer Class | 
AcMapMbTileNameGeneratorIndexer Class

Retrieves the starting Index.

```cpp
int Index() const;
```

Returns

Returns the starting Index as a number.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves whether the indexing should start at the top or front or at the back or bottom.

```cpp
bool IsInverted() const;
```

Returns indexing direction.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapMbTileNameGeneratorIndexer Class,
AcMapMbTileNameGeneratorIndexer Class
AcMapMbTileNameGeneratorIndexer:: IsNumeric Method
AcMapMbTileNameGeneratorIndexer Class |
AcMapMbTileNameGeneratorIndexer Class

Retrieves whether the indexing is numeric or alphabetic.

bool IsNumeric() const;

Returns

Returns if Indexing is numeric.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links

AcMapMbTileNameGeneratorIndexer Class, AcMapMbTileNameGeneratorIndexer Class
AcMapMbTileNameGeneratorIndexer:: Orientation Method
AcMapMbTileNameGeneratorIndexer Class | AcMapMbTileNameGeneratorIndexer Class

Retrieves the orientation or type of Index.

EIndexOrientation Orientation() const;
Returns

Returns the orientation.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
This is StreamIn, a member of class AcMapMbTileNameGeneratorIndexer.

```cpp
virtual Acad::ErrorStatus StreamIn(
    AcDbDwgFiler* pFiler
);
```

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
This is StreamIn, a member of class AcMapMbTileNameGeneratorIndexer.

```
virtual Acad::ErrorStatus StreamIn(
    AcDbDxfFiler* pFiler
);
```

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
This is StreamOut, a member of class AcMapMbTileNameGeneratorIndexer.

```cpp
virtual Acad::ErrorStatus StreamOut(
    AcDbDwgFiler* pFiler
) const;
```

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbTileNameGeneratorIndexer Class,
AcMapMbTileNameGeneratorIndexer Class
AcMapMbTileNameGeneratorIndexer:: StreamOut Method
AcMapMbTileNameGeneratorIndexer Class |
AcMapMbTileNameGeneratorIndexer Class

This is StreamOut, a member of class AcMapMbTileNameGeneratorIndexer.

virtual Acad::ErrorStatus StreamOut(
    AcDbDxfFiler* pFiler
) const;
Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

\texttt{virtual \ ~AcMapMbTileNameGeneratorSequence();}

Returns nothing.

Created with a commercial version of \texttt{Doc-O-Matic}. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at \texttt{support@toolsfactory.com}.
Constructs an instance of this class.

AcMapMbTileNameGeneratorSequence();

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Iterates through tile set and sets the names of individual tiles.

```cpp
virtual Acad::ErrorStatus GenerateTileNames(
    const AcMapMbTileNameGeneratorSettings& tSettings,
    AcMapMbTileSet& tileSet
);
```

**Parameters**

- `tSettings` Input Tile Name settings class.
- `tileSet` Output Tile set class.

**Returns**

Returns Acad::eOk if successful; otherwise, returns another value

Returns Acad::eInvalidInput if the settings are incorrect for the Generator type.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
enum ETileNamingScheme {
    kTileNamingSchemeUnknown = 0,
    kTileNamingSchemeGridColRow,
    kTileNamingSchemeGridSequential,
    kTileNamingSchemeSequential,
    kTileNamingSchemeData
};
Destroys an instance of this class.

```cpp
virtual ~AcMapMbTileNameGeneratorSettings();
```

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Constructs an instance of this class.

AcMapMbTileNameGeneratorSettings();

Returns

Returns nothing.

Remarks

The user should never directly create an instance of this class, only the derived classes.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbTileNameGeneratorSettings Class,
AcMapMbTileNameGeneratorSettings Class
AcMapMbTileNameGeneratorSettings:: AllocateTileNameGenerator Method
AcMapMbTileNameGeneratorSettings Class |
AcMapMbTileNameGeneratorSettings Class

Returns the associated Name Generator.

virtual AcMapMbTileNameGenerator* AllocateTileNameGenerator() const;
Returns

Returns Name Generator.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbTileNameGeneratorSettings Class,
AcMapMbTileNameGeneratorSettings Class
AcMapMbTileNameGeneratorSettings:: copyFrom Method
AcMapMbTileNameGeneratorSettings Class |
AcMapMbTileNameGeneratorSettings Class

Copies the contents of an object into the messaged object, if feasible. See also
copyFrom() in the AutoCAD ObjectARX Developer's Guide. This overridden
function is called by the system as needed; it is unlikely that you will need to call
it directly.

virtual Acad::ErrorStatus copyFrom(
    const AcRxObject* other
);

Parameters | Description
- other | Input pointer to the object to copy from.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this
message disappear you need to register this software. If you have problems
registering this software please contact us at support@toolsfactory.com.
Returns the count disable tiles data mamber.

```cpp
bool CountDisabledTiles() const;
```

Returns whether disabled tiles should be counted.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbTileNameGeneratorSettings Class,
AcMapMbTileNameGeneratorSettings Class
AcMapMbTileNameGeneratorSettings:: dwgInFields Method
AcMapMbTileNameGeneratorSettings Class |
AcMapMbTileNameGeneratorSettings Class

Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dwgInFields(
    AcDbDwgFiler* pFiler
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pFiler</td>
<td>Input filer to use to read the object's data.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
virtual Acad::ErrorStatus dwgOutFields(
   AcDbDwgFiler* pFiler
) const;

Parameters | Description
------------|-------------------------------------------------
pFiler     | Input filer to use to write the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
virtual Acad::ErrorStatus dxfInFields(
    AcDbDxfFiler* pFiler
);

Parameters Description
pFiler Input filer to use to read the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Let this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dxfOutFields(
    AcDbDxfFiler* pFiler
) const;
```

**Parameters**
- **pFiler**: Input filer to use to write the object's data.

**Returns**
- Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Returns the implementation object.

```cpp
virtual AcMapMbImpTileNameGeneratorSettings* Implementation();
```

Returns the implementation, or NULL if unsuccessful.

Remarks

This function is for internal use only; do not call it directly.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the count disable tiles data member.

```cpp
Acad::ErrorStatus SetCountDisabledTiles(
    const bool bCountDisabledTiles
);
```

**Parameters**

bCountDisabledTiles  Input to specify how to handle disabled tiles.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

**Remarks**

When true, disabled tiles are named, and the index is incremented. When false, disabled tiles are skipped and the index is not incremented.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns an enum which specifies the tile naming scheme.

```cpp
Acad::ErrorStatus SetTileNamingScheme(
    const ETileNamingScheme tileNamingScheme
);
```

**Parameters**
- `tileNamingScheme`: variable to set enum value

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapMbTileNameGeneratorSettings Class,
AcMapMbTileNameGeneratorSettings Class
AcMapMbTileNameGeneratorSettings:: TileNamingScheme Method
AcMapMbTileNameGeneratorSettings Class |
AcMapMbTileNameGeneratorSettings Class

Returns an enum which specifies the tile naming scheme.

ETileNamingScheme TileNamingScheme() const;
Returns

Returns tile naming scheme.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

```cpp
virtual ~AcMapMbTileNameGeneratorSettingsData();
```

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Constructs an instance of this class.

AcMapMbTileNameGeneratorSettingsData();

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns the associated Name Generator.

```
virtual AcMapMbTileNameGenerator* AllocateTileNameGenerator() const;
```

Returns Name Generator.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Clones the object. See also clone() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual AcRxObject* clone() const;
```

Returns a pointer to the clone, or NULL if cloning was unsuccessful.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbTileNameGeneratorSettingsData Class,
AcMapMbTileNameGeneratorSettingsData Class
AcMapMbTileNameGeneratorSettingsData:: copyFrom Method
AcMapMbTileNameGeneratorSettingsData Class | AcMapMbTileNameGeneratorSettingsData Class

Copies the contents of an object into the messaged object, if feasible. See also copyFrom() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus copyFrom(
    const AcRxObject* other
);
```

Parameters    Description
other          Input pointer to the object to copy from.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbTileNameGeneratorSettingsData Class, AcMapMbTileNameGeneratorSettingsData Class
AcMapMbTileNameGeneratorSettingsData:: dwgInFields Method
AcMapMbTileNameGeneratorSettingsData Class | AcMapMbTileNameGeneratorSettingsData Class

Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

virtual Acad::ErrorStatus dwgInFields(
    AcDbDwgFiler* pFiler
);

Parameters Description
pFiler Input filer to use to read the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Let this object write its data. See also `dwgOutFields()` in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
template
virtual Acad::ErrorStatus dwgOutFields(
    AcDbDwgFiler* pFiler
) const;
```

Parameters Description
---
pFiler Input filer to use to write the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbTileNameGeneratorSettingsData Class, AcMapMbTileNameGeneratorSettingsData Class
AcMapMbTileNameGeneratorSettingsData:: dxfInFields Method
AcMapMbTileNameGeneratorSettingsData Class | AcMapMbTileNameGeneratorSettingsData Class

Lets this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dxfInFields(
    AcDbDxfFiler* pFiler
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pFiler</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](http://example.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
virtual Acad::ErrorStatus dxfOutFields(
    AcDbDxfFiler* pFiler
) const;

Parameters Description
pFiler Input filer to use to write the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbTileNameGeneratorSettingsData Class,
AcMapMbTileNameGeneratorSettingsData Class
AcMapMbTileNameGeneratorSettingsData:: Expression Method
AcMapMbTileNameGeneratorSettingsData Class |
AcMapMbTileNameGeneratorSettingsData Class

Retrieves the expression to be applied to the reference object.

```cpp
const ACHAR * Expression() const;
```

Returns

Returns the expression.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
This enum indicates the naming order for the grid.

```cpp
enum EGridNamingOrder {
    kGridNamingOrderUnknown = 0,
    kGridNamingOrderLRTB,
    kGridNamingOrderLRBT,
    kGridNamingOrderRLTB,
    kGridNamingOrderRLBT,
    kGridNamingOrderTBLR,
    kGridNamingOrderTBRL,
    kGridNamingOrderBTRL,
    kGridNamingOrderBTRL
};
```

File

AcMapMbTileNameGeneratorSettingsGrid.h

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbTileNameGeneratorSettingsGrid Class,
AcMapMbTileNameGeneratorSettingsGrid Class
AcMapMbTileNameGeneratorSettingsGrid::
~AcMapMbTileNameGeneratorSettingsGrid Destructor
AcMapMbTileNameGeneratorSettingsGrid Class |
AcMapMbTileNameGeneratorSettingsGrid Class

Destroys an instance of this class.

virtual ~AcMapMbTileNameGeneratorSettingsGrid();

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbTileNameGeneratorSettingsGrid Class,
AcMapMbTileNameGeneratorSettingsGrid Class
AcMapMbTileNameGeneratorSettingsGrid::
AcMapMbTileNameGeneratorSettingsGrid Constructor
AcMapMbTileNameGeneratorSettingsGrid Class |
AcMapMbTileNameGeneratorSettingsGrid Class

Constructs an instance of this class.

AcMapMbTileNameGeneratorSettingsGrid();

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbTileNameGeneratorSettingsGrid Class,
AcMapMbTileNameGeneratorSettingsGrid Class
AcMapMbTileNameGeneratorSettingsGrid:::
AllocateTileNameGenerator Method
AcMapMbTileNameGeneratorSettingsGrid Class |
AcMapMbTileNameGeneratorSettingsGrid Class

Returns the associated Name Generator.

virtual AcMapMbTileNameGenerator* AllocateTileNameGenerator() const;
Returns

Returns Name Generator.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Clones the object. See also clone() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

```c++
virtual AcRxObject* clone() const;
```

Returns

Returns a pointer to the clone, or NULL if cloning was unsuccessful.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Copies the contents of an object into the messaged object, if feasible. See also copyFrom() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus copyFrom(
    const AcRxObject* other
);
```

**Parameters**

- `other`  
  Input pointer to the object to copy from.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
virtual Acad::ErrorStatus dwgInFields(
   AcDbDwgFiler* pFiler
);

Parameters Description
pFiler Input filer to use to read the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbTileNameGeneratorSettingsGrid Class,
AcMapMbTileNameGeneratorSettingsGrid Class
AcMapMbTileNameGeneratorSettingsGrid:: dwgOutFields Method
AcMapMbTileNameGeneratorSettingsGrid Class |
AcMapMbTileNameGeneratorSettingsGrid Class

Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dwgOutFields(
    AcDbDwgFiler* pFiler
) const;
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pFiler</td>
<td>Input filer to use to write the object's data.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbTileNameGeneratorSettingsGrid Class,
AcMapMbTileNameGeneratorSettingsGrid Class
AcMapMbTileNameGeneratorSettingsGrid:: dxfInFields Method
AcMapMbTileNameGeneratorSettingsGrid Class |
AcMapMbTileNameGeneratorSettingsGrid Class

Lets this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

```
virtual Acad::ErrorStatus dxfInFields(
    AcDbDxfFiler* pFiler
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pFiler</td>
<td>Input filer to use to read the object's data.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Lets this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dxfOutFields(
    AcDbDxfFiler* pFiler
) const;
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pFiler</td>
<td>Input filer to use to write the object's data.</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns the primary indexer.

```cpp
Acad::ErrorStatus GetPrimaryIndex(
    AcMapMbTileNameGeneratorIndexer& primaryIndex
) const;
```

Parameters

<table>
<thead>
<tr>
<th>rowIndex</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Class encapsulating Index orientation and incrementing</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns the column index data member.

```cpp
Acad::ErrorStatus GetSecondaryIndex(
    AcMapMbTileNameGeneratorIndexer & secondaryIndex
) const;
```

Parameters

- `nColumnIndex`: Variable to hold value set.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Returns the separator data member.

```cpp
const ACHAR* Separator() const;
```

### Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sSeparator</td>
<td>Variable to hold value set</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the primary indexer.

```cpp
Acad::ErrorStatus SetPrimaryIndex(
    const AcMapMbTileNameGeneratorIndexer& primaryIndex
);
```

**Parameters**  **Description**

primaryIndex  Class encapsulating Index orientation and incrementing

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the primary indexer.

```cpp
Acad::ErrorStatus SetSecondaryIndex(
    const AcMapMbTileNameGeneratorIndexer& secondaryIndex
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>nColumnIndex</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Destroys an instance of this class.

```cpp
virtual ~AcMapMbTileNameGeneratorSettingsSequence();
```

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Constructs an instance of this class.

AcMapMbTileNameGeneratorSettingsSequence();
Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns the associated Name Generator.

```cpp
virtual AcMapMbTileNameGenerator* AllocateTileNameGenerator() const;
```

Returns Name Generator.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Clones the object. See also clone() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual AcRxObject* clone() const;
```

Returns a pointer to the clone, or NULL if cloning was unsuccessful.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links

AcMapMbTileNameGeneratorSettingsSequence Class,
AcMapMbTileNameGeneratorSettingsSequence Class
AcMapMbTileNameGeneratorSettingsSequence::copyFrom Method
AcMapMbTileNameGeneratorSettingsSequence Class |
AcMapMbTileNameGeneratorSettingsSequence Class

Copies the contents of an object into the messaged object, if feasible. See also copyFrom() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

```
virtual Acad::ErrorStatus copyFrom(
    const AcRxObject* other)
```

Parameters

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>other</td>
</tr>
</tbody>
</table>

Input pointer to the object to copy from.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbTileNameGeneratorSettingsSequence Class,
AcMapMbTileNameGeneratorSettingsSequence Class
AcMapMbTileNameGeneratorSettingsSequence:: dwgInFields Method
AcMapMbTileNameGeneratorSettingsSequence Class |
AcMapMbTileNameGeneratorSettingsSequence Class

Lets this object read its data. See also dwgInFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

virtual Acad::ErrorStatus dwgInFields(
    AcDbDwgFiler* pFiler
);

Parameters Description
pFiler Input filer to use to read the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbTileNameGeneratorSettingsSequence Class,
AcMapMbTileNameGeneratorSettingsSequence Class
AcMapMbTileNameGeneratorSettingsSequence:: dwgOutFields Method
AcMapMbTileNameGeneratorSettingsSequence Class |
AcMapMbTileNameGeneratorSettingsSequence Class

Lets this object write its data. See also dwgOutFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

virtual Acad::ErrorStatus dwgOutFields(
    AcDbDwgFiler* pFiler
) const;

Parameters Description
pFiler Input filer to use to write the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbTileNameGeneratorSettingsSequence Class
AcMapMbTileNameGeneratorSettingsSequence Class
AcMapMbTileNameGeneratorSettingsSequence:: dxfInFields Method
AcMapMbTileNameGeneratorSettingsSequence Class
AcMapMbTileNameGeneratorSettingsSequence Class

Lets this object read its data. See also dxfInFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

virtual Acad::ErrorStatus dxfInFields(
    AcDbDxfFiler* pFiler
);

Parameters                      Description
pFiler                           Input filer to use to read the object's data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbTileNameGeneratorSettingsSequence Class,
AcMapMbTileNameGeneratorSettingsSequence Class
AcMapMbTileNameGeneratorSettingsSequence:: dxfOutFields Method
AcMapMbTileNameGeneratorSettingsSequence Class |
AcMapMbTileNameGeneratorSettingsSequence Class

Lets this object write its data. See also dxfOutFields() in the AutoCAD ObjectARX Developer's Guide. This overridden function is called by the system as needed; it is unlikely that you will need to call it directly.

```cpp
virtual Acad::ErrorStatus dxfOutFields(
    AcDbDxfFiler* pFiler
) const;
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pFiler</td>
<td>Input filer to use to write the object's data.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the sequence indexer.

```cpp
Acad::ErrorStatus GetIndex(
    AcMapMbTileNameGeneratorIndexer& sequenceIndex
) const;
```

**Parameters**

- `sequenceIndex` Output class encapsulating Index orientation and incrementing

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the primary indexer.

```cpp
Acad::ErrorStatus SetIndex(
    const AcMapMbTileNameGeneratorIndexer& sequenceIndex
);
```

**Parameters**

- **sequenceIndex**
  
  *Output class encapsulating Index orientation and incrementing*

**Returns**

- Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
This is record AcMapMbTileSet::EMbNameSchema.

```
enum EMbNameSchema {
    kNumericSequence,
    kAlphaSequence,
    kByDataSequence,
    kGrid,
    kLastSchema
};
```

File

AcMapMbTileSet.h

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
This is record AcMapMbTileSet::EMbTreeType.

```cpp
enum EMbTreeType {
    kSequence,
    kAlpha,
    kRowCol,
    kColRow,
    kCustom
};
```

File

AcMapMbTileSet.h

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbTileSet Class
AcMapMbTile::: AcMapContainer Class
AcMapMbTileSet Class

Alternate interface for tree.

class AcMapContainer;
File
AcMapMbTileSet.h

Remarks

Since this tree is modeling a directory structure, some nodes are containers and some are tiles. Also some containers contain other containers. This alternate interface handles this and may be more easily understandable. It also allows the user to look at grids either bottom to top, or top to bottom; right to left, or left to right.

Methods

- `~AcMapContainer`: Destroys an instance of this class.
- `AcMapContainer`: Constructs an instance of this class.
- `GetContainerIterator`: Returns the container iterator for this container.
- `GetTileIterator`: Returns the tile iterator for this container.
- `Name`: Sets the Node name for the node associated with this container.
- `SetName`: Sets a node name for the node associated with this container.
- `TreeNode`: Returns the node associated with this container.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
This is class AcMapMbTileSet::AcMapContainerIterator.

```cpp
class AcMapContainerIterator;
```

File

AcMapMbTileSet.h

- **Methods**
  - `~AcMapContainerIterator` Destroys an instance of this class.
  - `AcMapContainerIterator` Constructs an instance of this class.
  - `Done` Tells the user if we are done.
  - `Get` Returns the next node in the chain as a container.
  - `Length` Returns the number of containers in the iterator.
  - `Rewind` Resets the iterator.
  - `Step` Moves to the next node in the chain as a container.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
This is class AcMapMbTileSet::AcMapTileIterator.

```cpp
class AcMapTileIterator;
```

File

AcMapMbTileSet.h

- **Methods**
  - `~AcMapTileIterator`_DESTROYS an instance of this class.
  - `AcMapTileIterator`_CONSTRUCTS an instance of this class.
  - `Done`_TELLS the user if we are done.
  - `Get`_Returns the current tile in the chain.
  - `GetNode`_Returns the node associated with the current tile in the chain.
  - `Length`_Returns the number of containers in the iterator.
  - `Rewind`_Resets the iterator.
  - `SetCurrentSecondaryIndexName`_Sets a node name for the node associated with this tile.
  - `Step`_Moves to the next tile in the chain.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Class AcMapTreeNode models the most general structure for an organization of tiles.

class AcMapTreeNode;

File

AcMapMbTileSet.h

Remarks

It resembles a directory structure. Every node contains either a tile or a child. The root node contains only a child. Thus a node may be thought of as either a container or a tile. The current organizations are either a 2 dimensional grid, or a sequence of tiles. For a grid, the root node contains one child, this child has siblings for each row in the grid, or each column in the grid, depending on what the user asked for. If the user asked for Row, Column then each sibling is a row container, that is each sibling has no tiles but has a child. The siblings of that child correspond to the columns, and each one has a tile pointer but no children. For a sequence, the root node contains one child. The child and all its siblings contain tiles. If the naming scheme is numeric, then its appropriate to ask for the sequence. If alphabetic, then its more appropriate to ask for an alphabetized list.

Methods

- ~AcMapTreeNode: Destroys an instance of this class.
- AcMapTreeNode: Constructs an instance of this class.
- AddChild: Sets a child for this node.
- AddSibling: Sets a sibling for this node.
- Child: Returns the child of this node.
- ConstChild: Returns the const child of this node.
- ConstSibling: Returns the const sibling of this node.
- ConstTile: Returns the const tile of this node.
- Name: Sets the Node name.
**SetName**
Sets a node name.

**SetSibling**
Sets a sibling for this node.

**SetTile**
Sets the tile for this node.

**Sibling**
Returns the sibling of this node.

**Tile**
Returns the tile of this node.

**TileSet**
Returns the owner tile set of this node.

Created with a commercial version of [Doc-O-Matic](https://toolsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

```cpp
virtual ~AcMapMbTileSet();
```

Returns

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbTileSet Class, AcMapMbTileSet Class
AcMapMbTileSet:: AcMapMbTileSet Constructor
AcMapMbTileSet Class | AcMapMbTileSet Class

Constructs an instance of this class.

AcMapMbTileSet(
    AcDbDatabase * pDb
);
Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the new tiles into the tile set.

```cpp
def AddTiles(pTree, treeType):
    # Function implementation
```

**virtual Acad::ErrorStatus AddTiles(**
    ```cpp
    AcMapTreeNode* pTree,
    EMbTreeType treeType
    ```
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pTree</td>
<td>Input tree of tiles and nodes. The structure of this tree is based on the tree nodes, children, and siblings described above.</td>
</tr>
<tr>
<td>treeType</td>
<td>Input type of tree of tiles and nodes. A kSequential or kAlpha tree should be flat, while a kRowCol or kColRow tree should be a grid with depth 2. kCustom tree may have a more complex tree structure. A kRowCol tree may be viewed as kColRow and vice versa. All trees may be viewed as sequential with some ordering. kCustom trees may not be viewed as kRowCol or kColRow, but may be viewed as sequential. A breadth first search order is assumed.</td>
</tr>
</tbody>
</table>

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbTileSet Class, AcMapMbTileSet Class
AcMapMbTileSet:: BuildAdjacentInfo Method
AcMapMbTileSet Class | AcMapMbTileSet Class

Generate adjacency information and assigns it to every tile.

`Acad::ErrorStatus BuildAdjacentInfo();`

Returns

Returns `Acad::eOk` if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://example.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Returns the number of columns in a grid tiling.

```cpp
virtual unsigned int ColumnNumber() const;
```

Returns the number of columns in a grid tiling. Returns 0 if this is not a grid tiling.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns the root of the tree organized according to the tree type as a container.

.virtual const AcMapContainer* ConstContainer(EMbTreeType howToGet) const;

Parameters Description
howToGet Input type of tree to return.

Returns

Returns the const container if successful; otherwise, returns NULL to indicate the tree has not been set, or the tree type is invalid.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbTileSet Class, AcMapMbTileSet Class
AcMapMbTileSet:: ConstTree Method
AcMapMbTileSet Class | AcMapMbTileSet Class

Returns the root of the tree organized according to the tree type.

```cpp
virtual const AcMapTreeNode* ConstTree(
    EMbTreeType howToGet
) const;
```

Parameters

- howToGet: Input type of tree to return.

Returns

Returns the const root tree node if successful; otherwise, returns NULL to indicate the tree has not been set, or the tree type is invalid.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Creates a new node for the tree.

```cpp
virtual AcMapTreeNode* CreateNode(
    bool bRootNode
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bRootNode</td>
<td>Input boolean indicating whether this node is the root or a branch on the tree.</td>
</tr>
</tbody>
</table>

Returns

Returns a tree node if successful; otherwise, returns NULL.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at **support@toolsfactory.com**.
Returns the pointer to the drawing database the tile set is associated with.

```cpp
AcDbDatabase* Database() const;
```

Returns a valid database pointer if succeeded, NULL if the object is not initialized yet.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Removes the specified tile from the tile set.

```cpp
Acad::ErrorStatus Erase(
    AcMapMbMapBookManager::TILEID id,
    bool bEraseSheet
);
```

**Parameters**

- `id`: Input tile id.
- `bEraseSheet`: Input flag to erase the corresponding sheet.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

**Remarks**

Erases all the corresponding objects including a Sheet.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Returns the tile corresponding to the given tile id if it exists.

```cpp
virtual AcMapMbTile* Find(
    AcMapMbMapBookManager::TILEID id);
```

**Parameters**

- **id**
  - Description: Input tile id.

**Returns**

Returns the const tile if successful; otherwise, returns a NULL.

**Remarks**

If the corresponding Map Book is open for read, the client can call only the
constant methods; nonconstant methods require the Map Book being opened for
write. Use the retrieved pointer only while the Map Book object is opened.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this
message disappear you need to register this software. If you have problems
registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Returns the total extents of the view ports of this tile set.

```cpp
virtual Acad::ErrorStatus GetExtents(
    AcDbExtents& extents
) const;
```

**Parameters**
- `extents`  

**Description**
- Output extents.

**Returns**
- Returns true if successful; otherwise, returns false.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns the implementation object.

\texttt{virtual AcMapMbImpTileSet* Implementation();}

Returns the implementation, or NULL if unsuccessful.

Remarks

This function is for internal use only; do not call it directly.

Created with a commercial version of \texttt{Doc-O-Matic}. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at \texttt{support@toolsfactory.com}.
When a tile set gets generated (except manual), it creates polylines per every tile and puts them to the specified layer.

```cpp
AcDbObjectId LayerId() const;
```

Returns

Id of the AcDbLayerTableRecordObject or AcDbObjectId::kNull in the case of manual grid.

Remarks

This method retrieves Id of this layer.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns the Id of the AcMapMbMapBook object it belongs to.

`AcMapMbMapBook* MapBook() const;`

Returns a valid Id if succeeded, NULL Id if the object is not initialized yet or is not in the database.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns the way or schema used when naming the tiles.

```cpp
virtual EMbNameSchema NamingScheme() const;
```

Returns an enum indicating how the tiles were named.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links

AcMapMbTileSet Class, AcMapMbTileSet Class
AcMapMbTileSet:: RowNumber Method
AcMapMbTileSet Class | AcMapMbTileSet Class

Returns the number of rows in a grid tiling.

virtual unsigned int RowNumber() const;
Returns

Returns the number of rows in a grid tiling. Returns 0 if this is not a grid tiling.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Iterates a Tile Set and sets adjacent properties for the corresponding Sheets.

```
Acad::ErrorStatus SetAdjacentProperties();
```

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbTileSet Class, AcMapMbTileSet Class
AcMapMbTileSet:: StreamIn Method
AcMapMbTileSet Class | AcMapMbTileSet Class

Reads data from a dwg file.

virtual Acad::ErrorStatus StreamIn(
   AcDbDwgFiler* pFiler
);

Parameters              Description
pFiler                  Input filer. The filer should be open and ready to read this tile data from.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbTileSet Class, AcMapMbTileSet Class
AcMapMbTileSet:: StreamIn Method
AcMapMbTileSet Class | AcMapMbTileSet Class

Reads data from a dxf file.

**virtual** Acad::ErrorStatus StreamIn(
    AcDbDxfFiler* pFiler
);

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pFiler</td>
<td>Input filer. The filer should be open and ready to read this tile data from.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Loads data into a dwg file.

```cpp
virtual Acad::ErrorStatus StreamOut(
    AcDbDwgFiler* pFiler
)
const;
```

Parameters | Description
--- | ---
pFiler | Input filer. The filer should be open and ready to write this tile data.

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Loads data into a dxf file.

```cpp
virtual Acad::ErrorStatus StreamOut(
    AcDbDxfFiler* pFiler
) const;
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pFiler</td>
<td>Input filer. The filer should be open and ready to write this tile data.</td>
</tr>
</tbody>
</table>

Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapMbTileSet Class, AcMapMbTileSet Class
AcMapMbTileSet:: TileNumber Method
AcMapMbTileSet Class | AcMapMbTileSet Class

Returns the total number of tiles in the set.

virtual unsigned int TileNumber() const;

Returns the total number of tiles in the set

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbTileSet Class, AcMapMbTileSet Class
AcMapMbTileSet:: TilingScheme Method
AcMapMbTileSet Class | AcMapMbTileSet Class

Returns the schema or organization used when creating the tiles.

**virtual** EMBTreeType TilingScheme() const;
Returns

Returns an enum indicating how the tiles are organized.

Remarks

Once the tiles are created, they may be viewed in the same format they were created, or in a simpler way. For instance, any tree may be viewed as a sequential tree. Any RowCol tree may be viewed as a ColRow tree and vice versa. The following trees and containers are available based on the tree type given in the AddTiles method.

AddTiles ConstContainer, ConstTree or Tree kSequence: kSequence, kAlpha kAlpha: kSequence, kAlpha kRowCol: kRowCol, kColRow, kSequence, kAlpha kColRow: kRowCol, kColRow, kSequence, kAlpha kCustom: kCustom, kSequence, kAlpha.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns the root of the tree organized according to the tree type.

```cpp
virtual AcMapTreeNode* Tree(
    EMbTreeType howToGet
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>howToGet</td>
<td>Input type of tree to return.</td>
</tr>
</tbody>
</table>

Returns

Returns the root tree node if successful; otherwise, returns NULL to indicate the tree has not been set, or the tree type is invalid.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
map_dwgbreakobj

Map Boundary Functions

Breaks linear objects where they cross boundaries.

```c
int map_dwgbreakobj(
    ads_name ssCut,
    ads_name boundary,
    short skipTopo,
    short keepod,
    ads_name ssreturn);
```

Returns **RTNORM** or **RTERROR**.

- `ssCut` Selection set of objects to cut.
- `boundary` Object name of a single object or a selection set of multiple objects. Valid objects: line, polyline, circle, arc.
- `skipTopo` Skip flag:
  - 1 Skip objects referenced by a topology
  - 0 Trim objects referenced by a topology
- `keepod` Keep flag that sets whether to keep object data of clipped objects in result object:
  - 1 Retain all object data on any clipped object
  - 0 Drop object data on any clipped object
- `ssreturn` A selection set of cut objects.

This function cuts linear objects, such as lines, polylines, circles, and arcs, that cross the selected boundary. Unlike the `map_dwgtrimobj` function, this function does not delete the parts of the object on either side of the boundary. For example, you could mark a boundary and divide one map into two section maps along this boundary.

The following sample creates a filtered selection set of objects to be cut, (broken) and a filtered selection set containing one object to be used as the cutting edge. `Map_dwgBreakObj()` is called with all required parameters and if the operation is successful, the number of objects retained is reported. `Resbufs` are released as required and selection sets are freed.
struct resbuf* pFilteredCutObjsRb = acutBuildList(
    RTDXF0, "LWPOLYLINE",
    8, "Landbase",
    0);

ads_name ssObjsToCut;
acedSSGet("_X", NULL, NULL, pFilteredCutObjsRb, ssObjsToCut);

struct resbuf* pFilteredObjsToCutRb = acutBuildList(
    RTDXF0, "LWPOLYLINE",
    8, "MapEdge",
    0);

ads_name boundaryObj;
acedSSGet("X", NULL, NULL, pFilteredObjsToCutRb, boundaryObj);

short skipTopoObjs = 1; // Skip
short keepObjectData = 1; // Retain
ads_name ssObjsAreCut;

int resultCode = map_dwgBreakObj(
    ssObjsToCut,
    boundaryObj,
    skipTopoObjs,
    keepObjectData,
    ssObjsAreCut);

if (RTNORM == resultCode){
    long ssObjsAreCutLength;
    acedSSLength(ssObjsAreCut, &ssObjsAreCutLength);
    acutPrintf("\n%d objects have been retained."
        , ssObjsAreCutLength);
} else {
    acutPrintf("\nUnable to complete the boundary break.");
}
acutRelRb(pFilteredCutObjsRb);
acutRelRb(pFilteredObjsToCutRb);
resultCode = acedSSFree(boundaryObj);
map_dwgtrimobj

Map Boundary Functions

Trims linear objects inside or outside of a specified boundary.

int
map_dwgtrimobj(
    ads_name ssclip,
    ads_name boundary,
    int inorout,
    int skiptopo,
    int keepod,
    int bitflag,
    ads_name ssreturn);

Returns RTNORM or RERROR.

ssclip Selection set of objects to trim.
boundary Entity name of a boundary object. Valid objects: a single circle or a single closed 2D polyline.
inorout Trim flag:
   1 Trim outside boundary
   0 Trim inside boundary
skiptopo Skip flag:
   1 Skip objects referenced by a topology
   0 Trim objects referenced by a topology
keepod Keep flag that sets whether to keep object data of trimmed objects in result object:
   1 Drop object data from all trimmed objects
   0 Retain object data on all trimmed objects
bitflag Bit flag that sets the way to handle objects that cannot be trimmed:
   0 Delete these objects within or on trim boundary
   1 Ignore these objects within or on trim boundary
   2 Reference the insertion point of any of these objects within or on trim boundary
ssreturn A selection set of trimmed objects.
The following example prompts you to select an object to trim and make choices about the trim operation. It includes error reporting.

```c
ads_name ssCut, boundary, result;
ads_point point;
int ret, nb, i, skiptopo, keepod, inorout, bitflag;
char kword[50];

ads_printf("\nSelect objects to trim : ");
if (ads_ssget(NULL, NULL, NULL, NULL, NULL, ssCut) != RTNORM)
    return;

if (ads_entsel("\nSelect boundary object : ", boundary, point) != RTNORM) {
    ads_ssfree(ssCut);
    return;
}

ads_initget(0, "Yes No");
ret = ads_getkword("\nSkip objects referenced by a topology Yes/No <Yes> : ", kword);
if (ret != RTNORM && ret != RTNONE) {
    ads_ssfree(ssCut);
    return;
}

if (ret == RTNONE || kword[0] == 'Y')
    skiptopo = 1;
else
    skiptopo = 0;

ads_initget(0, "Yes No");
ret = ads_getkword("\nRetain object data Yes/No <Yes> : ", kword);
if (ret != RTNORM && ret != RTNONE) {
    ads_ssfree(ssCut);
    return;
}

if (ret == RTNONE || kword[0] == 'Y')
    keepod = 1;
else
    keepod = 0;

ads_initget(RSG_NONULL, "Inside Outside");
```
ret = ads_getkword("\nTrim Inside or Outside : ", kword);
if (ret != RTNORM) {
    ads_sssfree(sscut);
    return;
}

if (kword[0] == 'I')
    inorout = 0;
else
    inorout = 1;

ads_initget(0, "Delete Ignore insertionPoint");
ret = ads_getkword("\nObjects that cannot be trimmed Delete/Ignore/insertionPoint : ", kword);
if (ret != RTNORM && ret != RTNONE) {
    ads_sssfree(sscut);
    return;
}

if (ret == RTNONE || kword[0] == 'P')
    bitflag = 2;
else if (kword[0] == 'I')
    bitflag = 1;
else
    bitflag = 0;

ret = map_dwgtrimobj(sscut, boundary, inorout, skiptopo, keepod, bitflag, result);
if (ret != RTNORM) {
    ads_printf("\nObject(s) successfully trimmed.");
    ads_sssfree(result);
}
else {
    nb = ade_errqty();
    for (i = 0; i < nb; i++)
        ads_printf("\nError %d of %d: %s", i + 1, nb, ade_errmsg(i));
}
ads_sssfree(sscut);
map_pltblkatts

Plotting Functions

Gets a list of block attributes.

```c
int map_pltblkatts(
    char *name,
    struct resbuf **res);
```

Returns RTNORM or RTERRO.

name Layout block name.

res Pointer to pointer to result buffer of block attributes.

If the function returns RTNORM, you must release the resbuf.

The following sample populates a resbuf with plot layout block attribute name(s) using map_pltBlkAtts(). If the operation is successful the attribute name(s) are displayed, and the resbuf is released as required.

```c
char* pszLayoutBlockName = "TITLE-A";
struct resbuf* pLayoutBlockAttsRb = NULL;
int returnCode = map_pltBlkAtts(pszLayoutBlockName, &pLayoutBlockAttsRb);
if (RTNORM == returnCode){
    struct resbuf* rb = pLayoutBlockAttsRb;
    while(NULL != rb) {
        acutPrintf("The following block attribute has been found on the layout block \"%s\": %s",
                   pszLayoutBlockName, rb->resval.rstring);
        rb = rb->rbnext;
    }
}
else {
    acutPrintf("No block attributes have been found on the layout block \"%s\".");
}
acutRelRb(pLayoutBlockAttsRb);
```
map_pltblklist

Plotting Functions

**map_pltblklist**

**Plots Functions**

Gets a list of valid plot layouts for the current work session.

```c
struct resbuf
*map_pltblklist();
```

Returns a result buffer of block names or **NULL**

You must release the **resbuf**.

This function returns the plot layout (block) names that are usable as plot layouts. To qualify, a block must have at least one unique viewport on one of its layers. That is, if the block has more than one viewport, it must have one layer that contains only one viewport.

The unique viewport can share its layer with objects of other types, such as lines, polylines, blocks, and text.

The following sample populates a **resbuf** with the plot layout block name using **map_pltBlkList**(). If the operation is successful the layout block name is displayed, and the **resbuf** is released as required.

```c
struct resbuf* pLayoutBlockNameRb = NULL;
pLayoutBlockNameRb = map_pltBlkList();
if(pLayoutBlockNameRb != NULL) {
    acutPrintf("The plot layout block is named: %s.\n", pLayoutBlockNameRb->resval.rstring);
} else {
    acutPrintf("No plot layout blocks are defined.\n");
}
acutRelRb(pLayoutBlockNameRb);
```
map_pltblkvps

**Plotting Functions**

Returns a list of valid viewport layers in a specified layout blocks.

```c
int map_pltblkvps(
    char *name,
    struct struct resbuf **res);
```

Returns **RTNORM** if the list is initialized or **RERROR** if the block name or layout block is invalid.

- **name** Layout block name.
- **res** Pointer to pointer to result buffer of viewport layers.

If the function returns **RTNORM**, you must release the **resbuf**.

The following sample populates a **resbuf** with plot layout block viewport layer name(s) using **map_pltBlkVps()**. If the operation is successful the viewport layer name(s) are displayed, and the **resbuf** is released as required.

```c
char* pszLayoutBlockName = "TITLE-A";
struct resbuf* pLayoutBlockVportLayerRb = NULL;
int returnCode = map_pltBlkVps(pszLayoutBlockName, &pLayoutBlockVportLayerRb);
if (RTNORM == returnCode){
    struct resbuf* rb = pLayoutBlockVportLayerRb;
    while(NULL != rb) {
        acutPrintf("\nThe following viewport layer(s) have been found on the layout block "%s": %s",
            pszLayoutBlockName, rb->resval.rstring);
        rb = rb->rbnext;
    }
} else {
    acutPrintf("\nNo viewport layers have been found on the layout block "%s".");
}
```
acutRelRb(pLayoutBlockVportLayerRb);
Restores settings altered by map_pltinit.

int
map_pltcleanup();

Returns RTNORM or RERROR.

This function restores certain settings to the state they were in before map_pltinit was called. See map_pltinit for a list of affected settings. Before you can use other plot functions, you must call map_pltinit again.
Selects or creates a plot set.

```c
int map_pltcurrence(char *name);
```

Returns **RTNORM** or **RTERROR**.

- **name**  The name of the plot set.

If a plot set called **name** does not exist in the current work session or if the function **map_pltdefread** was not called, this function creates a new plot set called **name**.

Use **map_pltcurrset** to define the attributes of the plot set.

If the plot set **name** exists and the function **map_pltdefread** was called, this function loads a copy of **name** into memory. Use **map_pltcurrenceGet** and **map_pltcurrenceSet** to examine or change its attributes.

**Note**  Only one plot set can be current.
map_pltcurrdel

_Plotting Functions_

Resets a plot set attribute to its default value.

```c
int map_pltcurrdel(char *attr);
```

Returns _RTNORM_ or _RTERROR_.

**attr**    The name of the attribute to reset.
Map_pltcurrget

Plotting Functions

Gets an attribute value for the current plot set.

int
map_pltcurrget(
    char *attr,
    struct struct resbuf **res);

Returns RTNORM or RTERROR.

attr The name of the plot set attribute to retrieve.
res Resbuf containing the value of the attribute.

This function retrieves the value of a specific attribute for the current plot set. The data type of the return value depends upon the attribute. To set plot set attributes, use map_pltdefset.

If the function returns RTNORM, you must release the resbuf.

The following sample populates a resbuf with the plot query name using map_pltCurrGet(). If the operation is successful the query name is displayed, and the resbuf is released as required.

```c
char* pszPlotSetAttrb = "name";
struct resbuf* pPlotSetAttrbValueRb = NULL;
int returnCode = map_pltCurrGet(pszPlotSetAttrb, &pPlotSetAttrbValueRb);
if (RTNORM == returnCode){
    acutPrintf("The plot set attribute \"%s\" contained the following value: \n\t\"%s\"
    , pszPlotSetAttrb, pPlotSetAttrbValueRb->resval.rstring);
}
else {
    acutPrintf("The plot set attribute \"%s\" contained no value."
    , pszPlotSetAttrb);
}
acutRelRb(pPlotSetAttrbValueRb);
```
map_pltcurrsave
Plotting Functions

Appends the current plot set definition to the plot set list.

int
map_pltcurrsave();

Returns RTNORM or RTERROR.

This function appends the current plot set definition to the plot set list of the current work session. This list is not saved in the work session until the user executes a save with a call to map_pltdefsave.

To edit and save a plot set definition

1. Get the plot set definition with map_pltdefread and map_pltcurrdef.
2. Make the necessary changes.
3. Save the definition in the list with map_pltcurrsave.
4. Save the list in the current work session with map_pltdefsave.
5. Save the work session.

If you do not save before the end of the processing, another application can overwrite your changes with a call to map_pltcurrdef.
Sets the value of an attribute for the current plot set.

```c
int map_pltcurrset(
    char attr,
    struct resbuf *res);
```

Returns **RTNORM** or **RTERROR**.

- **attr** The name of the plot attribute to set.
- **res** Pointer to `resbuf` for attribute value.

If the function returns **RTNORM**, you must release the `resbuf`.

The following sample creates a `resbuf` containing a value representing the plot set description. **Map_pltCurrSet()** is called with all required parameters, if the operation is successful the plot set attribute name and the updated value are displayed. The `resbuf` is then released as required.

```c
char* pszPlotSetAttrb = "desc";
struct resbuf* pPlotSetAttrbValueRb = acutBuildList(
    RTSTR,
    "A new plot set description",
    0);
int returnCode = map_pltCurrSet(
    pszPlotSetAttrb,
    pPlotSetAttrbValueRb);
if (RTNORM == returnCode){
    acutPrintf(
        "\nThe plot set attribute \"%s\" has been set to the following value: \n\n\t\"%s\"
    , pszPlotSetAttrb, pPlotSetAttrbValueRb->resval.rstring);
}
else {
    acutPrintf(
        "\nThe plot set attribute \"%s\" was not set."
```
, pszPlotSetAttrb);
}
acutRelRb(pPlotSetAttrbValueRb);
map_pltdefdelete

Plotting Functions

Deletes a plot set definition.

int
map_pltdefdelete(
    char *name);

Returns RTNORM or RTERROR.

name The name of the plot set.

The function updates the plot definition dictionary in the current work session.
map_pltdefget

Plotting Functions

Gets the value of an attribute of the plot set definition.

```c
int map_pltdefget(
    char *name,
    char *attr,
    struct resbuf **res);
```

Returns **RTNORM** or **RTERROR**.

- **name**: The name of the plot set.
- **attr**: Name of the plot set attribute to retrieve.
- **res**: Pointer to a pointer to the resbuf containing the given plot set attribute

This function returns the value of the attribute through the resbuf parameter. If the function returns **RTNORM**, you must release the resbuf.

To get the value of an attribute for the current plot set, use `map_pltCurrGet`.

The data type of the return value depends on the attribute.

The following sample populates a resbuf with the value of the plot set description for a specified plot set. `Map_pltDefGet()` is called with all required parameters, and if the operation is successful the specified plot set and description are displayed. The resbuf is then released as required.

```c
char* pszPlotSetName = "ADSRX_Sample";
char* pszPlotSetAttrb = "desc";
struct resbuf* pPlotSetAttrbValueRb = NULL;
int returnCode = map_pltDefGet(pszPlotSetName, pszPlotSetAttrb, &pPlotSetAttrbValueRb);
if (RTNORM == returnCode){
    acutPrintf(
        "%nThe %s plot set attribute; %s contains the following value: %s"
        , pszPlotSetName, pszPlotSetAttrb, pPlotSetAttrbValueRb->resval.rstring);
}
```
else {
    acutPrintf("\nThe plot set attribute \"%s\" could not be obtained.\n", pszPlotSetAttrb);
}
acutRelRb(pPlotSetAttrbValueRb);
**map_pltdeflist**

*Plotting Functions*

Gets the names of available plot set definitions.

```c
int
map_pltdeflist(
    struct resbuf **res);
```

Returns **RTNORM** or **RTERROR**.

*res* Pointer to pointer to *resbuf* for list of returned plot set definitions.

This function returns the value of the attribute through the *resbuf* parameter. If the function returns **RTNORM**, you must release the *resbuf*.

The following sample populates a *resbuf* with the plot set names using *map_pltDefList().* If the operation is successful the plot set name(s) are displayed, and the *resbuf* is released as required.

```c
struct resbuf* pPlotSetListRb = NULL;
int returnCode = map_pltDefList(&pPlotSetListRb);
if (RTNORM == returnCode){
    struct resbuf* rb = pPlotSetListRb;
    acutPrintf("\nThe current project contains the following plot set(s): ");
    while(NULL != rb) {
        acutPrintf("\n\"%s\", rb->resval.rstring);
        rb = rb->rbnext;
    }
} else {
    acutPrintf("\nThe current project contains no plot sets.\n");
}
acutRelRb(pPlotSetListRb)
```
map_pltdefread

Plotting Functions

Reads in a plot set definition.

```
int
map_pltdefread();
```

Returns RTNORM or RTERROR.

This function provides access to plot sets in the plot definition dictionary for the current work session.

**Warning** If you do not call map_pltdefsave, a new call to map_pltdefread will erase your new plot set definition or your changes to an existing plot set.
map_pltdefsave

**Plotting Functions**

Writes the current plot set definition to the plot definition dictionary.

```c
int
map_pltdefsave(
    );
```

Returns RTNORM or RERROR.

This function stores the plot set definition list in the current work session.

If you do not call this function, a new call to map_pltdefread will erase your new plot set definition or your changes to an existing plot set.
map_pltdefvalid

Plotting Functions

Performs a cursory check of the validity of a plot set.

```
int map_pltdefvalid(
    char *name);
```

Returns RTNORM or RERROR.

name  The name of the plot set.

The function performs a cursory check of the given plot set to see if all required attributes have been set. It does not attach and query the boundary drawing. When used within a dialog box, this call can quickly check on a plot set's usability.

If the check fails, you can use data extension error message functions (ade_err[xx]) to retrieve errors from the error stack.

To perform an extensive check, use the map_pltdefverify function.
map_pltdefverify

**Plotting Functions**

Performs an extensive check of the validity of a plot set.

```c
int
map_pltdefverify(
    char *name);
```

Returns **RTNORM** or **RTERROR**.

**name** The name of the plot set.

This function makes sure that no errors occur when plotting takes place. Since the check includes querying for all boundary objects, it could take some time.

To perform a cursory check, use the `map_pltdefvalid` function.

If the check fails, you can use data extension error message functions (`ade_err[xx]`) to retrieve errors from the error stack.
map_pltdisplay
Plotting Functions

Generates the plot display for the specified boundary.

int
map_pltdisplay(
    char *bndryname);

Returns RTNORM or RERROR.

bndryname The name of a boundary object.

This function prepares the display to plot for the given plot set definition and boundary object name. Generating the plot display for the specified boundary includes

- Switching to paper mode (if necessary)
- Inserting the layout block
- Mapping boundary object data to layout block attributes (if applicable)
- Executing the query(ies) to collect the objects to plot
- Trimming the objects to the boundary (if applicable)
- Displaying the objects in the view port

You must call map_pltdisplay before you call map_pltplot.
Plots the plot set for the specified plot set name.

```c
int
map_pltexecute(
    char *name);
```

Returns **RTNORM** or **RTERROR**.

- **name** The name of the plot set to execute.

This function generates and issues plots for each defined boundary.

To get a list of available plot set names, use **map_pltdeflist**.
map_pltinit

Plotting Functions

Initializes environment for plotting.

```c
int map_pltinit();
```

Returns `RTNORM` always.

This function must be called before any other plotting functions. The `map_pltinit` function modifies the following settings and checks that the plot set description file exists:

<table>
<thead>
<tr>
<th>AutoCAD Variable</th>
<th>Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>CMDECHO</td>
<td>0</td>
</tr>
<tr>
<td>expert</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Data Extension Preference</th>
<th>Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>DontAddObjectsToSaveSet</td>
<td>T</td>
</tr>
<tr>
<td>ActivateDwgsOnAttach</td>
<td>T</td>
</tr>
<tr>
<td>MkSelSetWithQryObj</td>
<td>T</td>
</tr>
</tbody>
</table>

To restore these settings to their original values, use `map_pltcleanup`. Before you can use other plot functions, you must call `map_pltinit` again.
map_pltplot

**Plotting Functions**

Executes the current plot set script.

```c
int
map_pltplot();
```

Returns **RTNORM** or **RERROR**.
map_pltrestore

Plotting Functions

Restores display altered by map_pltdisplay.

int
map_pltrestore();

Returns RTNORM or RTERROR.
map_topoaudit
Map Topology Functions

Determines whether the specified topology is correct.

```c
int map_topoaudit(
    ade_id tpm_id);
```

Returns **RTNORM** or **RERROR**.

**tpm_id**  The ID of the topology to check for correctness. The topology must be open for Read.

The function audits the geometry of a topology to determine whether the geometrical relationships defined by the topology object data are correct. It shows the location of errors.

The following sample checks a topology named "I-95_Buffer" and provides error reporting based on the results of the check.

```c
char* pszTopoName = "I-95_Buffer";
int resultCode = tpm_acload(pszTopoName, 0);
ade_id topoId = tpm_acopen(
    pszTopoName,
    0);
if (ADE_NULLID != topoId)
{
    resultCode = ade_errclear();
    int resultCode = map_topoAudit(topoId);
    if (RTNORM == resultCode) {
        acutPrintf(
            "\nTopology %s is correct and complete.\n", pszTopoName);
    }
    else {
        int errorStkQnty = ade_errqty();
        if (errorStkQnty > 0)
        {
```
for (int i = 0; i < errorStkQnty; i++)
{
    int nErrCode = ade_errcode(i);
    char* pszErrorMsg = adeErrMsg(i);
    acutPrintf ("\n%", pszErrorMsg);
}

else {
    acutPrintf ("The topology was not opened.");
}

resultCode = tpm_acclose(topoId);
resultCode = tpm_acunload(pszTopoName);
map_topoclose

Map Topology Functions

Converts all polygons in a topology to closed polylines.

```c
int map_topoclose(
    char *topoName,
    char *layer,
    int group,
    int odata,
    int aselink,
    ads_name ssreturn);
```

Returns **RTNORM** or **RTERROR**.

- **toponame**: The name of a topology. The topology must be closed.
- **layer**: The name of the layer that will contain the closed polylines. If the layer name is incorrect (for example "@@"), polylines are created on the current layer.
- **group**: Group flag for complex polygons: 1 or 0.
  - 1 group complex polygons
  - 0 do not group complex polygons
- **odata**: Object data flag that sets whether to copy object data on the polygon centroid to the resulting polyline: 1 or 0.
  - 1 copy object data on the centroid
  - 0 ignore object data on the centroid
- **aselink**: ASE link flag that sets whether to copy ASE link data on the centroid to the resulting polyline: 1 or 0.
  - 1 copy ASE link data on the centroid
  - 0 ignore ASE link data on the centroid
- **ssreturn**: The returned selection set containing the closed polylines.

This function passes the selection set through the **ssreturn** parameter. If the function returns **RTNORM**, you must release the returned selection set with the **ads_ssfree** function.
map_topocomplete

Map Topology Functions

Completes all objects in a loaded partial topology.

int
map_topocomplete(
    char *toponame,
    ads_name ssreturn);

Returns RTNORM or RTERROR.

toponame The name of the topology to complete. The topology must be closed.

ssreturn The returned selection set of objects retrieved to complete the topology.

This function returns the selection set through the ssreturn parameter. If the function returns RTNORM, you must release the returned selection set with the ads_sssfree function.

This function performs a query to retrieve objects into the work session. The tpm_infocomplete function determines whether the topology is completely represented in the work session.

This function can only complete objects imported from an existing source drawing. For a polygon topology, this function imports links, nodes, and a centroid, if it is missing, from the source drawing. For a network topology, this function imports links and nodes. For a network topology, it imports only nodes.
map_topostat

Map Topology Functions

Gets statistics about a topology.

```
struct resbuf
*map_topostat(
    ade_id tpm_id);
```

Returns a resbuf list containing the statistics for the specified topology, or NULL if an error occurs.

**tpm_id**  The ID of the topology for which to get statistics. The topology must be open for read.

You must release the resbuf.

The following tables show statistic names, the data type and the topology types they apply to:

### All Topologies

<table>
<thead>
<tr>
<th>Statistic</th>
<th>Data Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>node_count</td>
<td>RTSHORT</td>
</tr>
<tr>
<td>link_count</td>
<td>RTSHORT</td>
</tr>
<tr>
<td>polygon_count</td>
<td>RTSHORT</td>
</tr>
<tr>
<td>min_x</td>
<td>RTREAL</td>
</tr>
<tr>
<td>min_y</td>
<td>RTREAL</td>
</tr>
<tr>
<td>max_x</td>
<td>RTREAL</td>
</tr>
<tr>
<td>max_y</td>
<td>RTREAL</td>
</tr>
</tbody>
</table>

**Note**  This function is not designed to count polygons in a partial topology. If the topology in question is partial, the polygon_count statistic may be overstated. This is because map_topostat counts not only the polygons in the partial topology, but also any polygons that share common edges with them in the complete topology, even if the adjacent polygons are not actually present in the current drawing.

### Network Topologies
<table>
<thead>
<tr>
<th>Polygon Topologies</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>area_total</td>
<td>RTREAL</td>
</tr>
<tr>
<td>area_average</td>
<td>RTREAL</td>
</tr>
<tr>
<td>area_min</td>
<td>RTREAL</td>
</tr>
<tr>
<td>area_max</td>
<td>RTREAL</td>
</tr>
<tr>
<td>area_variance</td>
<td>RTREAL</td>
</tr>
<tr>
<td>area_deviation</td>
<td>RTREAL</td>
</tr>
<tr>
<td>perimeter_total</td>
<td>RTREAL</td>
</tr>
<tr>
<td>perimeter_average</td>
<td>RTREAL</td>
</tr>
<tr>
<td>perimeter_min</td>
<td>RTREAL</td>
</tr>
<tr>
<td>perimeter_max</td>
<td>RTREAL</td>
</tr>
<tr>
<td>perimeter_variance</td>
<td>RTREAL</td>
</tr>
<tr>
<td>perimeter_deviation</td>
<td>RTREAL</td>
</tr>
</tbody>
</table>

The following sample loads and then opens a network topology for read using `tpm_acload()` and `tpm_acopen()` respectively. A `resbuf` is populated with that topologies information, then is displayed. The `resbuf` is released as required.

```c
char* pszTopoName = "NetTopo";
int topoWriteAccess = 0;
int returnCode = tpm_acload(pszTopoName, 0);
ade_id topoId = tpm_acopen(pszTopoName, topoWriteAccess);
struct resbuf* pTopoStatsRb = map_topoStat(topoId);
if (NULL != pTopoStatsRb) {
```
struct resbuf* rb = pTopoStatsRb;
while(NULL != rb) {
    if (rb->restype == RTSTR) {
        acutPrintf(  
            "\nThe \"%s\" property contained the value:"  
            , rb->resval.rstring);
        if (NULL != (rb = rb->rbnext)) {
            if (rb->restype == RTDOTE) {
                rb = rb->rbnext;
                switch(rb->restype)
                {
                    case RTSTR:
                        acutPrintf(  
                            " \"%s\" RTSTR"  
                            , rb->resval.rstring);
                        break;
                    case RTREAL:
                        acutPrintf(  
                            " %.2lf RTREAL"  
                            , rb->resval.rreal);
                        break;
                    case RTSHORT:
                        acutPrintf(  
                            " %d RTSHORT"  
                            , rb->resval.rint);
                        break;
                    case RTLONG:
                        acutPrintf(  
                            " %d RTLONG"  
                            , rb->resval.rlong);
                        break;
                    default:
                        break;
                }
            }
        }
        rb = rb->rbnext;
    } else {
        acutPrintf(  
            "\nThe \"%s\" property contained the value:"  
            , rb->resval.rstring);
    }
}
"No topologies are accessible in this project."

acutRelRb(pTopoStatsRb);
tpm_acclose(topoId);
tpm_acunload(pszTopoName);
The map boundary functions begin with `map_dwg`.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>map_dwgbreakobj</code></td>
<td>Breaks objects where they cross boundary edges.</td>
</tr>
<tr>
<td><code>map_dwgtrimobj</code></td>
<td>Trims linear objects inside or outside of a specified boundary.</td>
</tr>
</tbody>
</table>
The map plotting functions begin with `map_plt`.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>map_plblkatts</code></td>
<td>Gets a list of block attributes.</td>
</tr>
<tr>
<td><code>map_plblklist</code></td>
<td>Returns a sorted list of block names that are usable as plot layouts.</td>
</tr>
<tr>
<td><code>map_plblkvps</code></td>
<td>Returns a list of valid viewport layers in layout blocks.</td>
</tr>
<tr>
<td><code>map_pltcleanup</code></td>
<td>Restores settings altered by <code>map_pltinit</code>.</td>
</tr>
<tr>
<td><code>map_pltcurref</code></td>
<td>Selects or creates a plot set.</td>
</tr>
<tr>
<td><code>map_pltcurrefdel</code></td>
<td>Resets a plot set attribute to its default value.</td>
</tr>
<tr>
<td><code>map_pltcurrefget</code></td>
<td>Retrieves the value for a specific attribute for the current plot set.</td>
</tr>
<tr>
<td><code>map_pltcurrefsave</code></td>
<td>Appends the current plot set definition to the plot set list.</td>
</tr>
<tr>
<td><code>map_pltcurrefset</code></td>
<td>Sets the value of an attribute for the current plot set.</td>
</tr>
<tr>
<td><code>map_pltdefdelete</code></td>
<td>Deletes a plot set definition.</td>
</tr>
<tr>
<td><code>map_pltdefget</code></td>
<td>Gets the value of an attribute of the plot set definition.</td>
</tr>
<tr>
<td><code>map_pltdeflist</code></td>
<td>Returns a list of available plot set definitions in the project.</td>
</tr>
<tr>
<td><code>map_pltdefread</code></td>
<td>Reads in a plot set definition from the plot definition dictionary for the project.</td>
</tr>
<tr>
<td><code>map_pltdefsave</code></td>
<td>Writes the current plot set definition to the plot definition dictionary.</td>
</tr>
<tr>
<td><code>map_pltdefvalid</code></td>
<td>Tests the plot set definition for validity.</td>
</tr>
<tr>
<td><code>map_pltdefverify</code></td>
<td>Validates the given plot set to prevent plotting errors.</td>
</tr>
<tr>
<td><code>map_pltdisplay</code></td>
<td>Generates the plot display for the specified boundary.</td>
</tr>
<tr>
<td><code>map_pltexecute</code></td>
<td>Executes a plot, given a specified plot set.</td>
</tr>
<tr>
<td><code>map_pltinit</code></td>
<td>Initializes environment for plotting.</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td><code>map_pltplot</code></td>
<td>Executes the plot script of the current plot set definition.</td>
</tr>
<tr>
<td><code>map_pltrestore</code></td>
<td>Restores display altered by <code>map_pltdisplay</code>.</td>
</tr>
</tbody>
</table>
The map topology functions begin with `map_topo`.

<table>
<thead>
<tr>
<th>Function</th>
<th>Synopsis</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>map_topoaudit</code></td>
<td>Checks the geometrical relationships defined by the topology object data.</td>
</tr>
<tr>
<td><code>map_topoclose</code></td>
<td>Converts all polygons in a topology to closed polylines.</td>
</tr>
<tr>
<td><code>map_topocomplete</code></td>
<td>Completes all objects in a loaded partial topology.</td>
</tr>
<tr>
<td><code>map_topostat</code></td>
<td>Gets the statistics for a topology.</td>
</tr>
</tbody>
</table>
Enumerates the types of layer-status filters.

```cpp
class AcDbBasicFilter {
    // Enumerates the types of layer-status filters.
    enum ELayerStatus {
        eIncludeFrozenLayers = 0x0001,
        eIncludeLockedLayers = 0x0002,
        eIncludeOffLayers = 0x0004
    };
}
```

File

AcDbObjectFilter.h

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>eIncludeFrozenLayers</td>
<td>Filter objects on layers that are frozen.</td>
</tr>
<tr>
<td>eIncludeLockedLayers</td>
<td>Filter objects on layers that are locked.</td>
</tr>
<tr>
<td>eIncludeOffLayers</td>
<td>Filter objects on layers that are turned off.</td>
</tr>
</tbody>
</table>

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

```cpp
virtual ~AcDbBasicFilter();
```

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Constructs an instance of this class with the default value filter values set to "*" to filter all entities.

`AcDbBasicFilter();`

Returns

Returns nothing.

Created with a commercial version of [Doc-O-Matic](http://www.doc-omatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the block(s) on which to filter. The block filter is defined with SetBlocks() (twoforms).

```cpp
const ACHAR* Blocks() const;
```

Returns the blocks on which to filter as any of the following values: A single block name ("Block1", for example) A list of comma-separated block names ("Block1, Block2", for example) A "*" indicating all blocks A "" (empty string) indicating that blocks are ignored during filtering Block names with wildcard characters ("Block?" or "Block-*", for example)

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the feature class(es) on which to filter. The feature-class filter is defined with SetFeatureClasses() (twoforms).

```cpp
const ACHAR* FeatureClasses() const;
```

Returns

Returns the feature classes on which to filter as any of the following values: A single feature-class name ("Feature1", for example)A list of comma-separated feature-class names ("Feature1,Feature2", for example)A "*" indicating all feature classesA "" (empty string) indicating that feature classes are ignored during filteringFeature-class names with wildcard characters ("Feature?" or "Feature-*", for example)

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Filters objects on the current drawing based on the layer, feature-class, and block criteria.

A filtering criterion can take the following values: "*" - (Default) Filters all entities that reside on layers with normal layer status. Unless SetLayerStatusMask() has been called with specific layer mask settings, all objects on layers which are not frozen, off, or locked are considered in the filtering process. "" (empty string) - Indicates that a particular filter is to be ignored during the filtering process. NULL - Indicates that particular filtered entities are to be excluded from the filtered set. A name - The name of a single entity. "Layer1", for example. Multiple names - A list of comma-separated entity names. "Block1,Block2", for example. Wildcards - "?" matches any single character and "*" matches zero of more characters. "Feature?" and "Feature-*", for example.

```cpp
virtual Acad::ErrorStatus FilterObjects(
    AcDbObjectIdArray& outputIds,
    const AcDbObjectIdArray& inputIds
) const;
```

### Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>outputIds</td>
<td>Output array of the IDs of the objects that met the criteria and were filtered. You are responsible for checking the length of this array; a valid filter combination can return no filtered objects.</td>
</tr>
<tr>
<td>inputIds</td>
<td>Input array of the IDs of the candidate objects to inspect for matching filter criteria.</td>
</tr>
</tbody>
</table>

### Returns

Returns Acad::ErrorStatus eOk if successful. Returns Acad::ErrorStatus eInvalidInput if the filter fails.
Retrieves the layer(s) on which to filter. The layer filter is defined with SetLayers() (twoforms). NULL and empty strings (""") are invalid values.

```cpp
const ACHAR* Layers() const;
```

Returns

Returns the layers on which to filter as any of the following values: A single layer name ("Layer1", for example) A list of comma-separated layer names ("Layer1,Layer2", for example) A "*" indicating all layers except off, locked, or frozen ones Layer names with wildcard characters ("layer?" or "lay*", for example)

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links

AcDbBasicFilter Class, AcDbBasicFilter Class
AcDbBasicFilter:: LayerStatusMask Method
AcDbBasicFilter Class | AcDbBasicFilter Class

Determines whether the layer status mask has been set by using
SetLayerStatusMask(). A nonzero return value indicates that objects on layers
which are frozen, locked, or off are to be considered in the filtering process. A
zero return value indicates that no layer mask is set and filtering will be applied
to objects with normal layer status.

int LayerStatusMask() const;

Returns

Returns nonzero (true) if the layer mask is set, or zero (false) if it is not set.

Created with a commercial version of Doc-O-Matic. In order to make this
message disappear you need to register this software. If you have problems
registering this software please contact us at support@toolsfactory.com.
Clears the filter list of blocks.

```cpp
Acad::ErrorStatus ResetBlocks();
```

Returns Acad::ErrorStatus eOk if successful. Returns Acad::ErrorStatus eInvalidInput if parameter validation failed.

Remarks

This function sets the block filter to the default value "*" (filter all blocks).

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Clears the filter list of feature classes.

```cpp
Acad::ErrorStatus ResetFeatureClasses();
```

Returns Acad::ErrorStatus eOk if successful. Returns Acad::ErrorStatus eInvalidInput if parameter validation failed.

Remarks

This function sets the feature-class filter to the default value "*" (filter all feature classes).

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Clears the list of filter layers.

```
Acad::ErrorStatus ResetLayers();
```

Returns

Returns Acad::ErrorStatus eOk if successful. Returns Acad::ErrorStatus eInvalidInput if parameter validation failed.

Remarks

This function sets the layer filter to the default value "*" (filter all layers).

Created with a commercial version of [Doc-O-Matic](https://www.doc-o-matic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the layer-status filter criterion that specifies whether objects on frozen, locked, or off layers are filtered. Use LayerStatusMask() to determine whether the layer-status filter is set.

```cpp
Acad::ErrorStatus SetLayerStatusMask(
    const int statusMask
);
```

**Parameters**

- `statusMask`: Input ELayerStatusmask bits of the criteria to set.

**Returns**

- Returns Acad::ErrorStatus eOk if successful. Returns Acad::ErrorStatus eInvalidInput if the input mask contains invalid ELayerStatus enum values.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

virtual ~AcDbObjectFilter();

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links

AcDbObjectFilter Class, AcDbObjectFilter Class
AcDbObjectFilter:: AcDbObjectFilter Constructor
AcDbObjectFilter Class | AcDbObjectFilter Class

Constructs an instance of this class.

AcDbObjectFilter();

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Filters objects on the current drawing based on the filtering criteria.

```cpp
virtual Acad::ErrorStatus FilterObjects(
    AcDbObjectIdArray& outputIds,
    const AcDbObjectIdArray& inputIds
) const = 0;
```

**Parameters**

- **outputIds**: Output array of the IDs of the objects that met the criteria and were filtered.
- **inputIds**: Input array of the IDs of the candidate objects to inspect for matching filter criteria.

**Returns**

Returns Acad::ErrorStatus eOk if successful; otherwise, returns a different error code.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determine whether this filter is active.

```cpp
template<>
bool IsActive() const;
```

Returns true if this filter is active, or false if it is inactive.

创建使用商业版的`Doc-O-Matic`。为了使该消息消失，您需要注册该软件。如果您在注册该软件时遇到问题，请联系support@toolsfactory.com。
Activates or deactivates this filter. When this filter is active, objects are filtered based on the filtering criteria when FilterObjects() in the derived class is called. When this filter is inactive, no objects are filtered. This setting applies only when the **AcDbObjectFilterGroup** class is used to filter objects.

```cpp
Acad::ErrorStatus SetActive(
    bool bActive
);
```

**Parameters**

**Description**

bActive

Input true to activate this filter, or false to deactivate it.

**Returns**

Returns Acad::ErrorStatus eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

```cpp
virtual ~AcDbObjectFilterGroup();
```

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Constructs an instance of this class.

AcDbObjectFilterGroup();

Returns

Returns nothing.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Adds a filter to the end of the filter list.

```cpp
Acad::ErrorStatus AddObjectFilter(
    const AcDbObjectFilter& objFilter
);
```

**Parameters**

- `objFilter` Input AcDbObjectFilter to add.

**Returns**

Returns Acad::ErrorStatus eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Filters objects on the current drawing based on the filtering criteria of all the filters in the filter list.

```c++
virtual Acad::ErrorStatus FilterObjects(
    AcDbObjectIdArray& outputIds,
    const AcDbObjectIdArray& inputIds
) const;
```

**Parameters**

- `outputIds`:
  Output array of the IDs of the objects that met the criteria and were filtered.

- `inputIds`:
  Input array of the IDs of the candidate objects to inspect for matching filter criteria.

**Returns**

Returns Acad::ErrorStatus eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://toolsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves a filter from the filter list.

```cpp
Acad::ErrorStatus GetObjectFilter(
    const AcDbObjectFilter*& objFilter,
    int index
) const;
```

Parameters

- **objFilter**: Output retrieved AcDbObjectFilter.
- **index**: Input list index of the filter to retrieve.

Returns

Returns Acad::ErrorStatus eOk if successful; otherwise, returns Acad::ErrorStatus eOutOfRange.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Inserts a filter into the filter list.

```cpp
Acad::ErrorStatus InsertObjectFilter(
    int index,
    const AcDbObjectFilter& objFilter
);
```

Parameters

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>index</td>
</tr>
<tr>
<td>objFilter</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::ErrorStatus eOk if successful; otherwise, returns Acad::ErrorStatus eOutOfRange.

Created with a commercial version of [Doc-O-Matic](https://www.doc-o-matic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcDbObjectFilterGroup Class, AcDbObjectFilterGroup Class
AcDbObjectFilterGroup::IsEmpty Method
AcDbObjectFilterGroup Class | AcDbObjectFilterGroup Class

Determines whether the filter list is empty.

```cpp
bool IsEmpty() const;
```

Returns

Returns true if the filter list is empty, or false if it is not empty.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Counts the number of filters in the filter list.

```cpp
int ObjectFilterCount() const;
```

Returns the number of filters in the list.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Clears the filter list.

Acad::ErrorStatus RemoveAllObjectFilter();

Returns

Returns Acad::ErrorStatus eOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Removes a filter from the filter list.

```cpp
    Acad::ErrorStatus RemoveObjectFilter(
        int index
    );
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>index</td>
<td>Input list index of the filter to remove.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::ErrorStatus eOk if successful; otherwise, returns Acad::ErrorStatus eOutOfRange.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Gets the connection object.
Include AdMapOracleConnection.h.

```
AcMapOSEConnection *
AcMapOSEGetConnection();
```

Returns the connection object.

Before you can do anything with Oracle Spatial data, you must get the AcMapOSEConnection object.

```
AcMapOSEConnection *pConnection = AcMapOSEGetConnection();
```

See Managing the connection (overview).
Manages the Oracle database connection and associated reactors.
Instantiated automatically.
Accessed by AcMapOSEGetConnection(), a global function.
Include AdMapOracleConnection.h.

Overviews
Managing the connection  Managing reactors

Code samples
Connecting to an Oracle Database

Functions
~AcMapOSEConnection  IsSchemaValid  RefreshSchema  AddConnectionReactor  RemoveConnectionReactor  AddExportReactor  RemoveExportReactor  AddImportReactor  RemoveImportReactor  Connect  Database  Disconnect  Features  Service  IsConnected  UserName
Base class for custom connection reactors.
Include AdMapOracleReactor.h.

To monitor and react to connection events, subclass a custom reactor from AcMapOSEConnectionReactor, and then add an instance of it to your application using AcMapOSEConnection::AddConnectionReactor().

**Overviews**

*Managing reactors*

**Code samples**

*Connecting to an Oracle Database*

**Functions**

~AcMapOSEConnectionReactor  
AcMapOSEConnectionReactor

BeforeConnect  
BeforeDisconnect  
BeforeSchemaChange  
Connected  
Disconnected  
SchemaChanged
AcMapOSEExport class

Functions  Related Classes  See Also

Manages exporting data to an Oracle database from a project drawing.
Include AdMapOracleExport.h

Overviews
Exporting data

Code samples
Exporting to an Oracle Database (sample)

Functions
~AcMapOSEExport  AcMapOSEExport
ExportObjects
ExportObjectsAll
Init

Enumerations
EExportOptions
Base class for custom export reactors. Include AdMapOracleReactor.h.

To monitor and react to export events, subclass a custom reactor from AcMapOSEExportReactor, and then add an instance of it to your application using AcMapOSEConnection::AddExportReactor().

**Overviews**

Managing reactors

**Code samples**

Connecting to an Oracle Database

**Functions**

~AcMapOSEExportReactor  AcMapOSEExportReactor  
BeforeObjectCached  
BeforeObjectsExported  
ObjectCached  
ObjectRejected  
ObjectsExported
AcMapOSEImport class

**Functions**
- AcMapOSEImport
- Import

**Overviews**

**Querying and importing data**

**Code samples**

Importing from an Oracle Database (sample)

**Functions**

~AcMapOSEImport  AcMapOSEImport
Import
Base class for custom import reactors.
Include AdMapOracleReactor.h.

To monitor and react to import events, subclass a custom reactor from AcMapOSEImportReactor, and then add an instance of it to your application using AcMapOSEConnection::AddImportReactor().

**Overviews**

Managing reactors

**Code samples**

Connecting to an Oracle Database

**Functions**

~AcMapOSEImportReactor  AcMapOSEImportReactor
BeforeRecordImport
RecordImported
RecordRejected
Represents the AutoCAD and Oracle counterparts of a queried object, where the AutoCAD counterpart resides in the project drawing, and the Oracle counterpart resides in the Oracle database. You can initialize the object with the ID of either counterpart: an AutoCAD ID or an Oracle ID.

AcMapOSEObject functions add or remove the object's AutoCAD counterpart from the EditSet, and they provide information about the object, such as

- Whether the AutoCAD counterpart has been erased or modified.
- Whether the AutoCAD counterpart is in the EditSet.
- If the object has an AutoCAD counterpart in another user's EditSet, who the user is.
- The version number of the AutoCAD counterpart, and whether the Oracle and AutoCAD version numbers agree (that is, whether the object is "up to date").
- The object's feature type.

Include AdMapOracleIdentification.h.

**Overviews**

Oracle Spatial and AutoCAD IDs Getting corresponding IDs

**Code samples**

Getting corresponding IDs (sample)

**Functions**

~AcMapOSEObject
AcMapOSEObject
AddToEditSet
FeatureName
GetAcadID
GetOracleID
IsErased
IsInEditSet
IsModified
IsUpToDate
Init
RemoveFromEditSet
Version
WhoHasIt
Represents the drawing database of the project drawing. Lets you filter queried or new drawing objects, and add or remove objects from the EditSet.

Include AdMapOracleProject.h.

**Overviews**

Filtering objects

**Code samples**

Filtering objects (sample)

**Functions**

~AcMapOSEProject AcMapOSEProject
AddToEditSet
FilterNewObjects
FilterQueriedObjects
Init
RemoveFromEditSet

**Enumerations**

EProjectOptions
AcMapOracleQuery class

Specifications the objects to import for the AcMapOracleImport class. Include AcMapOracleQuery.h.

Overviews
Querying and importing data

Code samples
Importing from an Oracle Spatial database (sample)

Functions
~AcMapOracleQuery AcMapOracleQuery
ConvertToSqlString
Init
InitWithCurrent
Load
Save
Gets the connection object.
Include AcMapOracleConnection.h.

```
AcMapOracleConnection *
AcMapOracleGetConnection();
```

Returns the connection object.

Before you can do anything with Oracle Spatial data, you must get the AcMapOracleConnection object.

```
AcMapOracleConnection *pConnection = AcMapOracleGetConnection();
```

See [Managing the connection (overview)](index.html).
AcMapOracleConnection class

Functions  Related Classes  See Also

Manages the Oracle Spatial database connection and associated reactors.
Instantiated automatically.
Accessed by AcMapOracleGetConnection(), a global function.
Include AcMapOracleConnection.h.

Overviews
Managing the connection  Managing reactors

Code samples
Connecting to an Oracle Spatial database

Functions
~AcMapOracleConnection  IsSchemaValid
AddConnectionReactor  RemoveConnectionReactor
AddExportReactor  RemoveExportReactor
AddImportReactor  RemoveImportReactor
Connect  Schema
Database  Service
Disconnect  UserName
IsConnected
Base class for custom connection reactors.
Include AcMapOracleReactor.h.

To monitor and react to connection events, subclass a custom reactor from
AcMapOracleConnectionReactor, and then add an instance of it to your application using
AcMapOracleConnection::AddConnectionReactor().

**Overviews**
- Managing reactors

**Code samples**
- Connecting to an Oracle Spatial database

**Functions**
- ~AcMapOracleConnectionReactor
- AcMapOracleConnectionReactor
- BeforeConnect
- BeforeDisconnect
- BeforeSchemaChange
- Connected
- Disconnected
- SchemaChanged
AcMapOracleExport class

Functions  Related Classes  See Also

Manages exporting data to an Oracle Spatial database from a project drawing.
Include AcMapOracleExport.h

Overviews
Exporting data

Code samples
Exporting to an Oracle Spatial database (sample)

Functions
~AcMapOracleExport  AcMapOracleExport
ExportObjects
ExportObjectsAll

Enumerations
EExportOptions Enum
Base class for custom export reactors.
Include AcMapOracleReactor.h.

To monitor and react to export events, subclass a custom reactor from AcMapOracleExportReactor, and then add an instance of it to your application using AcMapOracleConnection::AddExportReactor().

**Overviews**
Managing reactors

**Code samples**
Connecting to an Oracle Spatial database

**Functions**

~AcMapOracleExportReactor  AcMapOracleExportReactor  
BeforeObjectCached  
BeforeObjectsExported  
ObjectCached  
ObjectRejected  
ObjectsExported
For imported entities, uses the AutoCAD ID (AcDbObjectId) to get the Oracle ID, and vice versa. Include AcMapOracleIdentification.h.

**Overview**

Oracle Spatial and AutoCAD IDs  Getting corresponding IDs

**Code samples**

Getting corresponding IDs (sample)

**Functions**

~AcMapOracleIdentification
AcMapOracleIdentification
GetAcadID
GetOracleID
Init
**AcMapOracleImport class**

Manages importing data from an Oracle Spatial database to a project drawing. Uses the AcMapOracleQuery class to specify the objects to import. Include AcMapOracleImport.h.

**Overviews**

*Querying and importing data*

**Code samples**

*Importing from an Oracle Spatial database (sample)*

**Functions**

~AcMapOracleImport  AcMapOracleImport  
CheckImport  
Import
Base class for custom import reactors.
Include AcMapOracleReactor.h.
To monitor and react to import events, subclass a custom reactor from AcMapOracleImportReactor, and then add an instance of it to your application using AcMapOracleConnection::AddImportReactor().

**Overviews**
- Managing reactors

**Code samples**
- Connecting to an Oracle Spatial database

**Functions**
- ~AcMapOracleImportReactor
- AcMapOracleImportReactor
- BeforeRecordImport
- RecordImported
- RecordRejected
Destroys an instance of this class.

\texttt{virtual \sim AcMapEntityCreationSettings();}

Returns nothing.

Created with a commercial version of \texttt{Doc-O-Matic}. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at \texttt{support@toolsfactory.com}. 
Destroys an instance of this class.

```
virtual ~AcMapFloodParameters();
```

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

```cpp
virtual ~AcMapMarkerStyles();
```

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the error-marker size.

```
AcMap::EErrCode SetMarkerSize(
    int newSize
);
```

**Parameters**
- **newSize**
  - Description: Input marker size.

**Returns**
- Returns `AcMap::EErrCode kOk` if successful; otherwise, returns a different error code.

Created with a commercial version of *Doc-O-Matic*. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the marker shape and color for a duplicate-centroid error.

```cpp
AcMap::EErrCode SetTopoDuplicateCentroidMarker(
    ETopologyMarkType type,
    int nColor
);
```

Parameters

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
</tr>
<tr>
<td>nColor</td>
</tr>
</tbody>
</table>

Returns

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the marker shape and color for an incomplete-area error.

```cpp
AcMap::EErrCode SetTopoIncompleteAreaMarker(
    ETopologyMarkType type,
    int nColor
);
```

**Parameters**

- **type**: Input ETopologyMarkTypeshape type.
- **nColor**: Input color index.

**Returns**

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the marker shape and color for an intersection error.

```cpp
AcMap::EErrCode SetTopoIntersectionMarker(
    ETopologyMarkType type,
    int nColor
);
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>type</code></td>
<td>Input ETopologyMarkTypeshape type.</td>
</tr>
<tr>
<td><code>nColor</code></td>
<td>Input color index.</td>
</tr>
</tbody>
</table>

Returns

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the marker shape and color for a missing-centroid error.

```c++
AcMap::EErrCode SetTopoMissingCentroidMarker(
    ETopologyMarkType type,
    int nColor
);
```

Parameters

- **type**: Input ETopologyMarkTypeshape type.
- **nColor**: Input color index.

Returns

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Destroys an instance of this class.

```cpp
virtual ~AcMapNetAnalysisParameters();
```

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapPointCreationSettings Class, AcMapPointCreationSettings Class
AcMapPointCreationSettings:: ~AcMapPointCreationSettings Destructor
AcMapPointCreationSettings Class | AcMapPointCreationSettings Class

Destroys an instance of this class.

**virtual** ~AcMapPointCreationSettings();

Returns

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Destroys an instance of this class.

`virtual ~AcMapTopoElement();`

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Constructs an instance of this class.

AcMapTopoElement();

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the object identifier of the AutoCAD entity associated with this object.

```cpp
virtual AcMap::EErrCode GetEntity(
    AcDbObjectId& featureID
) const = 0;
```

Parameters

- **featureID**: Output the feature's ID.

Returns

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapTopoElement Class, AcMapTopoElement Class
AcMapTopoElement:: GetID Method
AcMapTopoElement Class | AcMapTopoElement Class

Retrieves the unique identifier of this object in its topology.

\texttt{virtual long GetID() const = \theta;}

Returns

Returns the ID.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determine whether a specified point is within tolerance of this object.

```cpp
virtual bool IsOnThisObject(
    const AcGePoint3d& point,
    double dTolerance
) const = 0;
```

**Parameters**

- **point**: Input point to test.
- **dTolerance**: Input the maximum distance to the point (ignored for polygons). The measurement units depend on the current drawing's unit settings.

**Returns**

Returns true if the point is within tolerance of this object, or false if it is not.

**Remarks**

If this object is a polygon, then the point must be inside or on the boundary, regardless of tolerance.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapTopoElementPtrArray Template, AcMapTopoElementPtrArray Template
AcMapTopoElementPtrArray:: ~AcMapTopoElementPtrArray Destructor
AcMapTopoElementPtrArray Template | AcMapTopoElementPtrArray Template

Destroys the array.

~AcMapTopoElementPtrArray();
Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Releases memory allocated to the array.

```cpp
void Empty();
```

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.toolsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Enumerates the directions of a full edge.

```cpp
enum EDirection {
    eBiDir = 0,
    eForward = 1,
    eBackward = -1
};
```

File

`AcMapTopoFullEdge.h`

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>eBiDir</td>
<td>Bidirectional.</td>
</tr>
<tr>
<td>eForward</td>
<td>Direction from start node to end node.</td>
</tr>
<tr>
<td>eBackward</td>
<td>Direction from end node to start node.</td>
</tr>
</tbody>
</table>

Created with a commercial version of [Doc-O-Matic](https://www.doc-o-matic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

```cpp
virtual ~AcMapTopoFullEdge();
```

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links

class AcMapTopoFullEdge
AcMapTopoFullEdge::GetDirection Method

Retrieves the direction of this edge.

EDirection GetDirection();

Returns

Returns the EDirection edge direction.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the object identifier of the AutoCAD entity associated with this edge.

```cpp
virtual AcMap::EErrCode GetEntity(
    AcDbObjectId& featureID
) const;
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>featureID</td>
<td>Output the feature's ID.</td>
</tr>
</tbody>
</table>

Returns

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the half edge that traverses one way along this full edge.

```
AcMap::EErrCode GetHalfEdge(
    AcMapTopoHalfEdge*& pHalfEdge,
    bool bOnLeft
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pHalfEdge</td>
</tr>
<tr>
<td>bOnLeft</td>
</tr>
</tbody>
</table>

**Returns**

- Returns AcMap::EErrCode kOk if successful.
- Returns AcMap::EErrCode kErrTopTraceLinkNotExist if no edge is found.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Retrieves the identifier of this edge in its topology.

```cpp
virtual long GetID() const;
```

Returns

Returns the ID.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the length of this edge.

```cpp
double GetLength();
```

Returns the edge length.

Remarks

The measurement units depend on the current drawing's unit settings.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links

AcMapTopoFullEdge Class, AcMapTopoFullEdge Class
AcMapTopoFullEdge::GetNextEdge Method
AcMapTopoFullEdge Class | AcMapTopoFullEdge Class

Retrieves the next edge by moving forward or backward along this edge and turning left or right at the next node.

AcMap::EErrCode GetNextEdge(
    AcMapTopoFullEdge*& pOutGoingEdge,
    bool bGoForward,
    bool bGoLeft
);

Parameters

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pOutGoingEdge: Output next edge. This value points to a copy of the full</td>
</tr>
<tr>
<td>edge referenced by the full edge object. The caller is responsible for</td>
</tr>
<tr>
<td>freeing this object.</td>
</tr>
<tr>
<td>bGoForward: Input true to go forward, or false to go backward.</td>
</tr>
<tr>
<td>bGoLeft: Input true to turn left, or false to turn right.</td>
</tr>
</tbody>
</table>

Returns

Returns AcMap::EErrCode kOk if successful. Returns AcMap::EErrCode kErrTopTraceLinkNotExist if no edge is found.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the next node by moving forward or backward along this edge.

```c++
AcMap::EErrCode GetNextNode(
    AcMapTopoNode*& pNextNode,
    bool bGoForward
);
```

**Parameters**

- **pNextNode**
  - Description: Output next AcMapTopoNode node. This value points to a copy of the node referenced by the node object. The caller is responsible for freeing this object.

- **bGoForward**
  - Description: Input true to go forward, or false to go backward.

**Returns**

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the polygon associated with this edge (default level).

```cpp
AcMap::EErrCode GetPolygon(
    AcMapTopoPolygon*& pPolygon,
    bool bOnLeft
);
```

### Parameters

**pPolygon**
- **Output** `AcMapTopoPolygon*polygon`. This value points to a copy of the polygon referenced by the polygon object. The caller is responsible for freeing this object.

**bOnLeft**
- **Input** `true` to choose the left polygon, or `false` to choose the right polygon.

### Returns

Returns `AcMap::EErrCode kOk` if successful. Returns `AcMap::EErrCode kErrTopNotExist` if no polygon is associated with this edge.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Retrieves the resistance of this edge.

```c
double GetResistance(
    bool bGoForward
);
```

### Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bGoForward</td>
<td>Input true to go forward, or false to go backward.</td>
</tr>
</tbody>
</table>

### Returns

Returns the edge resistance.

### Remarks

The measurement units depend on your application. The default resistance is the length of this edge.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the ring associated with this edge (default level).

```
AcMap::EErrCode GetRing(
    AcMapTopoRing*& pRing,
    bool bOnLeft
);
```

### Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pRing</td>
<td>Output AcMapTopoRing. This value points to a copy of the ring referenced by the ring object. The caller is responsible for freeing this object.</td>
</tr>
<tr>
<td>bOnLeft</td>
<td>Input true to choose the left edge, or false to choose the right edge.</td>
</tr>
</tbody>
</table>

### Returns

Returns AcMap::EErrCode kOk if successful. Returns AcMap::EErrCode kErrTopNotExist if no ring is associated with this edge.

Created with a commercial version of [Doc-O-Matic](https://docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the topology that contains this edge.

```cpp
AcMap::EErrCode GetTopology(
    const AcMapTopology*& pTopology
);
```

**Parameters**

- `pTopology`  
  Output `AcMapTopology` reference. Do not delete this reference.

**Returns**

Returns `AcMap::EErrCode kOk` if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapTopoFullEdge Class, AcMapTopoFullEdge Class
AcMapTopoFullEdge::IsOnThisObject Method
AcMapTopoFullEdge Class | AcMapTopoFullEdge Class

Determines whether a specified point is within tolerance of this edge.

```cpp
virtual bool IsOnThisObject(
    const AcGePoint3d& point,
    double dTolerance
) const;
```

Parameters  Description
point      Input point to test.
dTolerance  Input maximum distance to the point.

Returns

Returns true if the point is within tolerance of this edge, or false if it is not.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the direction of this edge.

\[
\text{AcMap::EErrCode SetDirection(}
\text{ }\text{EDirection } \text{eDir}\n\text{);}
\]

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>eDir</td>
<td>Input EDirection direction to set.</td>
</tr>
</tbody>
</table>

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the resistance of this edge.

AcMap::EErrCode SetResistance(
    bool bGoForward,
    double dResistance
);  

Parameters Description
bGoForward Input true to go forward, or false to go backward.
dResistance Input resistance to set.

Returns

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

Remarks

The measurement units depend on your application. The default resistance is the length of this edge.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

**virtual** ~AcMapTopoHalfEdge();

Returns nothing.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Retrieves the full edge that contains this half edge.

\[
\text{AcMap::EErrCode GetFullEdge(}
\text{  const AcMapTopoFullEdge*\& pFullEdge)};
\]

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pFullEdge</td>
<td>Output AcMapTopoFullEdge full edge reference. Do not delete this reference.</td>
</tr>
</tbody>
</table>

Returns

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the next edge by turning left or right at the next node.

```
AcMap::EErrCode GetNextEdge(
    AcMapTopoHalfEdge*& pOutGoingEdge,
    bool bGoLeft
);
```

**Parameters**

- **pOutGoingEdge**
  - Description: Output next edge. This value points to a copy of the half edge referenced by the half edge object. The caller is responsible for freeing this object.

- **bGoLeft**
  - Description: Input true to turn left, or false to turn right.

**Returns**

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapTopoHalfEdge Class, AcMapTopoHalfEdge Class
AcMapTopoHalfEdge:: GetNextNode Method
AcMapTopoHalfEdge Class | AcMapTopoHalfEdge Class

Retrieves the next node by moving forward along this edge.

AcMap::EErrCode GetNextNode(
    AcMapTopoNode*& pNextNode
);

Parameters          Description
pNextNode           Output next AcMapTopoNode. This value points
to a copy of the node referenced by the node object.
The caller is responsible for freeing this object.

Returns

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the polygon associated with this edge (default level).

\texttt{AcMap::EErrCode GetPolygon(}
\texttt{   AcMapTopoPolygon*\& pPolygon}
\texttt{);}  

\textbf{Parameters} \quad \textbf{Description}  
\texttt{pPolygon} \quad \text{Output AcMapTopoPolygon polygon. This value points to a copy of the polygon referenced by the polygon object. The caller is responsible for freeing this object.}  

\textbf{Returns}  
Returns \texttt{AcMap::EErrCode kOk} if successful; otherwise, returns a different error code.

Created with a commercial version of \href{https://www.docomatic.com}{Doc-O-Matic}. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at \url{support@toolsfactory.com}. 

Links  
\url{AcMapTopoHalfEdge Class}, \url{AcMapTopoHalfEdge Class}  
\url{AcMapTopoHalfEdge:: GetPolygon Method}  
\url{AcMapTopoHalfEdge Class | AcMapTopoHalfEdge Class}
Retrieves the previous node by moving backward along this edge.

```
AcMap::EErrCode GetPreviousNode(
    AcMapTopoNode*& pPreviousNode
);
```

**Parameters**

- `pPreviousNode`: Output previous `AcMapTopoNode*` node. This value points to a copy of the node referenced by the node object. The caller is responsible for freeing this object.

**Returns**

Returns `AcMap::EErrCode kOk` if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Retrieve the resistance of this edge.

```cpp
double GetResistance();
```

Returns the edge resistance.

Remarks

The measurement units depend on your application. The default resistance is the length of this edge.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieve the ring associated with this edge (default level).

```cpp
AcMap::EErrCode GetRing(
    AcMapTopoRing*& pRing
);
```

### Parameters

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pRing</td>
</tr>
</tbody>
</table>

- **pRing**: Output AcMapTopoRing. This value points to a copy of the ring referenced by the ring object. The caller is responsible for freeing this object.

### Returns

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the topology that contains this edge.

```cpp
AcMap::EErrCode GetTopology(
    const AcMapTopology*& pTopology
);
```

**Parameters**
- **pTopology**
  - Output AcMapTopology topology reference. Do not delete this reference.

**Returns**
- Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](http). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets the resistance of this edge.

```cpp
AcMap::EErrCode SetResistance(
    double dResistance
);
```

**Parameters**

- **dResistance**: Input resistance to set.

**Returns**

Returns `AcMap::EErrCode kOk` if successful; otherwise, returns a different error code.

**Remarks**

The measurement units depend on your application. The default resistance is the length of this edge.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

```
virtual ~AcMapTopoIterator();
```

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Constructs an instance of this class.

AcMapTopoIterator();

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapTopoIterator Class, AcMapTopoIterator Class
AcMapTopoIterator:: Count Method
AcMapTopoIterator Class | AcMapTopoIterator Class

Counts the number of elements in the iteration.

```cpp
int Count();
```

Returns

Returns the number of elements.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Moves to the first element in the iteration.

`AcMap::EErrCode First();`

Returns

Returns `AcMap::EErrCode kOk` if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Retrieves the description of this topology model.

```cpp
const ACHAR* GetDescription() const;
```

Returns the description of the topology.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapTopoIterator Class, AcMapTopoIterator Class
AcMapTopoIterator:: GetName Method
AcMapTopoIterator Class | AcMapTopoIterator Class

Retrieves the name of this topology model.

\texttt{\textbf{const} ACHAR* GetName() const;}

Returns

Returns the name of the topology.

Created with a commercial version of \texttt{Doc-O-Matic}. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at \texttt{support@toolsfactory.com}. 
Retrieves the topology of the current element in the iteration.

```
AcMap::EErrCode GetTopology(
    AcMapTopology*& pTopology
);
```

**Parameters**

- **pTopology**: Output AcMapTopology. This value points to a copy of the topology referenced by the topology object. The caller is responsible for freeing this object.

**Returns**

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.toolsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Retrieves the type of this topology model.

```cpp
ETopologyType GetType() const;
```

Returns

Returns the ETopolgyType type of the topology model. The topology is `ePoint|eFixed` for a point topology, `eLinear|eFixed` for a network topology, or `ePolygon|eFixed` for a polygon topology.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determines whether the iterator has reached the end of the collection.

```cpp
bool IsDone();
```

Returns

Returns true if the iterator has reached the end of the collection; otherwise, returns false.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapTopoIterator Class, AcMapTopoIterator Class
AcMapTopoIterator:: Next Method
AcMapTopoIterator Class | AcMapTopoIterator Class

Advances to the next element in the iteration.

AcMap::EErrCode Next();
Returns

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Enumerates the result types of an audit.

```cpp
enum EAuditResults {
    eIncorrect = 0x0001,
    eIncomplete = 0x0002
};
```

File

AcMapTopology.h

Parameters Description
---
eIncorrect The topology is incorrect.
eIncomplete The topology is incomplete.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at **support@toolsfactory.com**.
Enumerates the options for creating a topology.

```cpp
enum ECreateOptions {
    eStopAtMultipleCentroid = 0x0001,
    eIgnoreIncompleteArea = 0x0002,
    eHighlightErrors = 0x0004,
    eUsePersistentMarkers = 0x0008,
    eHighlightSliverPolygons = 0x0010
};
```

File

AcMapTopology.h

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>eStopAtMultipleCentroid</td>
<td>Stop at multiple centroid.</td>
</tr>
<tr>
<td>eIgnoreIncompleteArea</td>
<td>Ignore an incomplete area.</td>
</tr>
<tr>
<td>eHighlightErrors</td>
<td>Highlight creation errors.</td>
</tr>
<tr>
<td>eUsePersistentMarkers</td>
<td>Use persistent markers.</td>
</tr>
<tr>
<td>eHighlightSliverPolygons</td>
<td>Highlight silver polygons.</td>
</tr>
</tbody>
</table>

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Enumerates the ways to open a topology model.

```cpp
enum EOpenMode {
    eForRead = 0,
    eForWrite = 1
};
```

### Parameters

<table>
<thead>
<tr>
<th>Description</th>
<th>Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open for read.</td>
<td>eForRead</td>
</tr>
<tr>
<td>Open for write.</td>
<td>eForWrite</td>
</tr>
</tbody>
</table>

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Enumerates the status types of a topology model.

```cpp
enum EStatus {
    eClosed,
    eOpenForRead,
    eOpenForWrite
};
```

File

AcMapTopology.h

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>eClosed</td>
<td>Closed.</td>
</tr>
<tr>
<td>eOpenForRead</td>
<td>Open for read.</td>
</tr>
<tr>
<td>eOpenForWrite</td>
<td>Open for write.</td>
</tr>
</tbody>
</table>

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

```cpp
virtual ~AcMapTopology();
```

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapTopology Class, AcMapTopology Class
AcMapTopology:: AddCurveObject Method
AcMapTopology Class | AcMapTopology Class

Adds a curve (linear object) to this topology model.

AcMap::EErrCode AddCurveObject(
    AcMapTopoFullEdge*& pFullEdge,
    const AcDbObjectId& curveId
);

Parameters  Description
            pFullEdge  Output created AcMapTopoFullEdge edge. This value points to a copy of the full edge referenced by the input curve. The caller is responsible for freeing this object.
            curveId    Input curve to create the edge on.

Returns
Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Adds a point to this topology model.

```cpp
AcMap::EErrCode AddPointObject(
    AcMapTopoNode*& pNode,
    const AcDbObjectId& pointId
);
```

**Parameters**

<table>
<thead>
<tr>
<th>pNode</th>
<th>Output created AcMapTopoNodenode. This value points to a copy of the node referenced by the specified point. The caller is responsible for freeing this object.</th>
</tr>
</thead>
<tbody>
<tr>
<td>pointId</td>
<td>Input ID of the point to create the node on.</td>
</tr>
</tbody>
</table>

**Returns**

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapTopology Class, AcMapTopology Class
AcMapTopology:: AddPolygons Method
AcMapTopology Class | AcMapTopology Class

Adds objects to a polygon topology model.

AcMap::EErrCode AddPolygons(
    const AcDbObjectIdArray& aObjIds
);
Parameters
aObjIds Input array of IDs of edge objects.

Returns
Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

Remarks
The added objects must be linear AutoCAD entities, and must not intersect with each other or with the polygon topology being added to.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the node-creation parameters of this topology model.

```
AcMap::EErrCode SetNodeCreationSettings(
    const AcMapPointCreationSettings& nodeCreationSettings
);
```

**Parameters**

- `nodeCreationSettings` : Input `AcMapPointCreationSettings` node-creation settings.

**Returns**

Returns `AcMap::EErrCode kOk` if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).

Links

- [AcMapTopology Class](#)
- [AcMapTopology Class](#)
- [AcMapTopology:: SetNodeCreationSettings Method](#)
- [AcMapTopology Class](#) | [AcMapTopology Class](#)
Sets the edge-creation parameters of this topology model.

```cpp
AcMap::EErrCode SetEdgeCreationSettings(
    const AcMapEntityCreationSettings& edgeCreationSettings);
```

Parameters

<table>
<thead>
<tr>
<th>edgeCreationSettings</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input AcMapEntityCreationSettings edge-creation settings.</td>
<td></td>
</tr>
</tbody>
</table>

Returns

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the centroid-creation parameters of this topology model.

```cpp
AcMap::EErrCode SetCentroidCreationSettings(
    const AcMapPointCreationSettings& centroidCreationSettings);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>centroidCreationSettings</td>
<td>Input <code>AcMapPointCreationSettings</code> centroid-creation settings.</td>
</tr>
</tbody>
</table>

Returns

Returns `AcMap::EErrCode kOk` if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](http://example.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Closes this topology model.

AcMap::EErrCode Close();

Returns

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Creates a topology. To set topology-creation values, call the three functions `SetNodeCreationSettings()`, `SetEdgeCreationSettings()`, and `SetCentroidCreationSettings()`. If caller does not call these functions, then node creation defaults to false, missing-centroid creation defaults to true, and the node or centroid block name defaults to ACAD_POINT.

```cpp
AcMap::EErrCode Create(
    const AcDb ObjectIdArray& edges,
    const AcDb ObjectIdArray& nodes,
    const AcDb ObjectIdArray& centroids,
    int nType,
    int nCreateOptions = eStopAtMultipleCentroid | eIgnoreIncompleteArea | eUsePersistentMarkers,
    double dTolerance = 0.01
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>edges</td>
<td>Input edge object IDs.</td>
</tr>
<tr>
<td>nodes</td>
<td>Input node object IDs.</td>
</tr>
<tr>
<td>centroids</td>
<td>Input centroid object IDs.</td>
</tr>
<tr>
<td>nType</td>
<td>Input ETopologyType topology type.</td>
</tr>
<tr>
<td>nCreateOptions</td>
<td>Input ECreateOption topology creation options. The default value is eStopAtMultipleCentroid</td>
</tr>
<tr>
<td>dTolerance</td>
<td>Input topology-creation tolerance. The default value is 0.01.</td>
</tr>
</tbody>
</table>

**Returns**

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapTopology Class, AcMapTopology Class
AcMapTopology:: Create Method
AcMapTopology Class | AcMapTopology Class

Creates a topology - this function is not yet implemented in AutoCAD Map; use the other Create() function.

AcMap::EErrCode Create(
  const AcDbObjectIdArray& geometry,
  int nType,
  int nCreateOptions = eStopAtMultipleCentroid | eIgnoreIncompleteArea | eUsePersistentMarkers,
  double dTolerance = 0.01
);

Parameters Description
geometry Input geometry array.
nType Input ETopologyTypetopology type.
nCreateOptions Input ECreateOptionstopology creation options. The default value is eStopAtMultipleCentroid | eIgnoreIncompleteArea | eUsePersistentMarkers.
dTolerance Input topology-creation tolerance. The default value is 0.01.

Returns

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Deletes an edge from this topology model.

```
AcMap::EErrCode DeleteEdge(
    long lEdgeID
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>lEdgeID</td>
</tr>
<tr>
<td>Input ID of the edge to delete.</td>
</tr>
</tbody>
</table>

**Returns**

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

**Remarks**

If the edge is in a polygon, that polygon becomes broken. Before deleting an edge, open the topology should for writing and free all references associated with the edge.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Deletes a node from this topology model.

\[
\text{AcMap::EErrCode DeleteNode(}\quad \text{const AcMapTopoNode}& \text{ node})
\]

**Parameters**

- **node**
  - Input object reference of the AcMapTopoNode to delete.

**Returns**

- Returns AcMap::EErrCode kOk if successful.
- Returns AcMap::EErrCode kErrNotClosed if the node object is open for read/write.
- Returns AcMap::EErrCode kErrTopDeleteDisabled if the node is not a pseudonode for a linear or polygon topology.

**Remarks**

Only current stand-alone nodes can be deleted. For a linear or polygon topology, only a pseudonode can be deleted. Before deleting a node, open a topology for writing. After the deletion succeeds, the object reference passed in the node parameter becomes invalid; the caller is responsible for freeing this object.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Deletes a node from this topology model.

\[
\text{AcMap::EErrCode DeleteNode(}\long\text{lNodeID})
\]

**Parameters**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input ID of the node to delete.</td>
</tr>
</tbody>
</table>

**Returns**

Returns AcMap::EErrCode kOk if successful. Returns AcMap::EErrCode kErrNotClosed if the node object is open for read/write. Returns AcMap::EErrCode kErrTopDeleteDisabled if the node is not a pseudonode for a linear or polygon topology.

**Remarks**

Only current stand-alone nodes can be deleted. For a linear or polygon topology, only a pseudonode can be deleted. Before deleting a node, open the topology for writing and free all references associated with the node. After the deletion succeeds, the node object referenced by lNodeID becomes invalid; the caller is responsible for freeing this object.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Deletes a polygon from the current topology model.

\[
\text{AcMap::EErrCode DeletePolygon(} \\
\text{ \hspace{1cm} const AcMapTopoPolygon& polygon) ;}
\]

**Parameters**

- **polygon**
  
  Input object reference of the AcMapTopoPolygon polygon to delete.

**Returns**

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

**Remarks**

Before deleting the polygon, open the topology for writing. After the deletion succeeds, the object reference passed in the polygon parameter becomes invalid; the caller is responsible for freeing this object.

Created with a commercial version of [Doc-O-Matic](https://example.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapTopology Class, AcMapTopology Class
AcMapTopology:: DeletePolygon Method
AcMapTopology Class | AcMapTopology Class

Deletes a polygon from the current topology model.

AcMap::EErrCode DeletePolygon(
    long lPolygongID
);
Parameters Description
lPolygongID Input ID of the polygon to delete.

Returns

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

Remarks

Before deleting a polygon, open the topology for writing and free all references associated with the polygon. After the deletion succeeds, the object reference with ID lPlygonID becomes invalid; the caller is responsible for freeing this object.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Finds the edge closest to a specified point.

```cpp
AcMap::EErrCode FindEdge(
    AcMapTopoFullEdge*& pEdge,
    double* pDistance,
    const AcGePoint3d& point
);
```

**Parameters**

- **pEdge**
  Output AcMapTopoFullEdge edge object. This value points to a copy of the full edge referenced by the found full edge. The caller is responsible for freeing this object.

- **pDistance**
  Output distance from the point to the nearest edge. The measurement units depend on the current drawing's unit settings.

- **point**
  Input point.

**Returns**

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

**Remarks**

In AutoCAD Map, the specified point must be on the edge; the returned distance always is 0.0.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links

AcMapTopology Class, AcMapTopology Class
AcMapTopology::FindEdge Method
AcMapTopology Class | AcMapTopology Class

Finds the edge with the specified ID.

```cpp
AcMap::EErrCode FindEdge(
    AcMapTopoFullEdge*& pEdge,
    long lId
);
```

Parameters

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Output AcMapTopoFullEdge edge object. This value points to a copy of the full edge referenced by the found full edge. The caller is responsible for freeing this object.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input ID of the edge to find.</td>
</tr>
</tbody>
</table>

Returns

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Finds all polygons that are neighbors of those edges that are based on a specified curve.

```cpp
AcMap::EErrCode FindNeighborPolygons(
    AcMapPolygonPtrArray& apPolygons,
    AcDbObjectId curveId
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>apPolygons</td>
<td>Output array of the neighbor polygons.</td>
</tr>
<tr>
<td>curveId</td>
<td>Input ID of a curve in the topology.</td>
</tr>
</tbody>
</table>

**Returns**

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Finds the node nearest a specified point.

```c
AcMap::EErrCode FindNode(
    AcMapTopoNode* & pNode,
    double* pDistance,
    const AcGePoint3d& point
);
```

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pNode</td>
<td>Output AcMapTopoNode object. This value points to a copy of the node referenced by the found node. The caller is responsible for freeing this object.</td>
</tr>
<tr>
<td>pDistance</td>
<td>Output distance from the point to the nearest node. The measurement units depend on the current drawing's unit settings.</td>
</tr>
<tr>
<td>point</td>
<td>Input point.</td>
</tr>
</tbody>
</table>

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapTopology Class, AcMapTopology Class
AcMapTopology:: FindPolygon Method
AcMapTopology Class | AcMapTopology Class

Finds the smallest polygon containing a specified point.

AcMap::EErrCode FindPolygon(
   AcMapTopoPolygon*& pPolygon,
   const AcGePoint3d& point
);

Parameters Description
pPolygon Output AcMapTopoPolygon polygon object. This value
points to a copy of the polygon referenced by the found
polygon. The caller is responsible for freeing this
object.
point Input point.

Returns
Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error
code.

Created with a commercial version of Doc-O-Matic. In order to make this
message disappear you need to register this software. If you have problems
registering this software please contact us at support@toolsfactory.com.
Finds the object (node, full edge, or polygon) nearest a specified point within a specified tolerance.

\[\text{AcMap}:\text{EErrCode } \text{FindTopologyObject}(\]
\[\text{AcMapTopoElement}^* \& \text{pElement},\]
\[\text{const AcGePoint3d} & \text{point},\]
\[\text{double } \text{dTolerance}\]
\[);\]

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pElement</td>
<td>Output closest AcMapTopoElement object. This value points to a copy of the node, full edge, or polygon referenced by the found object. The caller is responsible for freeing this object.</td>
</tr>
<tr>
<td>point</td>
<td>Input point.</td>
</tr>
<tr>
<td>dTolerance</td>
<td>Input tolerance.</td>
</tr>
</tbody>
</table>

Returns

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

Remarks

If a node is within tolerance, then it is returned. Otherwise, if an edge is within tolerance, then it is returned. Otherwise, the smallest enclosing polygon is returned. If no objects are within tolerance, NULL is returned.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the half edge with the specified object ID from this topology model.

```cpp
AcMap::EErrCode GetBackwardEdge(
    AcMapTopoHalfEdge*& pHalfEdge,
    long lFullEdgeId
);
```

**Parameters**

- **pHalfEdge**
  - Output AcMapTopoHalfEdge half edge object. This value points to a copy of the half edge referenced by the specified half edge object ID. The caller is responsible for freeing this object.

- **lFullEdgeId**
  - Input full edge ID.

**Returns**

- Returns AcMap::EErrCode kOk if successful. Returns AcMap::kErrWrongType if the topology is a polygon. Returns AcMap::kErrBadInput if the full edge ID is invalid.

**Remarks**

The direction of the half edge is from the end node to the start node of the full edge.

Created with a commercial version of [Doc-O-Matic](#). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the description of this topology model.

```cpp
const ACHAR* GetDescription() const;
```

Returns

Returns the description of the topology.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links

AcMapTopology Class, AcMapTopology Class
AcMapTopology:: GetEntityId Method
AcMapTopology Class | AcMapTopology Class

Retrieves the object ID of the AutoCAD entity associated with the specified topology element ID in the current drawing or source drawing.

AcMap::EErrCode GetEntityId(
    AcDbObjectId& aObjId,
    long lElementId
);

Parameters

aObjId          Output object ID of the AutoCAD entity.
lElementId      Input topology element ID.

Returns

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links

AcMapTopology Class, AcMapTopology Class
AcMapTopology:: GetEntityIds Method
AcMapTopology Class | AcMapTopology Class

Retrieves the object IDs of all the AutoCAD entities associated with the topology elements in the current drawing.

AcMap::EErrCode GetEntityIds(
   AcDbObjectIdArray& aObjIds
);

Parameters          Description
aObjIds            Output array of AutoCAD entity object IDs.

Returns

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the half edge with the specified object ID from this topology model.

```cpp
AcMap::EErrCode GetForwardEdge(
    AcMapTopoHalfEdge*& pHalfEdge,
    long lFullEdgeId
);
```

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pHalfEdge</td>
<td>Output AcMapTopoHalfEdge half edge object. This value points to a copy of</td>
</tr>
<tr>
<td></td>
<td>the half edge referenced by the specified half edge object ID. The caller</td>
</tr>
<tr>
<td></td>
<td>is responsible for freeing this object.</td>
</tr>
<tr>
<td>lFullEdgeId</td>
<td>Input full edge ID.</td>
</tr>
</tbody>
</table>

Returns AcMap::EErrCode kOk if successful. Returns AcMap::kErrWrongType if the topology is not a polygon. Returns AcMap::kErrBadInput if the full edge ID is invalid.

Remarks

The direction of the half edge is from the start node to the end node of the full edge.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the full edge with the specified object ID from this topology model.

```cpp
AcMap::EErrCode GetFullEdge(
    AcMapTopoFullEdge*& pFullEdge,
    long lId
);
```

**Parameters**

- **pFullEdge**
  - Description: Output `AcMapTopoFullEdge` full edge object. This value points to a copy of the full edge referenced by the specified full edge object ID. The caller is responsible for freeing this object.

- **lId**
  - Description: Input full edge object ID.

**Returns**

Returns `AcMap::EErrCode kOk` if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapTopology Class, AcMapTopology Class
AcMapTopology:: GetFullEdges Method
AcMapTopology Class | AcMapTopology Class

Retrieves all full edges from this topology model.

AcMap::EErrCode GetFullEdges(
   AcMapFullEdgePtrArray& apFullEdges
);

Parameters
apFullEdges Output array of full edges.

Returns

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the pointer to the error-marker settings of this topology model.

```
AcMapMarkerStyles* GetMarkerStyles();
```

Returns a pointer to the AcMapMarkerStyles error-marker settings.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Retrieves the name of this topology model.

```cpp
const ACHAR* GetName() const;
```

Returns

Returns the name of the topology.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Retrieves the node with the specified object ID from this topology model.

AcMap::EErrCode GetNode(
    AcMapTopoNode*& pNode,
    long lId
);  

Parameters  

<table>
<thead>
<tr>
<th>pNode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Output AcMapTopoNode node object. This value points to a copy of the node referenced by the specified node object ID. The caller is responsible for freeing this object.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>lId</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Input node object ID.</td>
</tr>
</tbody>
</table>

Returns

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves all nodes from this topology model.

\[
\text{AcMap::EErrCode GetNodes(}
\begin{align*}
&\text{AcMapNodePtrArray& apNodes}
\end{align*}
\);
\]

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>apNodes</td>
<td>Output array of nodes.</td>
</tr>
</tbody>
</table>

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

Created with a commercial version of \textbf{Doc-O-Matic}. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at \texttt{support@toolsfactory.com}. 

Links
\texttt{AcMapTopology Class}, \texttt{AcMapTopology Class}
\texttt{AcMapTopology:: GetNodes Method}
\texttt{AcMapTopology Class} | \texttt{AcMapTopology Class}
Retrieves the polygon with the specified object ID from this topology model.

```cpp
AcMap::EErrCode GetPolygon(
    AcMapTopoPolygon*& pPolygon,
    long lId
);
```

**Parameters**

- **pPolygon**
  - Output AcMapTopoPolygon polygon object. This value points to a copy of the polygon referenced by the specified polygon object ID. The caller is responsible for freeing this object.

- **lId**
  - Input polygon object ID.

**Returns**

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves all polygons from this topology model.

```cpp
AcMap::EErrCode GetPolygons(
    AcMapPolygonPtrArray& apPolygons
);
```

- **Parameters**
  - `apPolygons`
    - Description: Output array of polygons.

- **Returns**
  - Returns `AcMap::EErrCode kOk` if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapTopology Class, AcMapTopology Class
AcMapTopology:: GetStatus Method
AcMapTopology Class | AcMapTopology Class

Retrieves the status of this topology model.

EStatus GetStatus() const;
Returns

Returns the EStatus status of the topology model.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the type of this topology model.

```cpp
ETopologyType GetType() const;
```

Returns the ETopologyType type of the topology model.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determines whether this topology model is complete.

```cpp
bool IsComplete() const;
```

Returns

Returns true if the topology model is complete; otherwise, returns false.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Determines whether this topology model can contain only elements of its own type.

```cpp
bool IsFixedType() const;
```

Returns true if the topology model is fixed; otherwise, returns false.

Remarks

As currently implemented, this function always returns true. A non-fixed topology model (that is, one for which this function normally would return false) allows dangling edges in a polygon-only topology.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determine whether this topology model is a network topology.

```cpp
bool IsLinearType() const;
```

Returns

Returns true if the topology model is a network topology; otherwise, returns false.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determines whether the elements in this topology model can be connected if their graphical representations are farther apart than the tolerance value.

```cpp
bool IsLogicalType() const;
```

Returns true if the topology model is a logical topology; otherwise, returns false.

Remarks

As currently implemented, this function always returns false.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Determines whether this topology model is a point topology.

```cpp
bool IsPointType() const;
```

Returns true if the topology model is a point topology; otherwise, returns false.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determined whether this topology model is a polygon topology.

```cpp
bool IsPolygonType() const;
```

Returns true if the topology model is a polygon topology; otherwise, returns false.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Merges neighboring polygons, dissolving the boundaries between them.

\[ \text{AcMap::EErrCode MergePolygons(} \]
\[ \quad \text{AcArray<} \text{long}\text{>} & \quad \text{aPolygonIds} \]
\[ \text{)}; \]

**Parameters**
- **aPolygonIds**: Input array of IDs of the polygons to merge.

**Returns**
Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

**Remarks**
If one of the specified polygons shares no boundary with the others, this function fails. After a merge succeeds, the original polygon reference apPolygons[0] becomes invalid; the caller is responsible for retrieving the new polygon reference with ID apPolygons[0]. The other (now merged) polygons also become invalid and the caller is responsible for freeing them.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Merges neighboring polygons, dissolving the boundaries between them.

```
AcMap::EErrCode MergePolygons(
    AcMapPolygonPtrArray& apPolygons
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>apPolygons</td>
</tr>
<tr>
<td>Input array of polygons to merge.</td>
</tr>
</tbody>
</table>

**Returns**

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

**Remarks**

If one of the specified polygons shares no boundary with the others, this function fails. After a merge succeeds, the aPolygons[0] will reference the newly merged polygon. The other polygon references become invalid and are removed from aPolygons[].

Created with a commercial version of [Doc-O-Matic](https://example.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Merges two neighboring polygons, dissolving the boundaries between them.

\[ \text{AcMap::EErrCode MergePolygons(} \]
\[ \text{const AcMapTopoPolygon& polygon1,} \]
\[ \text{const AcMapTopoPolygon& polygon2} \]
\[ \text{)}; \]

**Parameters**

- **polygon1**
  - Input object reference of the first AcMapTopoPolygon polygon to merge.
- **polygon2**
  - Input object reference of the second AcMapTopoPolygon polygon to merge.

**Returns**

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

**Remarks**

After the merge succeeds, polygon1 will point to the new, merged polygon object and polygon2 becomes invalid; the caller is responsible for freeing the object referenced by polygon2.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Merges two neighboring polygons, dissolving the boundaries between them.

```cpp
AcMap::EErrCode MergePolygons(
    long lPolygonId1,
    long lPolygonId2
);
```

**Parameters**

- **lPolygonId1**: Input ID of the first polygon to merge.
- **lPolygonId2**: Input ID of the second polygon to merge.

**Returns**

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

**Remarks**

After a merge succeeds, the object reference lPolygonId1 becomes invalid; the caller is responsible for retrieving the new polygon reference from the polygon object with ID lPolygonId1. The object reference lPolygonId2 also becomes invalid and the caller is responsible for freeing the object referenced by lPolygonId2.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Moves a node to the new position.

```cpp
AcMap::EErrCode MoveNode(
    const AcMapTopoNode& node,
    const AcGePoint3d& point
);
```

**Parameters**

- **node**: Input object reference of the `AcMapTopoNode` node to move.
- **point**: Input point to move the node to.

**Returns**

Returns `AcMap::EErrCode kOk` if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Moves a node to the new position.

```cpp
AcMap::EErrCode MoveNode(
    long lNodeID,
    const AcGePoint3d& point
);
```

**Parameters**

- `lNodeID`: Input ID of the node to move.
- `point`: Input point to move the node to.

**Returns**

Returns `AcMap::EErrCode kOk` if successful; otherwise, returns a different error code.

**Remarks**

After the move succeeds, the node object referenced by `lNodeID` becomes invalid; the caller must get the new object reference.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Determine whether this topology model is correct.

```cpp
bool NeedsRefresh() const;
```

Returns true if the topology model is incorrect; otherwise, returns false.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Opens the topology model in only the current drawing. After you are finished with a topology model, call Close() to close it.

```
AcMap::EErrCode Open(
    EOpenMode eMode
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Description</th>
<th>eMode</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input EOpenModeopen mode.</td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Opens the topology model in both current and source (attached) drawings. After you are finished with a topology model, call Close() to close it.

```cpp
AcMap::EErrCode Open(
    EOpenMode eMode,
    bool bCreateObjects,
    bool bSelectObjectsForSaveBack,
    int* nAuditResults = NULL
);
```

### Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>eMode</td>
<td>Input EOpenMode open mode.</td>
</tr>
<tr>
<td>bCreateObjects</td>
<td>Input true to create objects from the source drawing; otherwise, false.</td>
</tr>
<tr>
<td>bSelectObjectsForSaveBack</td>
<td>Input true to select topology objects in the save set; otherwise, false.</td>
</tr>
<tr>
<td>nAuditResults</td>
<td>Input pointer that determines topology audit behavior. If NULL (the default value), no audit is performed. If the pointer is valid, the returned value zero indicates that the topology is correct and complete; otherwise, the returned value is a one of (or a combination of) the EAuditResults values.</td>
</tr>
</tbody>
</table>

### Returns

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Updates a topology element.

`AcMap::EErrCode Refresh();`

Returns

Returns `AcMap::EErrCode kOk` if successful. Returns `AcMap::EErrCode kErrTopOpenSourceDwgTopo` if the topology is loaded from the source drawing. Otherwise, returns a different error code.

Remarks

The topology must be loaded from the current drawing.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Highlights the topology model with the specified color.

\[
\text{AcMap::EErrCode ShowGeometry(}
\]
\[
\text{int nColor)};
\]

Parameters

<table>
<thead>
<tr>
<th>Description</th>
<th>Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input color index.</td>
<td>nColor</td>
</tr>
</tbody>
</table>

Returns

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](http://example.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Splits a polygon along a curve.

```
AcMap::EErrCode SplitPolygon(
    AcMapObjectPtrArray& apResults,
    const AcDbObjectId& curveId,
    const AcMapTopoPolygon& polygon
);
```

Parameters

<table>
<thead>
<tr>
<th>Description</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Output new and affected topological objects.</td>
<td>apResults</td>
</tr>
<tr>
<td>Input curve to split at.</td>
<td>curveId</td>
</tr>
<tr>
<td>Input object reference of AcMapTopoPolygon to split.</td>
<td>polygon</td>
</tr>
</tbody>
</table>

Returns

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.doc-o-matic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Splits a polygon along a curve.

```c
AcMap::EErrCode SplitPolygon(
    AcMapObjectPtrArray& apResults,
    const AcDbObjectId& curveId,
    long lPolygonId
);
```

Parameters

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>apResults</td>
</tr>
<tr>
<td>Output new and affected topological objects.</td>
</tr>
<tr>
<td>curveId</td>
</tr>
<tr>
<td>Input curve to split at.</td>
</tr>
<tr>
<td>lPolygonId</td>
</tr>
<tr>
<td>Input ID of polygon to split.</td>
</tr>
</tbody>
</table>

Returns

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

Remarks

After the split succeeds, the polygon object reference with ID lPolygonId becomes invalid and the caller is responsible for retrieving the new object reference.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Finds the least-cost path between start and end points with intermediate points to visit.

```cpp
AcMap::EErrCode TraceBestPath(
    AcMapObjectPtrArray& apObjectsOnPath,
    const AcMapTopoNode& start,
    const AcMapTopoNode& end,
    const AcMapNodePtrArray& apIntermediates,
    const AcMapTraceParameters& params
);
```

**Parameters**

- **apObjectsOnPath**: Output nodes and lines along the path.
- **start**: Input AcMapTopoNode node that starts the path.
- **end**: Input AcMapTopoNode node that ends the path.
- **apIntermediates**: Input array of intermediate nodes to visit.
- **params**: Input `AcMapTraceParameters` trace parameter settings.

**Returns**

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Traces the flood from a specified point.

```c++
AcMap::EErrCode TraceFlood(
    AcMapObjectPtrArray& apObjectsOnPath,
    const AcMapTopoNode& start,
    const AcMapFloodParameters& params
);
```

**Parameters**
- **apObjectsOnPath**
  - Description: Output nodes and lines on the trace flood path.
- **start**
  - Description: Input AcMapTopoNode that starts the trace.
- **params**
  - Description: Input `AcMapFloodParameters` flood trace parameter settings.

**Returns**

Returns `AcMap::EErrCode kOk` if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapTopology Class, AcMapTopology Class
AcMapTopology::TraceLeastCostPath Method
AcMapTopology Class | AcMapTopology Class

Finds the least-cost path between two points.

```
AcMap::EErrCode TraceLeastCostPath(
    AcMapObjectPtrArray& apObjectsOnPath,
    const AcMapTopoNode& start,
    const AcMapTopoNode& end,
    const AcMapTraceParameters& params
);
```

Parameters

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>apObjectsOnPath</td>
</tr>
<tr>
<td>start</td>
</tr>
<tr>
<td>end</td>
</tr>
<tr>
<td>params</td>
</tr>
</tbody>
</table>

Returns

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

```cpp
virtual ~AcMapTopologySource();
```

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapTopologySource Class, AcMapTopologySource Class
AcMapTopologySource:: AcMapTopologySource Constructor
AcMapTopologySource Class | AcMapTopologySource Class

Constructs an instance of this class.

AcMapTopologySource();

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Constructs an instance of this class by using a copy constructor.

```cpp
AcMapTopologySource(
    const AcMapTopologySource& src
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>src</td>
</tr>
<tr>
<td>Input reference to an object of class AcMapTopologySource.</td>
</tr>
</tbody>
</table>

**Returns**

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.doc-o-matic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the topology name.

```cpp
const ACHAR* GetName() const;
```

Returns

Returns the topology name.

Created with a commercial version of [Doc-O-Matic](http://www.docomat.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Retrieves the full path of the topology.

```cpp
const ACHAR* GetPath() const;
```

Returns the full path of the topology.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the source of the topology.

```cpp
const ACHAR* GetSource() const;
```

Returns the source of the topology.

Remarks

The source string will be "DWG" (the current drawing) or "Attached DWG".

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determines whether the topology is loaded.

```cpp
bool IsLoaded() const;
```

Returns true if the topology is loaded, or false if it is not loaded.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Destroys an instance of this class.

```cpp
virtual ~AcMapTopoNode();
```

Returns

Returns nothing.

Created with a commercial version of [Doc-O-Matic](http://www.toolsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Retrieves this node's edges in counter-clockwise order.

```cpp
AcMap::EErrCode GetEdges(
    AcMapHalfEdgePtrArray& apEdges
);
```

**Parameters**

- `apEdges`: Output array of half edges.

**Returns**

Returns `AcMap::EErrCode kOk` if successful. Returns `AcMap::EErrCode kErrTopTraceLinkNotExist` if no edge exists.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links

AcMapTopoNode Class, AcMapTopoNode Class
AcMapTopoNode::GetEntity Method

AcMapTopoNode Class | AcMapTopoNode Class

Retrieves the object identifier of the AutoCAD entity associated with this node.

```cpp
virtual AcMap::EErrCode GetEntity(
     AcDbObjectId& featureID
) const;
```

**Parameters**

- `featureID` Output the feature's ID.

**Returns**

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapTopoNode Class, AcMapTopoNode Class
AcMapTopoNode:: GetID Method
AcMapTopoNode Class | AcMapTopoNode Class

Retrieves the unique identifier of this node in its topology.

virtual long GetID() const;
Returns

Returns the ID.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieve the position of this node.

\[
\text{AcMap::EErrCode GetLocation(}
\quad \text{AcGePoint3d& point}
\)
\]

**Parameters**
- point: Output position.

**Returns**
- Returns `AcMap::EErrCode kOk` if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
AcMapTopoNode Class, AcMapTopoNode Class

AcMapTopoNode:: GetNextEdge Method

AcMapTopoNode Class | AcMapTopoNode Class

Retrieves the next edge given an incoming full edge and a left or right turn direction.

```cpp
AcMap::EErrCode GetNextEdge(
    AcMapTopoFullEdge*& pOutGoingEdge,
    const AcMapTopoFullEdge* pInComingEdge,
    bool bGoLeft
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
</table>
| pOutGoingEdge | Output next AcMapTopoFullEdge.
| pInComingEdge | Input incoming AcMapTopoFullEdge.
| bGoLeft | Input true to turn left, or false to turn right.

Returns

Returns AcMap::EErrCode kOk if successful. Returns AcMap::EErrCode kErrTopTraceLinkNotExist if no edge exists.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the next edge given an incoming half edge and a left or right turn direction.

```cpp
AcMap::EErrCode getNextEdge(
    AcMapTopoHalfEdge*& pOutGoingEdge,
    const AcMapTopoHalfEdge* pInComingEdge,
    bool bGoLeft
);
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pOutGoingEdge</td>
<td>Output next AcMapTopoHalfEdge edge. This value points to a copy of the half edge referenced by the half edge object. The caller is responsible for freeing this object.</td>
</tr>
<tr>
<td>pInComingEdge</td>
<td>Input incoming AcMapTopoHalfEdge half edge.</td>
</tr>
<tr>
<td>bGoLeft</td>
<td>Input true to turn left, or false to turn right.</td>
</tr>
</tbody>
</table>

Returns AcMap::EErrCode kOk if successful. Returns AcMap::EErrCode kErrTopTraceLinkNotExist if no edge exists.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapTopoNode Class, AcMapTopoNode Class
AcMapTopoNode:: GetResistance Method
AcMapTopoNode Class | AcMapTopoNode Class

Retrieves the resistance of this node.

**double** GetResistance();

Returns

Returns the node resistance.

Remarks

The measurement units depend on your application. The default resistance is zero.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Retrieves the topology that contains this node.

\[
\text{AcMap::EErrCode GetTopology(}
\text{const AcMapTopology*\& pTopology)};
\]

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pTopology</td>
<td>Output AcMapTopology topology reference. Do not delete this reference.</td>
</tr>
</tbody>
</table>

Returns

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

Created with a commercial version of \textbf{Doc-O-Matic}. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapTopoNode Class, AcMapTopoNode Class
AcMapTopoNode:: IsMoveable Method
AcMapTopoNode Class | AcMapTopoNode Class

Determines whether this node is moveable - this function is not yet implemented in AutoCAD Map and always returns true.

```cpp
bool IsMoveable();
```

Returns

Returns true if this node is moveable; otherwise, returns false.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Determine whether a specified point is within tolerance of this node.

```cpp
virtual bool IsOnThisObject(
    const AcGePoint3d& point,
    double dTolerance
) const;
```

**Parameters**

- `point`: Input point to test.
- `dTolerance`: Input maximum distance to the point. The measurement units depend on the current drawing's unit settings.

**Returns**

Returns true if the point is within tolerance of this node, or false if it is not.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets whether this node is moveable - this function is not yet implemented in AutoCAD Map.

```
AcMap::EErrCode SetIsMoveable(
    bool bMoveable
);
```

Parameters

- **bMoveable**: Input true to make this node moveable; otherwise, false.

Returns

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapTopoNode Class, AcMapTopoNode Class
AcMapTopoNode:: SetLocation Method
AcMapTopoNode Class | AcMapTopoNode Class

Sets the position of this node - this function is not yet implemented in AutoCAD Map.

AcMap::EErrCode SetLocation(const AcGePoint3d & point);

Parameters Description
point Input new position.

Returns

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the resistance of this node.

```cpp
AcMap::EErrCode SetResistance(
    double dResistance
);
```

Parameters | Description
---|---
dResistance | Input resistance.

Returns

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

Remarks

The measurement units depend on your application. The default resistance is zero.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

```cpp
virtual ~AcMapTopoOverlayData();
```

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Constructs an instance of this class.

AcMapTopoOverlayData();

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Constructs an instance of this class by using a copy constructor.

\begin{verbatim}
AcMapTopoOverlayData(const AcMapTopoOverlayData& src);
\end{verbatim}

Parameters

\begin{tabular}{|l|l|}
\hline
src & Input reference to an object of class AcMapTopoOverlayData. \\
\hline
\end{tabular}

Returns

Returns nothing.

Created with a commercial version of \texttt{Doc-O-Matic}. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the column name of the object data table.

```cpp
const ACHAR* GetColumnName() const;
```

Returns the column name.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the column data type of the object data table.

```cpp
AcMap::EDataType GetColumnType() const;
```

Returns the column data type.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the expression.

\texttt{const ACHAR* GetExpression() const;}

Returns the expression.

Created with a commercial version of \texttt{Doc-O-Matic}. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at \texttt{support@toolsfactory.com}. 
Sets the column data type of the object data table.

```
AcMap::EErrCode SetColumnType(  
    AcMap::EDataType enmColumnType
);
```

Parameters

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>enmColumnType</td>
</tr>
</tbody>
</table>

Returns

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links

AcMapTopoPolygon Class, AcMapTopoPolygon Class
AcMapTopoPolygon:: ~AcMapTopoPolygon Destructor
AcMapTopoPolygon Class | AcMapTopoPolygon Class

Destroys an instance of this class.

virtual ~AcMapTopoPolygon();

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the area of this polygon.

```cpp
double GetArea();
```

Returns

Returns the polygon area.

Remarks

The measurement units depend on the current drawing's unit settings.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the boundary of this polygon, represented by a list of rings.

```cpp
AcMap::EErrCode GetBoundary(
    AcMapRingPtrArray& apBoundingRings
);
```

**Parameters**
- `apBoundingRings` Output array of rings.

**Returns**
Returns `AcMap::EErrCode kOk` if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.doctomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the position of this polygon's centroid.

```cpp
AcMap::EErrCode GetCentroid(
    AcGePoint3d& centroid
) const;
```

**Parameters**

- `centroid`  
  Output centroid position.

**Returns**

Returns `AcMap::EErrCode kOk` if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Retrieves the object identifier of the AutoCAD entity associated with this polygon.

```cpp
virtual AcMap::EErrCode GetEntity(
    AcDbObjectId& featureID
) const;
```

**Parameters**
- `featureID`: Output the feature's ID.

**Returns**
Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Retrieves the hierarchical child polygons, if any, of this polygon - this function is not yet implemented in AutoCAD Map.

\[
\text{AcMap::EErrCode GetHierChildren(} \\
\text{AcMapPolygonPtrArray& apChildren)}; \\
\text{Parameters} \quad \text{Description} \\
\text{apChildren} \quad \text{Output array of child polygons.} \\
\text{Returns} \\
\text{Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.}
\]

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the hierarchical parent polygon, if any, of this polygon - this function is not yet implemented in AutoCAD Map.

```cpp
AcMap::EErrCode GetHierParent(
    AcMapTopoPolygon* & pParent
);
```

**Parameters**

- `pParent` 
  Output parent polygon.

**Returns**

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](http://www.docsomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapTopoPolygon Class, AcMapTopoPolygon Class
AcMapTopoPolygon:: GetID Method
AcMapTopoPolygon Class | AcMapTopoPolygon Class

Retrieves the unique identifier of this polygon in its topology.

virtual long GetID() const;
Returns

Returns the ID.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapTopoPolygon Class, AcMapTopoPolygon Class
AcMapTopoPolygon:: GetParent Method
AcMapTopoPolygon Class | AcMapTopoPolygon Class

Retrieves the multi-polygon parent, if any, of this polygon - this function is not yet implemented in AutoCAD Map.

AcMap::EErrCode GetParent(
   AcMapMultiPolygon*& pParent
);
Parameters Description
pParent Output multi-polygon parent.

Returns

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the perimeter of this polygon.

**double** GetPerimeter();

Returns

Returns the polygon perimeter.

Remarks

The measurement units depend on the current drawing's unit settings.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the topology that contains this polygon.

```cpp
AcMap::EErrCode GetTopology(
    const AcMapTopology*& pTopology
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pTopology</td>
<td>Output AcMapTopology topology reference. Do not delete this reference.</td>
</tr>
</tbody>
</table>

**Returns**

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Determines whether a specified point is within tolerance of this polygon.

```cpp
virtual bool IsOnThisObject(
    const AcGePoint3d& point,
    double dTolerance
) const;
```

**Parameters**

- `point` 
  Input point to test. This point must be inside or on the boundary, regardless of tolerance.

- `dTolerance` 
  (Ignored) Input maximum distance to the point.

**Returns**

Returns true if the point is within tolerance of this polygon, or false if it is not.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Traverses to another polygon by following a curve - this function is not yet implemented in AutoCAD Map.

```
AcMap::EErrCode Traverse(
    AcMapTopoPolygon*& pPolygon,  // Output destination polygon.
    const AcDbObjectId& path      // Input path curve to follow.
);  // Parameters Description

Returns

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

\textbf{virtual} \texttt{~AcMapTopoRing();}

Returns

Returns nothing.

Created with a commercial version of \texttt{Doc-O-Matic}. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at \texttt{support@toolsfactory.com}.
Retrieves the area of this ring.

```cpp
double GetArea();
```

Returns the ring area.

Remarks

The measurement units depend on the current drawing's unit settings.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Retrieves the edges that make up this ring.

```
AcMap::EErrCode GetEdges(
    AcMapHalfEdgePtrArray& apEdges
);
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>apEdges</td>
<td>Output array of edges.</td>
</tr>
</tbody>
</table>

**Returns**

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Retrieves the length of this ring.

```cpp
double GetLength();
```

Returns

Returns the ring length.

Remarks

The measurement units depend on the current drawing's unit settings.

Created with a commercial version of [Doc-O-Matic](https://www.doc-omatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Retrieves the first edge in this ring.

```cpp
AcMap::EErrCode GetStartEdge(
    AcMapTopoHalfEdge*& pStartEdge)
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pStartEdge</td>
<td>Output AcMapTopoHalfEdge edge. This value points to a copy of the half edge object. The caller is responsible for freeing this object.</td>
</tr>
</tbody>
</table>

Returns

Returns AcMap::EErrCode kOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://toolsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapTopoRing Class, AcMapTopoRing Class
AcMapTopoRing:: IsExterior Method
AcMapTopoRing Class | AcMapTopoRing Class

Determines whether this ring is exterior (that is, an outer boundary of its polygon).

**bool** IsExterior();

Returns

Returns true if this ring is exterior; otherwise, returns false.

Created with a commercial version of **Doc-O-Matic**. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at **support@toolsfactory.com**.
Destroys an instance of this class.

```
virtual ~AcMapTraceParameters();
```

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
The functions for cleaning topology objects before building the topology begin with `tpm_clean`.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>tpm_cleanactionlistdel</code></td>
<td>Deletes a cleanup action from the action list.</td>
</tr>
<tr>
<td><code>tpm_cleanactionlistgetat</code></td>
<td>Gets the cleanup action at a given list position.</td>
</tr>
<tr>
<td><code>tpm_cleanactionlistins</code></td>
<td>Inserts a cleanup action in the action list.</td>
</tr>
<tr>
<td><code>tpm_cleanactionlistqty</code></td>
<td>Gets the number of cleanup actions in the action list.</td>
</tr>
<tr>
<td><code>tpm_cleanalloc</code></td>
<td>Allocates the cleanup model.</td>
</tr>
<tr>
<td><code>tpm_cleananchorss</code></td>
<td>Gets a cleanup model's anchored entities.</td>
</tr>
<tr>
<td><code>tpm_cleancancel</code></td>
<td>Cancels the cleanup process without updating the drawing.</td>
</tr>
<tr>
<td><code>tpm_cleancomplete</code></td>
<td>Tests for a next group to be cleaned.</td>
</tr>
<tr>
<td><code>tpm_cleancreatedss</code></td>
<td>Gets created entities following a cleanup process.</td>
</tr>
<tr>
<td><code>tpm_cleanend</code></td>
<td>Completes the cleanup process and updates the drawing.</td>
</tr>
<tr>
<td><code>tpm_cleanerrorcur</code></td>
<td>Sets the next error to clean in the current group.</td>
</tr>
<tr>
<td><code>tpm_cleanerrordraw</code></td>
<td>Creates a temporary marker for the current error.</td>
</tr>
<tr>
<td><code>tpm_cleanerrorfix</code></td>
<td>Fixes the current error.</td>
</tr>
<tr>
<td><code>tpm_cleanerrorget</code></td>
<td>Gets the coordinates of the current error.</td>
</tr>
<tr>
<td><code>tpm_cleanerrormark</code></td>
<td>Creates a persistent marker for the current error.</td>
</tr>
<tr>
<td><code>tpm_cleanerrorset</code></td>
<td>Sets the coordinates of an error fix point.</td>
</tr>
<tr>
<td><code>tpm_cleanfree</code></td>
<td>Frees the cleanup model.</td>
</tr>
<tr>
<td><code>tpm_cleangroupdraw</code></td>
<td>Creates temporary markers for all errors of the current group.</td>
</tr>
<tr>
<td><code>tpm_cleangroupfix</code></td>
<td>Fixes all errors of the current group.</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td><code>tpm_cleangroupmark</code></td>
<td>Creates persistent markers for all errors of the current group.</td>
</tr>
<tr>
<td><code>tpm_cleangroupnext</code></td>
<td>Goes to the next error group.</td>
</tr>
<tr>
<td><code>tpm_cleangroupqty</code></td>
<td>Counts the errors in the current group.</td>
</tr>
<tr>
<td><code>tpm_cleangroupssubtype</code></td>
<td>Determines the subtype of the current group.</td>
</tr>
<tr>
<td><code>tpm_cleangrouptype</code></td>
<td>Determines the type of the current group.</td>
</tr>
<tr>
<td><code>tpm_cleanincludesss</code></td>
<td>Gets a cleanup model's target entities.</td>
</tr>
<tr>
<td><code>tpm_cleaninit</code></td>
<td>Initializes the cleanup model.</td>
</tr>
<tr>
<td><code>tpm_cleaninitanchorset</code></td>
<td>Specifies anchored entities for the cleanup model.</td>
</tr>
<tr>
<td><code>tpm_cleanmodifiedss</code></td>
<td>Gets changed entities following a cleanup process.</td>
</tr>
<tr>
<td><code>tpm_cleanprofileload</code></td>
<td>Loads a drawing cleanup profile.</td>
</tr>
<tr>
<td><code>tpm_cleanprofilessave</code></td>
<td>Saves a drawing cleanup profile.</td>
</tr>
<tr>
<td><code>tpm_cleanstart</code></td>
<td>Starts the cleanup process.</td>
</tr>
<tr>
<td><code>tpm_cleanunchangedss</code></td>
<td>Gets unchanged entities following a cleanup process.</td>
</tr>
</tbody>
</table>
Topology configuration variables include three subsets:

<table>
<thead>
<tr>
<th>Cleanup variables</th>
<th>Properties for cleanup models.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cleanup action variables</td>
<td>Properties for cleanup actions.</td>
</tr>
<tr>
<td>Topology variables</td>
<td>Properties for topologies.</td>
</tr>
</tbody>
</table>

To allocate a set of configuration variables, regardless of which subset you will be using, use `tpm_varalloc`, which returns an `ade_id` for the set that it creates.

To free a set of configuration variables, use `tpm_varfree`.

To get the value of a configuration variable, use `tpm_varget`.

To set the value of a configuration variable, use `tpm_varset`.

To get the properties of a given cleanup action, first allocate a new set of variables, and then use `tpm_cleanactionlistgetat`.

To get the properties of a given topology, first allocate a new set of variables, and then use `tpm_infobuildvar`.

To list all the values of a given set of variables, use `tpm_varlist`. 
Closes a topology.

```c
int
tpm_acclose( 
    ade_id tpm_id);
```

Returns RTNORM or an error code.

```c
tpm_id    The topology ID returned by tpm_acopen.
```

When you close a topology, it remains in memory until you unload it.
tpm_acexist

Access Functions

Checks whether a topology exists.

int
tpm_acexist(
    char *toponame,
    int source,
    int loaded);

Returns RTNORM if the topology exists, or RTERROR if no topology exists with the specified name.

toponame  The topology name for which to check.
source    The source flag, either 1 (check topologies in the current and source drawings) or 0 (check the current drawing only). The default is 0.
loaded    The loaded in memory flag, either 1 (check only topologies in memory) or 0 (check all topologies). The default is 0.
tpm_acload

Access Functions

Loads a topology into memory.

```c
int tpm_acload(
    char *toponame,
    int source);
```

Returns RTNORM or an error code.

- **toponame**   The name of the topology to load.
- **source**    The source flag, either 1 (read from source drawings only) or 0 (read from current drawing only). The default is 0.

Before loading a topology, you must first test for its existence with `tpm_acexist`.

Topology information is stored in the drawing, but the topology is not automatically loaded when you open the drawing. If you want, for example, to query, edit, or overlay a topology, you must load it into memory. Once loaded, the topology remains in memory until you unload it.
Opens a topology.

```c
ade_id
tpm_acopen(
    char *toponame,
    int write_access);
```

Returns the topology ID or ADE_NULLID.

- **toponame**: The name of the topology to open.
- **write_access**: Access status, either 1 (write) or 0 (read only). The default is 0.

The `tpm_acopen` function opens a topology and creates a new `tpm_id` that provides access to it. If your application opens a topology with write access, your user is the only one who can edit this topology. No other user can even open it. If your application opens topology with read access, other users can open it also, but with read access only. To find out the access status of an already-open topology, use `tpm_infostatus`.

If you use `tpm_acopen` to test the status of a topology, always close the topology with `tpm_acclose` to ensure that you do not leave the topology open with multiple IDs pointing to it.

A topology loaded from a source drawing cannot be opened for write access. To find out if a topology was loaded from a current drawing, use `tpm_infocurrent`.

**Important**: You must store the topology ID when it is returned by `tpm_acopen`, and make sure that you do not lose it. If you do not have the ID of an open topology, you have no way to get it, and you cannot close the topology. Your alternatives are to quit AutoCAD Map or start a new drawing.
tpm_acqty

**Access Functions**

Counts the number of topologies.

```c
int tpm_acqty(
    int source,
    int loaded);
```

Returns the number of topologies or an error code.

- **source**: The source flag, either 1 (count topologies in current and source drawings) or 0 (count topologies in current drawing only). The default is 0.
- **loaded**: The loaded in memory flag, either 1 (count only topologies in memory) or 0 (count all topologies). The default is 0.
Unloads a topology from memory.

```c
int tpm_acunload(
    char *toponame);
```

Returns RTNORM or an error code.

**toponame**  The name of the topology to unload.

Before a topology can be unloaded, all topology IDs that reference it must be closed with `tpm_acclose`. 
tpm_acupgradeopen

Access Functions

Changes the access of an open topology from read-only to write.

```c
int tpm_acupgradeopen(
    ade_id tpm_id);
```

Returns RTNORM or an error code.

- **tpm_id**: The topology ID returned by `tpm_acopen`.

If a topology is already open with write access, you cannot change its access with this function. In addition, you cannot change the read-only access of a topology loaded from a source drawing.
tpm_anabuffer

Analyzing Functions

Creates a buffer around a specified topology and stores the results in a new topology.

```c
int tpm_anabuffer(
    ade_id source_id,
    char *offset,
    ade_id var_id,
    char *result_name,
    char *result_desc);
```

Returns RTNORM or an error code.

- **source_id**: The topology ID of the topology you want to buffer.
- **offset**: The buffer offset distance (a standard data extension expression).
- **var_id**: Topology variables ID.
- **result_name**: The name of the resulting buffer topology or NULL if you want to omit this parameter.
- **result_desc**: The description of the resulting buffer topology or NULL if you want to omit this parameter.

The topology variables ID references a set of topology variables.

This function draws one or more buffer perimeters. If the result argument is omitted or NULL, the buffered topologies are AutoCAD objects only. Otherwise the function creates a new topology that is loaded but not open.

The source topology must be loaded and open. The Read/Write access of the source topology does not affect this function.

A buffer is a zone that is drawn around a topology using a specified offset value. You can buffer any of the three topology types, node, network, or polygon. For example, you might specify a buffer on either side of a river to show the extent of a flood plain.
Analyzing Functions

tpm_anadissolve

Combines topology elements that have the same value in the specified field and stores the results in a new topology.

Use this function to create a topology that displays less specific geographic information than the original topology. For example, if a data table attached to a map of city blocks has a field that lists the blocks as commercial or residential, a dissolve can create a new topology in which city block boundaries have been removed.

```
int tpm_anadissolve(
    ade_id source_id,
    char *field,
    ade_id var_id,
    char *result_name,
    char *result_desc,
    char *objTable,
    char *objColumn);
```

Returns RTNORM or an error code.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>source_id</td>
<td>The topology ID of the topology you want to dissolve. You can dissolve polygon or network topologies only.</td>
</tr>
<tr>
<td>field</td>
<td>The field used to dissolve the topology. The field is a standard data extension expression, for example, :Value@Tab1, .Layer. Topology elements that share the same data value in this field will be combined.</td>
</tr>
<tr>
<td>var_id</td>
<td>The topology variables ID.</td>
</tr>
<tr>
<td>result_name</td>
<td>The name of the resulting topology, which will be be loaded but not open. If argument is omitted or NULL, the dissolve results in a collection of AutoCAD objects.</td>
</tr>
<tr>
<td>result_desc</td>
<td>The description of the resulting topology or NULL.</td>
</tr>
<tr>
<td>objTable</td>
<td>The name of the object data table in the resulting topology that will store the shared dissolve field value or NULL.</td>
</tr>
<tr>
<td>objColumn</td>
<td>The name of the field in the object data table that will store the shared dissolve field</td>
</tr>
</tbody>
</table>
value or **NULL**.

The last four parameters are mutually dependent. If you want to omit these parameters, use **NULL** as a placeholder.

The topology variables ID references a set of topology variables.

The function works for polygon or network topologies. If it finds two or more attached objects that have the same value in the specified dissolve field, it dissolves them.

- Dissolving polygons deletes their centroids and shared links and creates a new polygon with a new centroid.
- Dissolving links deletes shared nodes. The links are merged into one polyline.

The function writes the shared dissolve field value to the object data table and field that you specify and attaches the table to the result object. If this table does not exist, the function creates it. If the table you specify is already attached to one of the source objects, a table with default values is attached to the result object.
tpm_anaoverlay

Analyzing Functions

Overlays two topologies.

```c
int tpm_anaoverlay(
    ade_id overlay_id,
    struct resbuf *overlay_data,
    ade_id source_id,
    struct resbuf *source_data,
    int oper,
    ade_id var_id,
    char *obj_table,
    char *obj_tabledesc,
    char *result_name,
    char *result_desc);
```

Returns RTNORM or an error code

- **overlay_id**: Overlay topology ID of topology 1. Must be a polygon topology.
- **overlay_data**: Overlay data expression composed of AutoCAD Map expressions and output object data columns or nil. See the Overlay Data Expressions section below.
- **source_id**: Source topology ID of topology 2. Must be a polygon topology.
- **source_data**: Overlay data expression composed of AutoCAD Map expressions and output object data columns or nil. See the Overlay Data Expressions section below.
- **oper**: Overlay operation. See the Overlay Operations table below.
- **var_id**: ID of topology variables for building the result topology.
- **obj_table**: Name of the new object data table that will contain the final data (created by the function). Cannot be an existing table.
- **obj_tabledesc**: Object data table description.
- **result_name**: Result topology name. Omit if function results produces AutoCAD objects instead of a topology.
result_desc  Result topology description.
overlay_data  Pointer to result buffer of overlay attributes.
source_data  Pointer to result buffer of source attributes.

See the Visual LISP description of `tpm_anaoverlay` for information about the list elements in the `expr1` `resbuf`.

If the `result_name` argument is `NULL`, the result is a collection of AutoCAD objects. Otherwise it is a new topology that is loaded but not open. The last four parameters have the same mutual dependencies in the ADSRX function as they do in the Visual LISP equivalent, except that you cannot omit any of them. You must use `NULL`.

The source topology must be of polygon type for union and paste operations. For all other operations, it can be of any topology type, such as node, network, and polygon.

The topology variables ID references a set of `topology variables`.

**Overlay Data Expressions**

The `overlay_data` and `source_data` arguments are `resbufs` that are constructed as follows:

```c
struct resbuf *list1;
list1 = acutBuildList(
    RTLB
    RTSTR expr1
    RTSTR colname1
    RTSTR coldsc1
    RTSHORT coltype1
    RTLE
    RTLB
    RTSTR expr2
    ...
    RTLE
    ...
    0);
```

**Overlay Data Expression Arguments**

- `expr1`  AutoCAD Map expression (RTSTR).
- `col_name1`  Object data column name (RTSTR).
- `col_desc1`  Object data column description (RTSTR).
Object data column type (RTSHORT): 1 through 4, where 1 = integer, 2 = real, 3 = character, and 4 = point.

### Overlay Operations

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Intersect</td>
</tr>
<tr>
<td>2</td>
<td>Union</td>
</tr>
<tr>
<td>3</td>
<td>Identity</td>
</tr>
<tr>
<td>4</td>
<td>Erase</td>
</tr>
<tr>
<td>5</td>
<td>Clip</td>
</tr>
<tr>
<td>6</td>
<td>Paste</td>
</tr>
</tbody>
</table>

### Result Topology

If the `result_name` argument is omitted or nil, the result is a collection of AutoCAD objects. Otherwise, it is a new topology that is loaded but not open.

You must specify the list of data values to come from each input topology and the specific data for each.

Additionally, you must specify the name of the result object data table in the `objTable` parameter to contain the final data. If you do not specify a result table, no data is attached to the resulting topology elements. The function creates this table. If the result table you specify already exists, the function returns an error and cancels the overlay process.

Data derived by the overlay process is also attached to the result object data table. This data is written for each topology element in the resulting topology. It is written for each polygon if the source is a polygon topology, for each link if it is a network topology, or for each node if it is a node topology. The table always includes the following fields:

### Field Names in Result Object Data Table

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TOPO_ID</td>
<td>Element ID of the new element in the result topology</td>
</tr>
<tr>
<td>T1_ID</td>
<td>Element ID of the parent polygon in the overlay topology</td>
</tr>
<tr>
<td>T1_PERCENTAREA</td>
<td>Area of the new polygon in the result topology compared to the area of the parent polygon in the overlay topology. Written only if both overlay and result are polygon topologies</td>
</tr>
<tr>
<td>T2_ID</td>
<td>Element ID of parent element in the source topology. The parent element can be a node, link, or polygon.</td>
</tr>
<tr>
<td>T2_PERCENTAREA</td>
<td>Area of the new polygon in the resulting topology compared to the area of the parent polygon in the overlay topology</td>
</tr>
</tbody>
</table>
parent polygon in the source topology. Written only if both source and result are polygon topologies

The result table includes these fields along with the fields that you specify in the arguments you supply for the overlay_data and source_data parameters. Each field name is prefixed with T1_ or T2_ to indicate which topology its data comes from.

For example, if you specify FIELD1, FIELD2 and FIELD3 from table SOIL for the first topology and FIELD1, FIELD4 and FIELD5 from table WATER for the second, the result table has the following fields:

T1_SOIL_FIELD1
T1_SOIL_FIELD2
T1_SOIL_FIELD3
T1_WATER_FIELD1
T1_WATER_FIELD4
T1_WATER_FIELD5

The following sample overlays two polygon topologies which are opened and loaded using tpm_acload() and tpm_acopen() respectively. A resbuf is created which contains data expressions associated with object data attached to the source topology. tpm_anaoverlay() is called with all required parameters. The resulting topology and object data table contains information about the percentage of the source polygons consumed by the overlay (buffer topology), the median household income and the population density per square mile among others. The input topologies are closed and then unloaded using tpm_acclose() and tpm_acunload() respectively. The resbufs are then released as required.

```c
char* pszOverlayTopoName = "I-95_Buffer";
int topoWriteAccess = 0;
int returnCode = tpm_acload(pszOverlayTopoName, 0);
ade_id topoForOverlayId = tpm_acopen(pszOverlayTopoName, topoWriteAccess);
struct resbuf* pOverlayDataRb = NULL;
char* pszSourceTopoName = "blkgrp";
returnCode = tpm_acload(pszSourceTopoName, 0);
ade_id sourceTopoId = tpm_acopen(pszSourceTopoName, topoWriteAccess);
struct resbuf* pSourceDataRb = acutBuildList(
    RTLB,
    RTSTR, ":MEDIAN_VAL@RIBLKGRP", // Data Extension expression
    RTSTR, "MEDIAN_VAL",          // Object data column name
    RTSTR, ",",                   // Object data column description
    RTSHORT, 2,                    // Object data column type
    RTLE,
    RTLB,
    RTSTR, ":POP90_SQMI@RIBLKGRP",
    RTSTR, "POP90_SQMI",
);```
int overlayOperation = 1;
ade_id topoConfigVarId = tpm_varalloc();
char* pszODTable = "BufferOverlayBlkGrp";
char* pszODTableDesc = "BufferOverlayBlkGrp-Desc";
char* pszResultTopoName = "BuffrOverBlkGrp";
char* pszResultTopoDesc = "BuffrOverBlkGrp-Desc";
returnCode = tpm_anaoverlay(
    topoForOverlayId,
    pOverlayDataRb,
    sourceTopoId,
    pSourceDataRb,
    overlayOperation,
    topoConfigVarId,
    pszODTable,
    pszODTableDesc,
    pszResultTopoName,
    pszResultTopoDesc
);
if (RTNORM == returnCode){
    acutPrintf("\nThe overlay operation was successfully completed.");
}
else {
    acutPrintf("\nThe overlay operation could not be performed.");
}
tpm_acclose(topoForOverlayId);
tpm_acunload(pszOverlayTopoName);
tpm_acclose(sourceTopoId);
tpm_acunload(pszSourceTopoName);
acutRelRb(pOverlayDataRb);
acutRelRb(pSourceDataRb);
Delete a cleanup action from the action list.

```c
int tpm_cleanactionlistdel
    ade_id clean_var_id,
    long index);
```

Returns **RTNORM** or an error code.

- `clean_var_id`: The cleanup variables ID returned by `tpm_varalloc`.
- `index`: The position in the list of the action to delete.

The `clean_var_id` argument references properties for the cleanup action that you are preparing to initiate (see [Cleanup Variables](#)). These properties include the action list.

The `index` argument is a zero-based position in the action list. A value greater than or equal to the list size or less than 0 returns an error.
tpm_cleanactionlistins

**Cleanup Functions**

Inserts a cleanup action in the action list.

```c
int tpm_cleanactionlistins
    ade_id clean_var_id,
    long index,
    int action,
    ade_id action_var_id);
```

Returns **RTNORM** or an error code.

- **clean_var_id** The cleanup variables ID returned by `tpm_varalloc`.
- **index** Where to insert in the action list.
- **action** The cleanup action to insert, a clean group type. See `tpm_cleangrouptype` for a list of types.
- **action_var_id** The cleanup action variables ID returned by `tpm_varalloc`.

The `clean_var_id` argument references properties for the cleanup operation that you are preparing to initiate (see [Cleanup Variables](#)). These properties include the action list.

The `index` argument is a zero-based position in the action list, or -1 for the last position. A value greater than or equal to the list size or less than -1 is taken as -1.

**Note** When you insert the Simplify Objects action (clean group type 128), it is always listed first, and you cannot insert it more than once.

The `action_var_id` argument references properties affecting the specific cleanup action that you are inserting (see [Cleanup Action Variables](#)). Use `tpm_varset` to set them before calling `tpm_cleanactionlistins`.
tpm_cleanactionlistqty

**Cleanup Functions**

Gets the number of cleanup actions in the action list.

```c
int tpm_cleanactionlistqty
    ade_id clean_var_id,
    long *qty);
```

Returns RTNORM or an error code.

- **clean_var_id**  The cleanup variables ID returned by `tpm_varalloc`.
- **qty**  Outputs the number of cleanup actions.

The `clean_var_id` argument references properties for the cleanup operation that you are preparing to initiate. See [Cleanup Variables](#). These properties include the action list.

The function passes the number of cleanup actions through a parameter (as a `long`) instead of returning it (as a `real`) as the AutoLISP function does.
Allocates the cleanup model.

```c
ade_id
tpm_cleanalloc();
```

Returns the model ID (real) or ADE_NULLID.

To clean the objects before they become the elements of a topology, you must construct a model of these objects and their relationships. You can use this model to discover and repair drawing errors that would prevent topology creation.

The following example allocates a cleanup model, initializes the cleanup model, counts errors by group type and subtype, and frees the cleanup model.

```c
ade_id var_id = ADE_NULLID;  // variable ID
ade_id cln_id = ADE_NULLID;  // clean ID
ads_name ss;                  // selection set
long qty = 0;                  // quantity
int type = 0;                  // clean error group type
int subtype = 0;               // clean error group subtype
int done = 0;
int result = 0;

var_id = tpm_varalloc();
cln_id = tpm_cleanalloc();

if( ! var_id || ! cln_id ) {
    ads_printf("\nMemory allocation failed.");
    tpm_varfree(var_id);
    tpm_cleanfree(cln_id);

    return;
}

// initialize a set of objects, in selection set ss,
// to clean
tpm_cleaninit(cln_id, var_id, ss);
result = tpm_cleanstart(cln_id);

if( result != RTNORM ) {
    ads_printf("nClean startup failed.");
    tpm_varfree(var_id);
    tpm_cleanfree(cln_id);
    return;
}

// Count errors by group type and subtype
while ( ! done ) {
    result = tpm_cleangroupnext(cln_id);
    if ( result == RTNORM ) {
        result = tpm_cleancomplete(cln_id);
        if ( result == RTNORM )
            done = 1;
        else {
            type = tpm_cleangrouptype(cln_id);
            subtype = tpm_cleangroupsubtype(cln_id);
            tpm_cleangroupqty(cln_id, &qty);
            ads_printf("Group type, subtype: %d,%d
Number of errors: %d", type, subtype, qty);
        } // else
    } // if
} // while

tpm_cleanfree(cln_id);
tpm_varfree(var_id);
tpm_cleananchorss

Cleanup Functions

Gets the anchored entities in a cleanup model.

```c
int tpm_cleananchorss
    ade_id clean_var_id,
    ads_name ss);
```

Returns **RTNORM** or an error code.

- `clean_var_id` The cleanup variables ID returned by `tpm_varalloc`.
- `ss` The anchored entities, a selection set.

Call this function after calling `tpm_cleaninit`.
Cancels the cleanup process without updating the drawing.

```c
int tpm_cleancancel(
    ade_id clean_id);
```

Returns RTNORM or an error code.

`clean_id` The cleanup model ID returned by `tpm_cleanalloc`.

The function clears the cleanup model without updating the drawing. Do one of the following:

- Because the model is still allocated, you can call `tpm_cleaninit` using the same cleanup model ID. You can then use `tpm_cleanstart` to start the cleanup process.
- Because `tpm_cleancancel` does not cancel initialized values, you can clear the cleanup model and continue without calling `tpm_cleaninit`. 
tpm_cleancomplete

Cleanup Functions

Tests for a next group to clean.

```c
int tpm_cleancomplete(
    ade_id clean_id);
```

Returns **TRUE** if no more groups need cleaning, or **FALSE**.

clean_id The model ID returned by **tpm_cleanalloc**.

You must call this function after **tpm_cleangroupnext** to determine if another group needs cleaning, because **tpm_cleangroupnext** does not provide this information.

You must usually execute several cleanup loops, because fixing one error sometimes causes others. After you process all the groups in the selection set, call **tpm_cleanstart** to return to the beginning, and then call **tpm_cleancomplete** to test if cleaning is complete. Repeat the loop until cleaning is complete.

For an example that shows how you can use the cleanup functions, see **tpm_cleanalloc**.
tpm_cleancreatedss

**Cleanup Functions**

Gets created entities following a drawing cleanup.

```c
int tpm_cleancreatedss
    ads_id clean_id,
    ads_name ss);
```

Returns RTNORM or an error code.

- **ads_id** Cleanup model ID returned by `tpm_cleanalloc`.
- **ss** Selection set of created entities.

This function gets entities that were created during the cleanup process.

**Note** Call this function after calling `tpm_cleanend`, which concludes the cleanup process. If you call this function earlier, it returns a selection set from the previous cleanup or the empty selection set.
tpm_cleanend

**Cleanup Functions**

Completes the cleanup process and updates the drawing.

```c
int tpm_cleanend(
    ade_id clean_id);
```

Returns **RTNORM** or an error code.

- **clean_id** — The model ID returned by [tpm_cleanalloc](#).

The function resets the cleanup model and updates the drawing. It fixes errors marked with the [tpm_cleanerrorfix](#) function. The model is still allocated. It is possible to call [tpm_cleaninit](#) using the same cleanup model ID.

Each error has some default method, which [tpm_cleanend](#) uses during error fixing. To change this method, use [tpm_cleanerrorset](#).
Sets the next error to clean in the current group.

```c
int tpm_cleanerrorcur(
    ade_id clean_id,
    long index);
```

Returns RTNORM or an error code.

- **clean_id**  The model ID returned by `tpm_cleanalloc`.
- **index**  The index of the error to clean. The index of the first error is 0.

The next error to clean is also called the current error.
tpm_cleanerrordraw

Cleanup Functions

Creates a temporary marker for the current error.

```c
int tpm_cleanerrordraw(
    ade_id clean_id);
```

Returns RTNORM or an error code.

- **clean_id**: The model ID returned by `tpm_cleanalloc`.

Unlike the persistent markers drawn by `tpm_cleanerrormark`, these markers are deleted when the drawing redraws.
tpm_cleanerrorfix

Cleanup Functions

Fixes the current error.

int
tpm_cleanerrorfix(
    ade_id clean_id);

Returns RTNORM or an error code.

    clean_id       The model ID returned by tpm_cleanalloc.

This function marks the current error to fix. It fixes the errors in the cleanup model, but does not fix the drawing. The objects in the drawing are not fixed until you call tpm_cleanend.

Each error has a default method that tpm_cleanend uses during error fixing. You can use tpm_cleanerrorset to change this method. For example, the default method for the dangling node error is to erase the link. If you call tpm_cleanerrorset for this error, the dangling node is moved to a new position, but is not erased.
tpm_cleanerrorget

Cleanup Functions

Gets the coordinates of the current error point.

```c
int tpm_cleanerrorget(
    ade_id clean_id,
    ads_point coords);
```

Returns RTNORM or an error code.

- **clean_id**  The model ID returned by `tpm_cleanalloc`.
- **coords**    The coordinates of the error point.

The error point is a misplaced node. For example, for a line undershoot, the end of the unattached line is the error point.

To specify the error to get, use `tpm_cleanerrorcur`. 
tpm_cleanerrormark

Cleanup Functions

Creates a persistent marker for the current error.

```c
int tpm_cleanerrormark(
    ade_id clean_id);
```

Returns RTNORM or an error code.

`clean_id` The model ID returned by `tpm_cleanalloc`.

Unlike the temporary marker drawn by `tpm_cleanerrordraw`, these markers are AutoCAD objects, and they become part of the drawing until you perform another cleanup. AutoCAD Map automatically erases persistent markers from any previous cleanup process.
tpm_cleanerrorset

Cleanup Functions

Sets the coordinates of an error fix point.

```c
int tpm_cleanerrorset(
    ade_id clean_id,
    ads_point coords);
```

Returns RTNORM or an error code.

- **clean_id**: The model ID returned by `tpm_cleanalloc`.
- **coords**: 2D point.

You can use this function to change the default method used by `tpm_cleanend` during error fixing. For example, the default method for the dangling node error is to erase the link. If you call `tpm_cleanerrorset` for this error, the dangling node is moved to the new position, but is not erased.

You can also use this function to merge a cluster of points to the point you specify.

To tell the clean engine which error to set, use `tpm_cleanerrorcur`. 
Frees the cleanup model.

```c
int tpm_cleanfree(
    ade_id clean_id);
```

Returns RTNORM or an error code.

- **clean_id** The model ID returned by `tpm_cleanalloc`.

For information about preparing the cleanup model, see [Drawing Cleanup](#).

For an example that shows how you can use the cleanup functions, see `tpm_cleanalloc`. 
Creates temporary markers for all errors in the current group.

```c
int tpm_cleangroupdraw(  
    ade_id clean_id);
```

Returns RTNORM or an error code.

- **clean_id** The model ID returned by tpm_cleanalloc.
tpm_cleangroupfix

Cleanup Functions

Fixes all errors in the current group.

```c
int tpm_cleangroupfix(
    ade_id clean_id);
```

Returns RTNORM or an error code.

- **clean_id** The model ID returned by `tpm_cleanalloc`.

This function marks errors in the entire current group for fixing. It fixes the errors in the cleanup model, but does not fix the drawing. The objects in the drawing are not fixed until you call `tpm_cleanend`.

To mark only the current error, use `tpm_cleanerrorfix`. 
tpm_cleangroupmark

Cleanup Functions

Creates persistent markers for all errors in the current group.

int
tpm_cleangroupmark(
    ade_id clean_id);

Returns RTNORM or an error code.

  clean_id The model ID returned by tpm_cleanalloc.
Goes to the next error group.

```c
int tpm_cleangroupnext(ade_id clean_id);
```

Returns RTNORM or an error code

- `clean_id`: The model ID returned by `tpm_cleanalloc`.

The function always returns RTNORM, and fails only when the `clean_id` parameter is invalid. It cannot inform you when there are no more groups to clean. To check for this condition, use `tpm_cleancomplete`.

For an example that shows how you can use the cleanup functions, see `tpm_cleanalloc`. 
Counts the errors in the current group.

```c
int tpm_cleangroupqty(
    ade_id clean_id,
    long *qty);
```

Returns RTNORM or an error code.

- clean_id: The model ID returned by `tpm_cleanalloc`.
- qty: The error count (`long`).

For an example that shows how you can use the cleanup functions, see `tpm_cleanalloc`. 
tpm_cleangroupsubtype

**Cleanup Functions**

Determines the subtype of the current group.

```c
int
tpm_cleangroupsubtype(
    ade_id clean_id);
```

Returns a subtype code or an error code.

- **clean_id** The model ID returned by `tpm_cleanalloc`.

For an example that shows how you can use the cleanup functions, see `tpm_cleanalloc`.

### Subtype Codes

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Degenerate entities (subtype of Erase Short Objects)</td>
</tr>
<tr>
<td>2</td>
<td>Short entities (subtype of Erase Short Objects)</td>
</tr>
<tr>
<td>3</td>
<td>Short segments (subtype of Erase Short Objects)</td>
</tr>
<tr>
<td>4</td>
<td>Vertex is near to segment (subtype of Extend Undershoots)</td>
</tr>
<tr>
<td>5</td>
<td>Vertex is near to vertex (subtype of Extend Undershoots)</td>
</tr>
</tbody>
</table>

Codes 1 through 3 are subtypes of the short type. Codes 4 and 5 are subtypes of the undershoot type. This function is useful for short and undershoot errors only.

**Degenerate Entities** A degenerate polyline has only one vertex. This invalid vertex type sometimes results from the drawing cleanup process, and can be removed by further drawing cleanup.
tpm_cleangrouptype

Cleanup Functions

Determines the type of the current group.

```c
int tpm_cleangrouptype(       
    ade_id clean_id);          
```

Returns a type code or an error code.

```c
clean_id    The model ID returned by `tpm_cleanalloc`.
```

For an example that shows how you can use the cleanup functions, see `tpm_cleanalloc`.

**Type Codes**

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Erase short objects</td>
</tr>
<tr>
<td>2</td>
<td>Break crossing objects</td>
</tr>
<tr>
<td>4</td>
<td>Extend undershoots</td>
</tr>
<tr>
<td>8</td>
<td>Delete duplicates</td>
</tr>
<tr>
<td>16</td>
<td>Snap clustered nodes</td>
</tr>
<tr>
<td>32</td>
<td>Dissolve pseudo nodes</td>
</tr>
<tr>
<td>64</td>
<td>Erase dangling objects</td>
</tr>
<tr>
<td>128</td>
<td>Simplify objects</td>
</tr>
<tr>
<td>256</td>
<td>Zero length objects</td>
</tr>
<tr>
<td>512</td>
<td>Apparent intersections</td>
</tr>
<tr>
<td>1024</td>
<td>Weed polylines</td>
</tr>
</tbody>
</table>
Gets a cleanup model's Include set.

```c
int tpm_cleanincludess
    ade_id clean_var_id,
    ads_name ss);
```

Returns RTNORM or an error code.

- **clean_var_id**    The cleanup variables ID returned by `tpm_varalloc`.
- **ss**              The entities to be cleaned, a selection set.

Call this function after calling `tpm_cleaninit`.
tpm_cleaninit

Cleanup Functions

Initializes the cleanup model.

```c
int tpm_cleaninit(
    ade_id clean_id,
    ade_id var_id,
    ads_name ss);
```

Returns RTNORM or an error code.

- `clean_id` The model ID returned by `tpm_cleanalloc`.
- `var_id` The cleanup variables ID, which references a set of cleanup variables. This is the ID that is returned by `tpm_varalloc`.
- `ss` The selection set or NULL. See note below about INCLUDEOBS_AUTOSELECT.

You can free the selection set after the cleanup model has been initialized.

For more information about the cleanup model, see Drawing Cleanup.

The function reads the cleanup settings and the selected entities into the cleanup model.

If the cleanup variable INCLUDEOBS_AUTOSELECT is set to 1 (select all), all entities in the drawing will be included, regardless of the ss argument, in which case this argument can be NULL instead of a selection set, as the following code sample illustrates.

If the cleanup variable INCLUDEOBS_LAYERS is set to "*" (all layers), all ss entities will be included. If it contains a list of layers, ss entities will be included only if they reside on those layers.

For an example that shows how you can use the cleanup functions, see `tpm_cleanalloc`.

The following sample shows outlines the steps leading up to the use of `tpm_cleaninit()`. Allocate memory for the cleanup model using `tpm_cleanalloc()`, then allocate a set of configuration variables using `tpm_varalloc()`. Define the cleanup variable "INCLUDEOBS_AUTOSELECT", (which must be used with a filtered selection set), then call `tpm_varset()` to set it. Define a selection set of objects to clean and finally, combine the cleanup variables and the selection set of objects using `tpm_cleaninit()`. Release any resbufs as required and free the selection set.
ade_id cleanupModelId = tpm_cleanalloc();
ade_id cleanupVarId = tpm_varalloc();
char* pszConfigVarName = "INCLUDEOBS_AUTOSELECT";
struct resbuf* pIncludeObjsVarValRb = acutBuildList(
    RTSHORT, 0,
    0);
int resultCode = tpm_varset(
    cleanupVarId,
    pszConfigVarName,
    pIncludeObjsVarValRb);
acutRelRb(pIncludeObjsVarValRb);

ads_name ssObjsForCleanup;
acedSSGet("X", NULL, NULL, NULL, ssObjsForCleanup);

resultCode = tpm_cleaninit(
    cleanupModelId,
    cleanupVarId,
    ssObjsForCleanup);
resultCode = acedSSFree(ssObjsForCleanup);
tpm_cleaninitanchorset

Cleanup Functions

Specifies anchored entities for the cleanup model.

```c
int tpm_cleaninitanchorset
    ade_id clean_id,
    ade_id clean_var_id,
    ads_name ssAnchor);
```

Returns RTNORM or an error code.

- **clean_id**: The cleanup model ID returned by `tpm_cleanalloc`.
- **clean_var_id**: The cleanup variables ID returned by `tpm_varalloc`.
- **ssanchor**: The entities to be anchored (a selection set).

Anchored entities remain fixed in position during the cleanup process. You can free the selection set after anchored entities have been specified.

Call `tpm_cleaninitanchorset` before calling `tpm_cleaninit`.

The **clean_var_id** argument references properties for the cleanup operation that you are preparing to initiate (see Cleanup Variables).

If the cleanup variable ANCHOROBSLAYERS is set to "\*" (all layers), all ssAnchor entities will be anchored. If it contains a list of layers, ssAnchor entities will be anchored only if they reside on those layers.
tpm_cleanmodifiedss

Cleanup Functions

Gets changed entities following a drawing cleanup.

```c
int tpm_tpm_cleanmodifiedss
       ads_id clean_id,
       ads_name ss);
```

Returns RTNORM or an error code.

<table>
<thead>
<tr>
<th>ads_id</th>
<th>Cleanup model ID returned by tpm_cleanalloc.</th>
</tr>
</thead>
<tbody>
<tr>
<td>ss</td>
<td>Selection set of changed entities.</td>
</tr>
</tbody>
</table>

This function gets members of the Include set that were changed during the cleanup process.

**Note** Call this function after calling tpm_cleanend, which concludes the cleanup process. If you call this function earlier, it returns a selection set from the previous cleanup or the empty selection set.
tpm_cleanprofileload

Cleanup Functions

Loads a drawing cleanup profile (.dpf file).

```c
int tpm_cleanprofileload
    ade_id clean_var_id,
    char *filename);
```

Returns RTNORM or an error code.

- **clean_var_id**  The cleanup variables ID returned by `tpm_varalloc`.
- **filename**  The full path and .dpf file name.

Loading a profile will reset all properties referenced by the `clean_var_id` argument.
Saves a drawing cleanup profile (.dpf file).

```c
int tpm_cleanprofilesave
    ade_id clean_var_id,
    char *filename);
```

Returns RTNORM or an error code.

- `clean_var_id`: The cleanup variables ID returned by `tpm_varalloc`.
- `filename`: The full path and .dpf file name.
**tpm_cleanstart**

**Cleanup Functions**

Starts the cleanup process.

```c
int tpm_cleanstart(
    ade_id clean_id);
```

Returns RTNORM or an error code.

**clean_id**  The cleanup model ID returned by tpm_cleanalloc.

The function finds the first group type or subtype that contains errors. See tpm_cleangrouptype and tpm_cleangroupsubtype for lists of types.

Cleanup is an iterative process. Depending on cleanup options and processing order, you may need to run the cleanup loop several times to achieve the desired results. After selecting the groups to process with tpm_cleangrouptype and processing all the groups in the selection set, call tpm_cleancomplete to test if cleaning is complete. If complete, call tpm_cleanstart and repeat the cleanup loop.

For an example of the cleanup process, see tpm_cleanalloc.
tpm_cleanunchangedss

Cleanup Functions

Gets unchanged entities following a drawing cleanup.

```c
int tpm_cleanunchangedss
    ads_id clean_id,
    ads_name ss);
```

Returns RTNORM or an error code.

- `ads_id` Cleanup model ID returned by `tpm_cleanalloc`.
- `ss` Selection set of unchanged entities.

This function gets members of the Include set that were not changed during the cleanup process. Note that it does not get members of the Anchor set, which are unchanged by definition.

**Note** Call this function after calling `tpm_cleanend`, which concludes the cleanup process. If you call this function earlier, it returns a selection set from the previous cleanup or the empty selection set.
tpm_editaddelem

Editing Functions

Adds an element to a topology.

```
int tpm_editaddelem(
    ade_id tpm_id,
    int type,
    struct resbuf *res);
```

Returns RTNORM or an error code.

- **tpm_id**: The topology ID.
- **type**: The element's type code: 1, 2, or 3, where 1 = Node, 2 = Link, and 3 = Polygon.
- **res**: The element to add. Depending on the type argument, specify one of the following: If 1, specify a point or the entity name of a point object. If 2, specify the entity name of a line object. If 3, specify a selection set.

The topology must be open with Write access, (1). If you add a node to a link, the link is split.

Set up the resbuf for the res argument like this:

<table>
<thead>
<tr>
<th>Resbuf Element</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>restype</td>
<td>Depends on the function's type argument: 1, 2, or 3. If 1, use RTPOINT, RT3DPOINT (if the topology's CREATE_NODE variable is set), or RTNAME. If 2, use RTNAME. If 3, use RTPICKS.</td>
</tr>
<tr>
<td>resval</td>
<td>Depending on the given restype, a new point, the entity name of an existing line or point entity, or a selection set. If the topology's CREATE_NODE variable is set (see tpm_varset), and you specify a point that does not yet exist, it is created and added to the topology.</td>
</tr>
<tr>
<td>rbnext</td>
<td>NULL (that is, no more data).</td>
</tr>
</tbody>
</table>

For a polygon topology only: If you add a link that closes an open sequence of links or is connected to the
topology at both ends of the link, \texttt{tpm\_editaddelem} creates a new polygon.

The following sample adds a new element to an existing network topology. A \texttt{resbuf} is created containing the information required to draw a new pline. The pline is drawn using \texttt{acedCmd()}. The new pline's entity id is then obtained using \texttt{acdbEntLast()}. The topology that will receive the new link is loaded and opened using \texttt{tpm\_acload()} and \texttt{tpm\_acopen()} respectively. A \texttt{resbuf} containing the new pline entity is constructed, then \texttt{tpm\_editaddelem()} is called with all required parameters. Appropriate status messages are displayed based on the return code. Topologies are closed and unloaded from memory and the \texttt{resbufs} are released as required.

```c
struct resbuf* pNewPlineRb = acutBuildList(
    RTSTR, "_pline",
    RTSTR, "54.0816,50.8243",
    RTSTR, "68,53",
    RTSTR, "",
    0);
int returnCode = acedCmd(pNewPlineRb);
ads_name newPline;
returnCode = acdbEntLast(newPline);
char* pszTopoName = "NetTopo";
int topoWriteAccess = 1;
returnCode = tpm_acload(pszTopoName, 0);
ade_id topoId = tpm_acopen(pszTopoName, topoWriteAccess);
struct resbuf* pNewElementRb = acutBuildList(RTENAME, newPline, 0);
int elementType = 2;
returnCode = tpm_editaddelem(topoId, elementType, pNewElementRb);
if (RTNORM == returnCode){
    acutPrintf(
        "\nThe new element was successfully added.");
} else {
    acutPrintf(
        "\nThe new element was not added.");
}
tpm_acclose(topoId);
tpm_acunload(pszTopoName);
acutRelRb(pNewPlineRb);
acutRelRb(pNewElementRb);
```
tpm_editdelelem

Editing Functions

Deletes an element from a topology.

```c
int tpm_editdelelem(
    ade_id tpm_id,
    ade_id elem_id,
    int delobj);
```

Returns RTNORM or an error code.

tpm_id The topology ID.

elem_id The ID of the desired element.

delobj Delete status, either 0 (deletes the object from the topology only) or non-zero (deletes the object from the drawing also).

The following example deletes from memory and erases all the links of a network topology and counts the number of successful deletions.

```c
void Delete_links()
{
    int indx = 0; // Element index
    int dindx = 0; // Deleted element count
    long qty = 0;
    ade_id tpm_id = ADE_NULLID;
    ade_id elm_id = ADE_NULLID;
    int result = 0;

    // load or build a network topology here
    // and also use tpm_acopen to assign a tpm_id for the topology

    // get the number of links
    tpm_elemqty (tpm_id, 2, &qty);
```
// delete all of the topology links

for ( ; indx < qty ; ) {
    // get the links ID
    elm_id = tpm_elemid(tpm_id, 2, indx);
    if ( elm_id == ADE_NULLID ) {
        indx++;
        ads_printf("\nUnable to obtain an element ID.");
    }
    else {
        // delete element and erase object
        result = tpm_editdelelem(tpm_id, elm_id, 1);

        if ( result == RTNORM )
            dindx++;
        else
            indx++;
    }
}

} // delete_links

Notes and Warnings

- This function does not erase corresponding entities in the drawing unless the delobj argument is set to something other than 0.
- Deleting an element can cause other deletions.
  - If you delete a node, you delete any link or polygon that contains it.
  - If you delete a link, you delete only the nodes belonging to that link. If the link belongs to one polygon only, you delete the polygon. If the link is shared by two polygons, you merge the polygons.
  - If you delete a polygon, you delete any node or link belonging to that polygon only.
tpm_editmodelem

Editing Functions

Modifies a topology element.

int tpm_editmodelem(
    ade_id tpm_id,
    ade_id elem_id,
    struct resbuf *new_val);

Returns RTNORM or an error code.

tpm_id The topology ID
elem_id The ID of the desired element.
new_val A list consisting of a code for the property to modify and a new value for the property. See Properties and Values below.

The following code constructs a resbuf for a new_val point argument:

    ads_point newpoint;
    newpoint[X] = 1.0;
    newpoint[Y] = 1.0;
    newpoint[Z] = 0.0;

    struct resbuf* new_val = ads_buildlist (    
        RTLB,
        RTSHORT, 10,
        RTPOINT, newpoint,
        RTLE,
        0);

Note The dotted pair is not supported for point data.

The following examples of build resbuf data structures for the new_val argument.

    // For a point value, given a point, p
rb = acutBuildList(
    RTLB, RTSHORT, 10, RTPOINT, p, RTLE, 0);
// For an integer value
rb = acutBuildList(
    RTLB, RTSHORT, code, RTSHORT, value, RTDOTE, 0);
// For a real value
rb = acutBuildList(
    RTLB, RTSHORT, code, RTREAL, value, RTDOTE, 0);

You must release the resbuf.

**Properties and Values**

<table>
<thead>
<tr>
<th>(10 . point)</th>
<th>New coordinates of node or centroid (RTPOINT)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(40 . f_res)</td>
<td>Resistance of node (RTREAL), or forward resistance of link</td>
</tr>
<tr>
<td>(41 . r_res)</td>
<td>Reverse resistance of link (RTREAL)</td>
</tr>
<tr>
<td>(70 . dir)</td>
<td>Link direction (RTSHORT):</td>
</tr>
<tr>
<td></td>
<td>-1 Reverse</td>
</tr>
<tr>
<td></td>
<td>0 Bidirectional</td>
</tr>
<tr>
<td></td>
<td>1 Forward</td>
</tr>
</tbody>
</table>

The following example modifies the direction variable on a link element in the "MyNetworkTopo" topology. The topology is first loaded into memory using `tpm_acload()`, then opened for write using `tpm_acopen()`. A resbuf is constructed containing the property/value pair associated with link direction. After calling `tpm_editmodelem()` with all required parameters, the `returnCode` is evaluated and an appropriate message is displayed. The resbuf is then released as required.

```c
char* pszTopoName = "NetTopo";
int topoWriteAccess = 1;
int returnCode = tpm_acload(pszTopoName, NULL);
ade_id topoId = tpm_acopen(pszTopoName, topoWriteAccess);
int elementType = 2;
long elementIndex = 1;
ade_id elementID = tpm_elemid(topoId, elementType, elementIndex);
struct resbuf* pModElementRb = acutBuildList(
    RTLB,
    RTSHORT, 70,
    RTSHORT, -1,
    RTDOTE,
    0);
returnCode = tpm_editmodelem(topoId, elementID, pModElementRb);
```
if (RTNORM == returnCode){
    acutPrintf(
        "\nThe element modification was successful.\n");
}
else {
    acutPrintf(
        "\nThe element modification was not successful.\n");
}
acutRelRb(pModElementRb);
tpm_acclose(topoId);
tpm_acunload(pszTopoName);
tpm_editupdelem

**Editing Functions**

Updates a topology element.

```c
int tpm_editupdelem(
    ade_id tpm_id,
    ade_id elem_id);
```

Returns **RTNORM** if the object was successfully updated or an error code if it was not.

- **tpm_id** The topology ID.
- **elem_id** The ID of the desired element.

This function updates a topology element so that it reflects the current state of the corresponding entity in AutoCAD. This function is normally used in conjunction with AutoCAD Notification so that changes made in AutoCAD can be reflected in the topology model.
tpm_elemadj

Element Information Functions

Compiles a list of adjacent elements for the specified element.

```
struct resbuf
*tpm_elemadj(
    ade_id tpm_id,
    ade_id elem_id,
    int adj_type);
```

Returns a list of element IDs or NULL.

- **tpm_id**: The topology ID.
- **elem_id**: An element ID, received from `tpm_elemid`.
- **adj_type**: The type of adjacent elements to be compiled into the list. Values can be: 1 (node), 2 (link), or 3 (polygon).

The following sample populates a `resbuf` with adjacent elements, (links) on a network topology using `tpm_elemadj()`. If the operation is successful the adjacent id(s) are displayed, and the `resbuf` is released as required. Note, `tpm_acopen()` was used to obtain the loaded topology id and `tpm_elemid()` was used to obtain a specific element id within the loaded topology.

```c
char* pszTopoName = "NetTopo";
int topoWriteAccess = 0;
ade_id topoId = tpm_acopen(pszTopoName, topoWriteAccess);
int adjElementType = 2;
long elementIndex = 1;
ade_id adjElementID = tpm_elemid(topoId, adjElementType, elementIndex);
struct resbuf* pAdjElementIdsRb = tpm_elemadj(topoId, adjElementID, adjElementType);
if (NULL != pAdjElementIdsRb){
    struct resbuf* rb = pAdjElementIdsRb;
    acutPrintf("The specified element has an ID of %.0lf. The following adjacent ID(s) were detected: ", adjElementID);
    while(NULL != rb) {
        ...
    }
}
```
acutPrintf("\n't %.0lf"
, rb->resval.rreal);
rb = rb->rbnext;
}
}
else {
    acutPrintf("\nNo adjacent elements were detected.");
}
acutRelRb(pAdjElementIdsRb);
tpm_acclose(topoId);
tpm_elemfind

Element Information Functions

Finds an element within a topology.

ade_id
tpm_elemfind(
    ade_id tpm_id,
    int type,
    struct resbuf *pattern);

Returns an element ID or ADE_NULLID.

tpm_id      The topology ID.

  type      The type of element to find. Values can be: 1 (node), 2 (link), or 3 (polygon).

  pattern    The name of the point or entity. If pattern is a point, in which case type must be 1, the function returns the nearest point or link, or the enclosing polygon. If pattern is an entity name, in which case type can have any value, the function returns the corresponding object.

The following sample finds a specific element in an existing network topology and uses the tpm_editaddelem sample as a baseline.

A resbuf is created containing the information required to draw a new pline. The pline is drawn using acedCmd(). The new pline's entity id is then obtained using acdbEntLast(). The topology that will recieve the new link is loaded and opened using tpm_acload() and tpm_acopen() respectively. A resbuf containing the new pline entity is constructed, then tpm_editaddelem() is called with all required parameters. Appropriate status messages are displayed based on the return code.

To find the new element; A resbuf is created containing the pattern parameter information, (entity name). Tpm_elemfind() is then called with all required parameters. The return value is validated against ADE_NULLID and appropriate status messages are displayed. Topologies are closed and unloaded from memory and the resbufs are released as required.

```c
struct resbuf* pNewPlineRb = acutBuildList(
    RTSTR, "_pline",
    RTSTR, "54.0816,50.8243",
    RTSTR, "68,53",
```
int returnCode = acedCmd(pNewPlineRb);
ads_name newPline;
returnCode = acdbEntLast(newPline);
char* pszTopoName = "NetTopo";
int topoWriteAccess = 1;
returnCode = tpm_acload(pszTopoName, 0);
ade_id topoId = tpm_acopen(pszTopoName, topoWriteAccess);
struct resbuf* pNewElementRb = acutBuildList(RTENAME, newPline, 0);
int elementType = 2;
returnCode = tpm_editaddelem(topoId, elementType, pNewElementRb);
if (RTNORM == returnCode){
    acutPrintf("\nThe new element was successfully added.");
} else {
    acutPrintf("\nThe new element was not added.");
}

// Find the new element.
struct resbuf* pSeekPatternRb = acutBuildList(RTENAME, newPline, 0);
ade_id elementId = tpm_elemfind(topoId, elementType, pSeekPatternRb);
if (ADE_NULLID != elementId){
    acutPrintf("\nThe new element has a id of: %.0lf", elementId);
} else {
    acutPrintf("\nThe specified element was not found.");
}

tpm_acclose(topoId);
tpm_acunload(pszTopoName);
acutRelRb(pNewPlineRb);
acutRelRb(pNewElementRb);
acutRelRb(pSeekPatternRb);
tpm_elemget

Element Information Functions

Lists information about an element in a topology.

struct resbuf
*tpm_elemget(
    ade_id tpm_id,
    ade_id elem_id);

Returns a resbuf list of value pairs or NULL.

tpm_id    The topology ID.

elem_id   An element ID returned by tpm_elemid.

For each value pair in the list that is returned, the first value is an integer code for the information type, and the second value is the information. The list format depends on the element type: node, link, or polygon.

The following examples illustrate possible contents of resbuf data structures returned by tpm_elemget.

If the referenced element is a node, the contents of the resbuf could be as follows.

<table>
<thead>
<tr>
<th>Restype</th>
<th>Resval</th>
</tr>
</thead>
<tbody>
<tr>
<td>RTLB</td>
<td></td>
</tr>
<tr>
<td>RTSHORT</td>
<td>0</td>
</tr>
<tr>
<td>RTREAL</td>
<td>10.000000</td>
</tr>
<tr>
<td>RTDOTE</td>
<td></td>
</tr>
<tr>
<td>RTLB</td>
<td></td>
</tr>
<tr>
<td>RTSHORT</td>
<td>-2</td>
</tr>
<tr>
<td>RTENNAME</td>
<td>2546033006, 3172571566</td>
</tr>
<tr>
<td>RTDOTE</td>
<td></td>
</tr>
</tbody>
</table>
If the referenced element is a link, the contents of the `resbuf` could be as follows.

<table>
<thead>
<tr>
<th>Restype</th>
<th>Resval</th>
</tr>
</thead>
<tbody>
<tr>
<td>RTLBM</td>
<td></td>
</tr>
<tr>
<td>RTSHORT</td>
<td>-1</td>
</tr>
<tr>
<td>RTSHORT</td>
<td>1</td>
</tr>
<tr>
<td>RTDOTE</td>
<td></td>
</tr>
<tr>
<td>RT3DPOINT</td>
<td>10.000000, 4.000000, 4.000000</td>
</tr>
<tr>
<td>RTLBM</td>
<td></td>
</tr>
<tr>
<td>RTSHORT</td>
<td>40</td>
</tr>
<tr>
<td>RTREAL</td>
<td>0.000000</td>
</tr>
<tr>
<td>RTDOTE</td>
<td></td>
</tr>
<tr>
<td>RTLBM</td>
<td></td>
</tr>
<tr>
<td>RTSHORT</td>
<td>-2</td>
</tr>
<tr>
<td>RTENAME</td>
<td>2546045206, 3172507766</td>
</tr>
<tr>
<td>RTDOTE</td>
<td></td>
</tr>
<tr>
<td>RTLBM</td>
<td></td>
</tr>
<tr>
<td>RTSHORT</td>
<td>-1</td>
</tr>
<tr>
<td>RTSHORT</td>
<td>2</td>
</tr>
<tr>
<td>RTDOTE</td>
<td></td>
</tr>
<tr>
<td>RTLBM</td>
<td></td>
</tr>
<tr>
<td>RTSHORT</td>
<td>1</td>
</tr>
<tr>
<td>RTREAL</td>
<td>4.000000</td>
</tr>
<tr>
<td>---------</td>
<td>----------</td>
</tr>
<tr>
<td>RTDOTE</td>
<td></td>
</tr>
<tr>
<td>RTLB</td>
<td></td>
</tr>
<tr>
<td>RTSHORT</td>
<td>2</td>
</tr>
<tr>
<td>RTREAL</td>
<td>8.000000</td>
</tr>
<tr>
<td>RTDOTE</td>
<td></td>
</tr>
<tr>
<td>RTLB</td>
<td></td>
</tr>
<tr>
<td>RTSHORT</td>
<td>3</td>
</tr>
<tr>
<td>RTREAL</td>
<td>19.000000</td>
</tr>
<tr>
<td>RTDOTE</td>
<td></td>
</tr>
<tr>
<td>RTLB</td>
<td></td>
</tr>
<tr>
<td>RTSHORT</td>
<td>4</td>
</tr>
<tr>
<td>RTREAL</td>
<td>20.000000</td>
</tr>
<tr>
<td>RTDOTE</td>
<td></td>
</tr>
<tr>
<td>RTLB</td>
<td></td>
</tr>
<tr>
<td>RTSHORT</td>
<td>40</td>
</tr>
<tr>
<td>RTREAL</td>
<td>2.320675</td>
</tr>
<tr>
<td>RTDOTE</td>
<td></td>
</tr>
<tr>
<td>RTLB</td>
<td></td>
</tr>
<tr>
<td>RTSHORT</td>
<td>41</td>
</tr>
<tr>
<td>RTREAL</td>
<td>2.320675</td>
</tr>
<tr>
<td>RTDOTE</td>
<td></td>
</tr>
<tr>
<td>RTLB</td>
<td></td>
</tr>
<tr>
<td>RTSHORT</td>
<td>70</td>
</tr>
<tr>
<td>RTSHORT</td>
<td>0</td>
</tr>
<tr>
<td>RTDOTE</td>
<td></td>
</tr>
</tbody>
</table>
If the referenced element is a polygon, the contents of the `resbuf` could be as follows.

<table>
<thead>
<tr>
<th>Restype</th>
<th>Resval</th>
</tr>
</thead>
<tbody>
<tr>
<td>RTLB</td>
<td></td>
</tr>
<tr>
<td>RTSHORT</td>
<td>0</td>
</tr>
<tr>
<td>RTREAL</td>
<td>20.000000</td>
</tr>
<tr>
<td>RTDOTE</td>
<td></td>
</tr>
<tr>
<td>RTLB</td>
<td></td>
</tr>
<tr>
<td>RTSHORT</td>
<td>-2</td>
</tr>
<tr>
<td>RTNAME</td>
<td>2546047206, 3172505766</td>
</tr>
<tr>
<td>RTDOTE</td>
<td></td>
</tr>
<tr>
<td>RTLB</td>
<td></td>
</tr>
<tr>
<td>RTSHORT</td>
<td>-1</td>
</tr>
<tr>
<td>RTSHORT</td>
<td>3</td>
</tr>
<tr>
<td>RTDOTE</td>
<td></td>
</tr>
<tr>
<td>RT3DPOINT</td>
<td>10.000000, 3.319400, 4.589200</td>
</tr>
<tr>
<td>RTLB</td>
<td></td>
</tr>
<tr>
<td>RTSHORT</td>
<td>50</td>
</tr>
<tr>
<td>RTREAL</td>
<td>12.775065</td>
</tr>
<tr>
<td>RTDOTE</td>
<td></td>
</tr>
<tr>
<td>RTLB</td>
<td></td>
</tr>
<tr>
<td>RTSHORT</td>
<td>51</td>
</tr>
<tr>
<td>RTREAL</td>
<td>11.674836</td>
</tr>
<tr>
<td>RTDOTE</td>
<td></td>
</tr>
</tbody>
</table>

For each dotted pair in the list that is returned, the first value is an integer code for the information type, and the second value is the information. The list format depends on the element type: node, link, or polygon.

**Information List Format for Nodes**
### Information List Format for Links

<table>
<thead>
<tr>
<th>Type</th>
<th>Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Persistent topology ID (RTREAL).</td>
</tr>
<tr>
<td>-1</td>
<td>Element type code (RTSHORT). With link lists, always 2, meaning <em>link element</em>.</td>
</tr>
<tr>
<td>-2</td>
<td>Entity name of the link object (RTREAL).</td>
</tr>
<tr>
<td>1</td>
<td>Topology ID of start node (RTREAL).</td>
</tr>
<tr>
<td>2</td>
<td>Topology ID of end node (RTREAL).</td>
</tr>
<tr>
<td>3</td>
<td>Topology ID of left polygon (RTREAL). Relevant only if the link belongs to a polygon topology. Links in a polygon topology can belong to two adjacent polygons, one on the left, and one on the right.</td>
</tr>
<tr>
<td>4</td>
<td>Topology ID of right polygon that shares this link (RTREAL). Relevant only if the link belongs to a polygon topology.</td>
</tr>
<tr>
<td>40</td>
<td>Forward resistance of the link (RTREAL).</td>
</tr>
<tr>
<td>41</td>
<td>Reverse resistance of the link (RTREAL).</td>
</tr>
<tr>
<td>70</td>
<td>Link direction (RTSHORT): -1, 0, or 1, where -1 = Reverse, 0 = Bidirectional, and 1 = Forward.</td>
</tr>
</tbody>
</table>

### Information List Format for Polygons

<table>
<thead>
<tr>
<th>Type</th>
<th>Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Persistent topology ID (RTREAL).</td>
</tr>
<tr>
<td>-1</td>
<td>Element type (RTSHORT). With polygon lists, always 3, meaning <em>polygon element</em>.</td>
</tr>
</tbody>
</table>
When a topology is built, it is given a set of object data fields. Their purpose is to contain the information listed in the preceding tables.

The following sample loads and then opens a network topology for read using `tpm_acload()` and `tpm_acopen()` respectively. A element id is obtained using `tpm_elemid()` with the topology id returned by `tpm_acopen()`. `Tpm_elemget()` is called with all required parameters. A resbuf is populated with information associated with the specified element, a portion of which is then displayed. The resbuf is then released as required.

```c
char* pszTopoName = "NetTopo";
int returnCode = tpm_acload(pszTopoName, NULL);
int topoWriteAccess = 0;
ade_id topoId = tpm_acopen(pszTopoName, topoWriteAccess);
int elementType = 2;
long elementIndex = 1;
ade_id elementID = tpm_elemid(topoId, elementType, elementIndex);
struct resbuf* pElementInfoRb = tpm_elemget(topoId, elementID);
if (NULL != pElementInfoRb){
    struct resbuf* rb = pElementInfoRb;
    while(NULL != rb) {
        if (rb->restype == RTSHORT && (rb->resval.rint == 0)) {
            acutPrintf("\nThe specified elements topology id is: %.0lf", rb->rbnext->resval.rreal);
            break;
        }
        rb = rb->rbnext;
    }
}
else {
    acutPrintf("\nThe specified element was not found.");
}
acutRelRb(pElementInfoRb);
tpm_acclose(topoId);
tpm_acunload(pszTopoName);
```
tpm_elemid

Element Information Functions

Gets the ID of an element.

```
ade_id
tpm_elemid(
    ade_id tpm_id,
    int type,
    long index);
```

Returns an `ade_id` or `ADE_NULLID`.

- **tpm_id**: The topology ID.
- **type**: An element type. Values can be: 1 (node), 2 (link), or 3 (polygon).
- **index**: Element index. The index of the first element is 0

The `ade_id` of a topology element is used by the functions `tpm_elemget`, `tpm_elemadj`, `tpm_editdelelem`, and `tpm_editmodelem`. 
**tpm_elemqty**

**Element Information Functions**

Counts topology elements.

```c
int tpm_elemqty(
    ade_id tpm_id,
    int type,
    long *qty);
```

Returns **RTNORM** or an error code.

- **tpm_id**: Topology ID
- **type**: Element type. Values can be: 1 (node), 2 (link), or 3 (polygon).
- **qty**: A pointer to the element count.
# tpm_elemss

**Element Information Functions**

Create a selection set of elements of a given type.

```c
int tpm_elemss(
    ade_id tpm_id,
    int type,
    ads_name ss);
```

Returns **RTNORM** or an error code.

<table>
<thead>
<tr>
<th>tpm_id</th>
<th>Topology ID.</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>Element type. Values can be: 1 (node), 2 (link), or 3 (polygon).</td>
</tr>
<tr>
<td>ss</td>
<td>Selection set ID.</td>
</tr>
</tbody>
</table>
Tests if a polygon topology is complete.

```c
int tpm_info_complete(  
    ade_id tpm_id);
```

Returns **TRUE** if the topology is complete or **FALSE**.

**tpm_id**  The topology ID of a polygon topology.

Use this function to verify that a polygon topology is complete before performing an element trace, overlay, or other topology operation. This function applies only to polygon topologies.

A topology is considered complete if all necessary entities exist in the current drawing. It is considered incomplete if at least one polygon is incomplete.

A query can bring an incomplete topology into the current drawing. Because this part has the same name as the complete topology that it came from, the part could be mistaken for the whole. Although you can execute any topology operation on an incomplete topology, the result may not be what you intend.
tpm_infocorrect

Topology Information Functions

Tests whether a topology is correct and performs a geometrical audit.

int
tpm_infocorrect(    
       ade_id tpm_id);

Returns TRUE or FALSE.

tpm_id  The topology ID.

If topology entities are changed using drawing tools, the topology information is modified according to the geometrical changes. If this is impossible, the topology becomes incorrect. Use this function to verify that a topology is correct before performing an element trace, overlay, or other topology operation. See also tpm_infomodified.

To fix an incorrect topology, try unloading and reloading it. If it does not reload, you must use topology edit to fix the topology. If this does not work, rebuild the topology.

Note  This function indicates when an error occurs, but does not show its location.
Checks the source from which the topology was loaded.

```c
int tpm_infocurrent ade_id tpm_id);
```

Returns 1 if the topology is loaded from the current drawing, or 0 if it comes from source drawings.

tpm_id Topology ID.
tpm_infodesc

Topology Information Functions

Gets a topology description.

```c
int tpm_infodesc(
    ade_id tpm_id,
    char *buffer,
    int buflen);
```

Returns RTNORM or an error code.

tpm_id   Topology ID.
buffer   Pointer to result buffer.
buflen   Result buffer length.
Checks if topology elements have been modified using drawing tools.

```c
int tpm_infomodified(  
    ade_id tpm_id);
```

Returns 1 if any elements have been modified, otherwise NULL.

**tpm_id**  The topology ID.

If topology elements have been modified using drawing tools, then possibly they are no longer correct topologies. See also [tpm_infocorrect](#).
**tpm_infoname**

*Topology Information Functions*

Gets a topology name.

```c
int tpm_infoname(
    ade_id tpm_id,
    char *buffer,
    int buflen);
```

Returns **RTNORM** or an error code.

- **tpm_id**  The topology ID.
- **buffer**  Pointer to result buffer.
- **buflen**  Result buffer length.
tpm_infostatus

Topology Information Functions

Checks whether the topology is open for Read or Write access.

int
tpm_infostatus(
    ade_id tpm_id);

Returns 1 if the topology is open for Write or 0 if open for Read.

tpm_id The topology ID.
tpm_infotype

Topology Information Functions

Gets a topology type.

int
tpm_infotype(
    ade_id tpm_id);

Returns a topology type code (1 = node, 2 = network, or 3 = polygon), or ADE_NULLID.

tpm_id    The topology ID.
tpm_infoversion

Topology Information Functions

Gets the version of a topology.

```c
int tpm_infoversion(
    ade_id tpm_id,
    char *buffer,
    int buflen);
```

Returns RTNORM or an error code.

**tpm_id**  The topology ID.

**buffer**  Pointer to result buffer.

**buflen**  Result buffer length.

A topology's version is the Data Extension version (ADE) in which the topology was created. For example, "2.026". This function gets the same result as `tpm_iterversion`. The difference is that `tpm_infoversion` requires a topology ID, and it works only on topologies that are open.
tpm_iterdesc

Topology Iterating Functions

Gets the description of a topology.

```
int tpm_iterdesc(
    ade_id iter_id,
    char *buffer,
    int buflen);
```

Returns RTNORM or an error code.

iter_id    Iterator ID.
buffer     Pointer to result buffer.
buflen     Result buffer length.

The function gets a description of the topology that the iterator points to.

For an example that shows how you can use Topology Iterating functions to find all the topologies the system knows about, see tpm_iterstart.
tpm_itername

Topology Iterating Functions

Gets the name of a topology.

int
tpm_itername(
    ade_id iter_id,
    char *buffer,
    int buflen);

Returns RTNORM or an error code.

iter_id       Iterator ID.
buffer        Buffer for result (char).
buflen        Length of buffer (int).

The function gets the name of the topology that the iterator points to.

For an example that shows how you can use Topology Iterating functions to find all the topologies the system knows about, see tpm_iterstart.
tpm_iternext

Topology Iterating Functions

Moves the iterator to the next topology.

```c
int tpm_iternext(
    ade_id iter_id);
```

Returns RTNORM or, if an error occurs or another topology is not present, RERROR.

`iter_id`  Iterator ID.

The first time this function is called after `tpm_iterstart`, it sets the iterator on the first topology definition.

For an example that shows how you can use Topology Iterating functions to find all the topologies the system knows about, see `tpm_iterstart`. 
Allocate a topology iterator.

```c
ade_id
tpm_iterstart(
    int source,
    int loaded);
```

Returns an iterator ID (ade_id) or ADE_NULLID.

- **source** Source flag. Values can be: 1 (iterate through the current and source drawings), or 0 (iterate through the current drawing only). The default is 0.
- **loaded** Loaded in memory flag. Values can be: 1 (iterate through topologies in memory only), or 0 (iterate through all topologies). The default is 0.

This function allocates an iterator and positions it before the first topology definition. This behavior has implications to remember when you use the function.

- Because `tpm_iterstart` always generates an iterator ID, even if the drawing has no topologies to iterate through, the function fails only when it is out of memory.
- Because `tpm_iterstart` positions the iterator before the first topology definition, the function cannot indicate whether any topologies exist in the drawing. The only way to determine whether the drawing has topologies is to call `tpm_internext`, which fails if no topology exists beyond the current position of the iterator.

You can have more than one iterator running at the same time.

The following sample shows how you can use Topology Iterating functions to find and then describe all the topologies in the current project.

```c
// Iterate through the current drawing only (default)
int nTopoInCurrentAndDwgSet = 0;
// Iterate through all topologies (default)
int nTopoInMemory = 0;
ade_id topoIteratorId = tpm_iterstart(
    nTopoInCurrentAndDwgSet,
```
nTopoInMemory);
int resultCode = RTERROR;
char* pszTopoName = "";
const int TOPONAMEMAXLEN = 17;
if (ADE_NULLID == topoIteratorId) {
    acutPrintf(
        "\nA topology iterator could not be generated.");
} else {
    resultCode = tpm_iternext(topoIteratorId);
    if (RTNORM == resultCode) {
        acutPrintf(
            "\nThe following named topologies were found in the current project:"
        );
        do {
           resultCode = tpm_itername(
                topoIteratorId,
                pszTopoName,
                TOPONAMEMAXLEN);
            int nTopoType = tpm_itertype(topoIteratorId);
            switch(nTopoType) {
                case 1: {
                    acutPrintf("\n\n\n%s" is a node topology.", pszTopoName);
                } break;
                case 2: {
                    acutPrintf("\n\n\n%s" is a network topology.", pszTopoName);
                } break;
                case 3: {
                    acutPrintf("\n\n\n%s" is a polygon topology.", pszTopoName);
                } break;
                default: {
                    acutPrintf("\nThe topology type can not be determined.");
                } break;
            }
        } while (RTNORM == (resultCode = tpm_iternext(topoIteratorId)));
    } else {
        acutPrintf("\nNo topologies were detected.");
    }
}
Frees an iterator.

```c
int tpm_iterstop(
    ade_id iter_id);
```

Returns RTNORM or an error code.

**iter_id**  Iterator ID.
tpm_itertype

*Topology Iterating Functions*

Gets the type of the topology.

```c
int
tpm_itertype(    
    ade_id iter_id);
```

Returns a topology type code (int) or ADE_NULLID.

**iter_id**  
Iterator ID.

The function gets the type of the topology that the iterator is pointing to. The topology type codes are 1 = node, 2 = network, 3 = polygon.

For an example that shows how you can use Topology Iterating functions to find all the topologies the system knows about, see [tpm_iterstart](#).
**tpm_iterversion**

*Topology Iterating Functions*

Gets the version of a topology.

```c
int tpm_iterversion(
    ade_id iter_id,
    char *buffer,
    int buflen);
```

Returns **RTNORM** or an error code.

- **iter_id**: Iterator ID.
- **buffer**: Buffer for result (char).
- **buflen**: Length of buffer (int).

The function gets the version of the topology that the iterator is pointing to. The version of a topology is the version of ADE in which it was created, for example, "2.026". The function gets the same result as **tpm_infoversion**. The difference is that **tpm_infoversion** requires a topology ID, and so it works only on topologies that are open.
tpm_mntbuild

Building and Erasing Functions

Builds a topology.

```c
int tpm_mntbuild(
    ade_id var_id,
    char *name,
    char *desc,
    int type,
    ads_name node_ss,
    ads_name link_ss,
    ads_name cntr_ss);
```

Returns RTNORM or an error code.

- **var_id**: Topology variables ID.
- **name**: Topology name.
- **desc**: Topology description.
- **type**: Topology type code. Values can be 1 (node), 2 (network), or 3 (polygon).
- **node_ss**: Entity selection set of nodes or NULL.
- **link_ss**: Entity selection set of links or NULL. Omit for node topology.
- **cntr_ss**: Entity selection set of centroids or NULL. Omit for node and network topology.

If you are building a node topology, the link_ss and cntr_ss arguments should be NULL. If you are building a network topology, the cntr_ss argument should be NULL.

The topology variables ID references a set of topology variables.

The new topology is loaded, but closed. You must open it with tpm_acopen.

The following AutoCAD object types are acceptable for topology elements:

- For links: line, arc, circle, 2D and 3D polyline
• For nodes and centroids: point, insert, and text

A node topology can contain only nodes. A network topology can contain nodes or links, but not centroids. A polygon topology can contain all three.

When the topology is built, all links are assigned a default direction of bidirectional (a value of 0). The forward and reverse resistance values are the length of the link. Nodes are assigned a resistance of 0.

When a topology is built, it is given a set of object data fields. These fields contain information about the elements of the topology.
Erases a closed topology from the current drawing.

```
int
tpm_mnterase(
    char *toponame);
```

Returns **RTNORM** or an error code.

- **toponame**    The name of the topology.

This function can erase a closed topology whether or not it is loaded.
tpm_mntrebuild
Building and Erasing Functions

Rebuilds a topology.

```
int tpm_mntrebuild(  
    char *toponame);  
```

Returns RTNORM or an error code.

**toponame**  The name of the topology.

The rebuilt topology is loaded, but closed. You must open it with `tpm_acopen`.

Rebuilding a topology restores all its object data fields to their default values. Any object data fields modified after the topology was built are lost. See Topology Object Data.

Whether you need to rebuild a topology after you change it depends upon the functions used to make the changes.

- If changes were made using AutoCAD drawing and editing functions, you may need to rebuild the topology.
  
  If the AutoCAD alterations introduce an error, the rebuild could fail. If this happens, you must clean the objects again and use `tpm_mntbuild`. You can use `tpm_infocorrect` to check for errors before attempting `tpm_mntrebuild`.

- If changes were made using Topology functions, you do not have to rebuild the topology. This applies to objects altered with functions such as `tpm_editaddelem`, `tpm_editdelelem`, and `tpm_editmodelem`. 
Renames a topology.

```c
int tpm_mntrename(
    char *toponame,
    char *newname,
    char *newdesc);
```

Returns RTNORM or an error code.

- **toponame**  The name of the topology you wish to rename.
- **newname**  The new name of the topology.
- **newdesc**  The new description of the topology.
tpm_qrygetresdesc

Topology Query Functions

Gets the description of the query result topology.

```
int tpm_qrygetresdesc(
    char *buffer,
    int buflen);
```

Returns RTNORM or an error code if there is no result topology or it has no description.

- **buffer**  
  Buffer for result description string (char).
- **buflen**  
  Length of buffer for string (int).

This function passes the topology description through a parameter.

For the **buffer** parameter, you must allocate at least the number of characters specified by the **buflen** parameter.
### tpm_qrygetrestopo

**Topology Query Functions**

Gets the name of the query result topology.

```c
int tpm_qrygetrestopo(
    char *buffer,
    int buflen);
```

Returns **RTNORM** or an error code if there is no result topology.

- **buffer**  Buffer for resulting topology name string (char).
- **buflen**  Length of buffer for string (int).

This function passes the topology name through a parameter.
tpm_qrygettoponame

Topology Query Functions

Gets the name of the query source topology.

int
int tpm_qrygettoponame(  
    char *toponame,  
    int buflen);  

Returns RTNORM or an error code if there is no query source topology.

toponame Pointer to topology name (char).

buflen Length of buffer for topology name string (int).

This function passes the topology name through a parameter.
tpm_qrysetrestopo

Topology Query Functions

Defines or or undefines a query result topology.

int
tpm_qrysetrestopo(
    char *result_name,
    char *result_desc);

Returns RTNORM or an error code.

result_name        The name of the query result topology or NULL.
result_desc        The description of the query result topology or NULL

A description is optional. Its presence or absence has no effect on the query. Regarding the name:

- If no name is specified, then no query result topology is defined. If a definition already exists, it is canceled.
- If the name begins with an asterisk (*), the next query result is a temporary topology.
- If the name begins with any other acceptable character, the next query result is a standard topology.
TPM_QRYSETTPONAME

Function: Defines or undefines a topology query.

```
int tpm_qrysettoponame(
    char *toponame);
```

Returns RTNORM or an error code.

- **toponame** The name of the topology to query, or NULL.

This function determines whether the current query is a topology query or a standard data extension query as follows:

- If the **toponame** argument is NULL, the current query becomes a standard data extension query.
- If the **toponame** argument is a name, the current query becomes a topology query. But if you specify a topology that does not exist, a subsequent call to ade_qryexecute returns NULL.

If you specify a topology that does not exist, a subsequent call to ade_qryexecute returns an error code.
tpm_tracealloc
Network Tracing Functions

Allocates the tracing model.

tpm_tracealloc(
    ade_id tpm_id,
    char *node_res,
    char *link_dir,
    char *link_forward_res,
    char *link_reverse_res);

Returns a tracing model ID or ADE_NULLID.

tpm_id The topology ID.
node_res Expression for node resistance. NULL = Default resistance.
link_dir Expression for link direction. NULL = Default direction.
link_forward_res Expression for forward link resistance. NULL = Default resistance.
link_reverse_res Expression for reverse link resistance. NULL = Default resistance.

This function sets the values for the specified topology to the parameters you enter. The values stored in topology object data when the topology was created are used as defaults if you omit parameters. You can enter any valid expression that evaluates to a numeric result.

The tracing model can be used only with a network or polygon topology.

When you enter a value other than NULL for any of the optional resistance arguments, this value is used for all objects in the topology of the appropriate type. It overrides the corresponding value attached to the object. For example, if node_res is set to NULL, the tracing model uses the value attached to topology object data when calculating the trace. If node_res is set to 10.0, all nodes in the topology are overridden with the value of 10.0 when the trace is calculated.

If you want to omit the optional parameters, you can either enter NULL to invoke a default value or leave out the parameter altogether, as with other Visual LISP functions. However, before you omit optional parameters, note the dependency relationships indicated by bracketed groups. For example, if you want to
use the `link_dir` parameter, you must enter a `node_res` parameter.

**Important!** When using the Topology API to perform a network trace, such as a Best Route analysis, the source topology used to create the trace topology should not be unloaded or erased until after all API calls relating to the trace have been made. This is because the trace topology references the nodes and links in the source topology. It does not create its own. So, in order to do anything with the elements of the trace, the source topology must remain loaded.

For example, the following Best Route code will silently fail:

```c
char* pszTopoName = "NetTopo";
int topoWriteAccess = 1;
int returnCode = tpm_acload(pszTopoName, NULL);
ade_id topoId = tpm_acopen(pszTopoName, topoWriteAccess);

ade_id networkTraceId = tpm_tracealloc(topoId, NULL, NULL, NULL, NULL);

struct resbuf* pNodeListRb = acutBuildList(
    RTREAL, 7.0,
    RTREAL, 1.0,
    RTREAL, 4.0,
    RTREAL, 9.0,
    RTREAL, 5.0,
    RTREAL, 6.0,
    RTREAL, 8.0,
    0);
ade_id bestRouteTraceId = tpm_tracebestroute(topoId, networkTraceId, pNodeListRb);

acutRelRb(pNodeListRb);
tpm_acclose(topoId);
tpm_acunload(pszTopoName);

ade_id BestRouteTraceElement = tpm_tracebestroutescan(networkTraceId, 3);
```

The simple solution is to defer the calls to `tpm_acclose` and `tpm_acunload` until after all `tpm_tracebestroutescan` calls.
tpm_tracebestroute

Network Tracing Functions

Calculates the best round-trip route.

```c
ade_id
tpm_tracebestroute
    ade_id tpm_id,
    ade_id trace_id,
    struct resbuf *nodes);
```

Returns a topology ID or ADE_NULLID.

- **tpm_id** The topology ID representing the network you are analyzing.
- **trace_id** The tracing model ID returned by `tpm_tracealloc`.
- **nodes** List of nodes to visit.

The list of nodes is implemented as a `resbuf` chain, which you can create like this:

```c
struct resbuf* pNodeListRb = acutBuildList(
    RTREAL, 8.0,// start node
    RTREAL, 1.0,// visit point 1
    RTREAL, 5.0,
    RTREAL, 11.0,
    RTREAL, 6.0,
    RTREAL, 7.0,
    RTREAL, 10.0,
    RTREAL, 13.0,
    RTREAL, 12.0,
    RTREAL, 9.0,
    RTREAL, 4.0,
    RTREAL, 2.0,// visit point 11
0);
```

The best route topology, whose ID this function returns if successful, is assigned an arbitrary name and is open for read. To get its name, use `tpm_infoname`. To change its name, use `tpm_mntrename`. 
For the best route trace to succeed, the total calculated resistance cannot be greater than the value set for the maximum resistance or less than the value set for the minimum resistance. See `tpm_tracesetmaxres` and `tpm_tracesetminres`. The accumulated resistance value is the total resistance of the nodes and links that make up the best route.

The following sample creates a best route using `tpm_tracebestroute()`. The `tpm_id` param is obtained using `tpm_acload()` and `tpm_acopen()`. The `trace_id` param is obtained using `tpm_tracealloc()`. A `resbuf` containing the list of nodes which serve as visit points is then created. `Tpm_tracebestroute()` is called with all required parameters, the returned value is checked against `ADE_NULLID`. A successful operation displays the name of the best route topology. The `resbuf` is then released as required.

```c
char* pszTopoName = "NetTopo";
int topoWriteAccess = 1;
int returnCode = tpm_acload(pszTopoName, NULL);
ade_id topologyId = tpm_acopen(pszTopoName, topoWriteAccess);
ade_id traceId = tpm_tracealloc(topologyId, NULL, NULL, NULL, NULL);
struct resbuf* pNodeListRb = acutBuildList(
    RTREAL, 8.0,
    RTREAL, 1.0,
    RTREAL, 5.0,
    RTREAL, 11.0,
    RTREAL, 6.0,
    RTREAL, 7.0,
    RTREAL, 10.0,
    RTREAL, 13.0,
    RTREAL, 12.0,
    RTREAL, 4.0,
    RTREAL, 2.0,
    0);
nade_id bestRouteTraceId = tpm_tracebestroute(topologyId, traceId, pNodeListRb);
if (bestRouteTraceId != ADE_NULLID) {
    char pszBestRouteName[25];
    int topNameLen = 25;
    int returnCode = tpm_infoname(bestRouteTraceId, pszBestRouteName, topoNameLen);
    if (RTNORM == returnCode) {
        acutPrintf("The best route is named: "%s",
                    pszBestRouteName);
    } else {
        acutPrintf("No best route could be calculated.");
    }
} else {
    acutPrintf("\nNo best route could be calculated.");
}
```
acutRelRb(pNodeListRb);
tpm_acclose(topoId);
tpm_acunload(pszTopoName);
tpm_tracebestroutroutescan

Network Tracing Functions

Gets the element ID of a link or node in the best route.

```c
ade_id
tpm_tracebestroutroutescan
    ade_id trace_id,
    int flag);
```

Returns a element ID or ADE_NULLID.

- **trace_id**: The tracing model ID returned by `tpm_tracealloc`.
- **flag**: Path element code. Values can be:
  - 0 = Current element
  - 1 = First element
  - 2 = Last element
  - 3 = Next element
  - 4 = Previous element

First use `tpm_tracebestroute` to calculate the best route.
Calculates the resistance of the best route.

```c
int tpm_tracebestrouteval
    ade_id trace_id,
    ads_real *resist);
```

Returns RTNORM or an error code.

- `trace_id`  
  Tracing model ID returned by `tpm_tracealloc`.

- `resist`    
  Resistance.

First use `tpm_tracebestroute` to calculate the best route.

This function passes the resistance of the best route through a parameter.
tpm_traceelemedit

Network Tracing Functions

Modifies a tracing model element.

```c
int tpm_traceelemedit(
    ade_id trace_id,
    ade_id elem_id,
    struct resbuf *new_val);
```

Returns RTNORM or an error code.

- `trace_id`: Model ID (returned by `tpm_tracealloc`).
- `elem_id`: Element ID.
- `new_val`: List consisting of a code for the property to modify and a new value for the property. See Properties and Values below.

Examples of building a `resbuf` for the `new_val` argument.

```c
// For an integer value
rb = acutBuildList(
    RTLB, RTSHORT, code, RTSHORT, value, RTDOTE, 0);
// For a real value
rb = acutBuildList(
    RTLB, RTSHORT, code, RTREAL, value, RTDOTE, 0);
```

You must release the `resbuf` when you are finished with it.

**Properties and Values**

<table>
<thead>
<tr>
<th>(40 . f_res)</th>
<th>Resistance of node (RTREAL), or forward resistance of link</th>
</tr>
</thead>
<tbody>
<tr>
<td>(41 . r_res)</td>
<td>Reverse resistance of link (RTREAL)</td>
</tr>
<tr>
<td>(70 . dir)</td>
<td>Link direction (RTSHORT):</td>
</tr>
<tr>
<td></td>
<td>-1 Reverse</td>
</tr>
</tbody>
</table>
The following sample uses `tpm_traceelemedit()` to modify an element created using `tpm_tracebestroute()`. The `trace_id` param is obtained using `tpm_tracealloc()` and `elem_id` is obtained using `tpm_elemid()`. A `resbuf` is created which contains the property/value pair required to modify the forward resistance of the specified element, (link). `Tpmp_traceelemedit()` is called with all required parameters, and the returned value is checked against `RTNORM`. A message indicating that the element has been modified is displayed, then the `resbuf` is released as required. Note, this example is based upon a trace which has been created, is loaded and open.

```c
char* pszTopoName = "BestRouteTopology1";
int topoWriteAccess = 1;
ad_ide_topoId = tpm_acopen(pszTopoName, topoWriteAccess);
ad_ide_traceId = tpm_tracealloc(topoId, NULL, NULL, NULL, NULL);
int elementType = 2;
long elementIndex = 5;
ad_ide_elementID = tpm_elemid(topoId, elementType, elementIndex);
struct resbuf* pModElementRb = acutBuildList(
    RTLB,
    RTSHORT, 40,
    RTREAL, 100.0,
    RTDOTE,
    0);
int returnCode = tpm_traceelemedit(traceId, elementID, pModElementRb);
if (RTNORM == returnCode) {
    acutPrintf(
        "The specified element has been modified.");
} else {
    acutPrintf(
        "The specified element was not modified.");
}
acutRelRb(pModElementRb);
tpm_acclose(topoId);
```
tpm_traceelemget

Network Tracing Functions

Lists information about a tracing model element.

```c
struct resbuf
*tpm_traceelemget(
    ade_id trace_id,
    ade_id elem_id);
```

Returns an information list in a `resbuf` or `NULL`

- `trace_id`    Tracing model ID returned by `tpm_tracealloc`
- `elem_id`    Trace element ID

The contents of a `resbuf` for a node element could be as follows.

<table>
<thead>
<tr>
<th>Restype</th>
<th>Resval</th>
</tr>
</thead>
<tbody>
<tr>
<td>RTLB</td>
<td></td>
</tr>
<tr>
<td>RTSHORT</td>
<td>-1</td>
</tr>
<tr>
<td>RTSHORT</td>
<td>1</td>
</tr>
<tr>
<td>RTDOTE</td>
<td></td>
</tr>
<tr>
<td>RTLB</td>
<td></td>
</tr>
<tr>
<td>RTSHORT</td>
<td>40</td>
</tr>
<tr>
<td>RTREAL</td>
<td>0.000000</td>
</tr>
<tr>
<td>RTDOTE</td>
<td></td>
</tr>
</tbody>
</table>

The contents of a `resbuf` for a link element could be as follows.

<table>
<thead>
<tr>
<th>Restype</th>
<th>Resval</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Column 1</td>
<td>Column 2</td>
</tr>
<tr>
<td>----------</td>
<td>----------</td>
</tr>
<tr>
<td>RTLB</td>
<td></td>
</tr>
<tr>
<td>RTSHORT</td>
<td>-1</td>
</tr>
<tr>
<td>RTSHORT</td>
<td>2</td>
</tr>
<tr>
<td>RTDOTE</td>
<td></td>
</tr>
<tr>
<td>RTLB</td>
<td></td>
</tr>
<tr>
<td>RTSHORT</td>
<td>1</td>
</tr>
<tr>
<td>RTREAL</td>
<td>9.000000</td>
</tr>
<tr>
<td>RTDOTE</td>
<td></td>
</tr>
<tr>
<td>RTLB</td>
<td></td>
</tr>
<tr>
<td>RTSHORT</td>
<td>2</td>
</tr>
<tr>
<td>RTREAL</td>
<td>18.000000</td>
</tr>
<tr>
<td>RTDOTE</td>
<td></td>
</tr>
<tr>
<td>RTLB</td>
<td></td>
</tr>
<tr>
<td>RTSHORT</td>
<td>40</td>
</tr>
<tr>
<td>RTREAL</td>
<td>3.000000</td>
</tr>
<tr>
<td>RTDOTE</td>
<td></td>
</tr>
<tr>
<td>RTLB</td>
<td></td>
</tr>
<tr>
<td>RTSHORT</td>
<td>41</td>
</tr>
<tr>
<td>RTREAL</td>
<td>3.000000</td>
</tr>
<tr>
<td>RTDOTE</td>
<td></td>
</tr>
<tr>
<td>RTLB</td>
<td></td>
</tr>
<tr>
<td>RTSHORT</td>
<td>70</td>
</tr>
<tr>
<td>RTSHORT</td>
<td>0</td>
</tr>
<tr>
<td>RTDOTE</td>
<td></td>
</tr>
</tbody>
</table>

The list format depends on the element type. For each a-list, the first component is an integer code for the information type, and the second is the information.
**List Format for Nodes**

<table>
<thead>
<tr>
<th>(-1 . elem_code)</th>
<th>Element type code (RTSHORT). With node lists, always 1, meaning node element.</th>
</tr>
</thead>
<tbody>
<tr>
<td>(40 . resistance)</td>
<td>Node resistance (RTREAL).</td>
</tr>
</tbody>
</table>

**List Format for Links**

<table>
<thead>
<tr>
<th>(-1 . elem_code)</th>
<th>Element type code (RTSHORT). With link lists, always 2, meaning link element.</th>
</tr>
</thead>
<tbody>
<tr>
<td>( 1 . topo_id)</td>
<td>Topology ID of start node (RTREAL).</td>
</tr>
<tr>
<td>( 2 . topo_id)</td>
<td>Topology ID of end node (RTREAL).</td>
</tr>
<tr>
<td>(40 . fwd_resist)</td>
<td>Forward resistance (RTREAL).</td>
</tr>
<tr>
<td>(41 . rev_resist)</td>
<td>Reverse resistance (RTREAL).</td>
</tr>
</tbody>
</table>
| (70 . link_dir)  | Link direction (RTSHORT): -1, 0, or 1. 
-1 = Reverse 
0 = Bidirectional 
1 = Forward |

The following sample opens a network trace topology for read using tpm_acopen(). A tracing model is allocated using tpm_tracealloc() with the topology id returned by tpm_acopen(). A trace element id is obtained using tpm_elemid(), which is used in the call to tpm_traceelemget(). A resbuf is populated with information associated with the specified element, a portion of which is then displayed. The resbuf is then released as required.

```c
char* pszTopoName = "BestRouteTopology1";
int topoWriteAccess = 0;
ade_id topoId = tpm_acopen(pszTopoName, topoWriteAccess);
ade_id traceId = tpm_tracealloc(topoId, NULL, NULL, NULL, NULL);
int elementType = 2;
long elementIndex = 5;
ade_id elementId = tpm_elemid(topoId, elementType, elementIndex);
struct resbuf* pTraceElementInfoRb = tpm_traceelemget(traceId, elementId);
if (NULL != pTraceElementInfoRb){
    struct resbuf* rb = pTraceElementInfoRb;
    while(NULL != rb) {
        if (rb->restype == RTSHORT) {
            const int nTraceVal = rb->resval.rint;
            switch(nTraceVal)
```
{ 
    case 1:
    acutPrintf("\nThe topology id of the starting node is: %0.1f" , rb->rbnext->resval.rreal);
    break;
    case 70:
    acutPrintf("\nThe element direction is: %d" , rb->rbnext->resval.rint);
    break;
    default:
    default: 
    break;
    }
}
rb = rb->rbnext;
}
} 
else {
    acutPrintf("\nThe specified element was not found.");
}
acutRelRb(pTraceElementInfoRb);
tpm_acclose(topoId);
Returns the ID of the selected element.

\[
\text{ade_id} \\
\text{tpm_traceelemid(} \\
\quad \text{ade_id trace_id,} \\
\quad \text{int type,} \\
\quad \text{long index);}
\]

Returns an element ID or \textbf{NULL}.

- \textbf{trace_id} \quad \text{Tracing model ID (real) returned by } \text{tpm_tracealloc}
- \textbf{type} \quad \text{Element type:}
  - 1 Node
  - 2 Link
- \textbf{index} \quad \text{Element index. The first element's index is } 0

This function returns the ID of the element selected. Use this function to find the trace result in the source topology after performing a trace with \text{tpm_traceshort} or \text{tpm_traceflood}.

The following sample opens a topology, performs a short path trace and gets the ID of the third link of the result path using \text{tpm_traceelemid()}. The topology ID for the corresponding link is displayed and the topology is unloaded.

```c
char* pszTopoName = "NetTopo";
int topoWriteAccess = 0;
int resultCode = tpm_acload(pszTopoName, NULL);
ade_id topoId = tpm_acopen(pszTopoName, topoWriteAccess);
ade_id traceId = tpm_tracealloc(topoId, NULL, NULL, NULL, NULL,
ade_id startNode = 8.0;
ade_id endNode = 13.0;
resultCode = tpm_traceshort(
    traceId,
    startNode,
    endNode);```
int elementType = 2; // link
long lLinkQuantity = 0;
resultCode = tpm_traceqty(
    traceId,
    elementType,
    &lLinkQuantity);

long lLinkIndex = 2;
ade_id linkId = tpm_traceelemid(
    traceId,
    elementType,
    lLinkIndex);
if (ADE_NULLID != linkId) {
    acutPrintf("\nThe specified link at index, (%d) contained the topology Id of %.0lf."
        , lLinkIndex, linkId);
} else {
    acutPrintf("\nThe specified link was not found.");
}
resultCode = tpm_acclose(topoId);
resultCode = tpm_acunload(pszTopoName);
Calculates flood paths.

```c
int tpm_traceflood(
    ade_id trace_id,
    ade_id start,
    ads_real maxres);
```

Returns RTNORM or an error code.

- `trace_id` The tracing model ID, returned by `tpm_tracealloc`.
- `start` The element ID of the start node.
- `maxres` The maximum allowed accumulated path resistance.

The accumulated resistance value is the total resistance of the nodes and links that make up the flood trace.
Frees a tracing model.

```c
int
tpm_tracefree(
    ade_id trace_id);
```

Returns RTNORM or an error code.

`trace_id` The tracing model ID returned by `tpm_tracealloc`. 
tpm_traceqty

Network Tracing Functions

Counts the selected elements after a trace.

```c
int tpm_traceqty(
    ade_id trace_id,
    int type,
    long *qty);
```

Returns RTNORM or an error code.

- **trace_id**: The tracing model ID, returned by `tpm_tracealloc`.
- **type**: Element type, either 1 (node) or 2 (link).
- **qty**: Element count.
Sets maximum resistance for the shortest path algorithm.

```c
int tpm_tracesetmaxres(  
    ade_id trace_id,  
    ads_real maxres);
```

Returns RTNORM or an error code.

- `trace_id` The tracing model ID, returned from `tpm_tracealloc`.
- `maxres` The maximum allowed accumulated path resistance.

This function sets an accumulated resistance value for path tracing.

For the shortest path trace to succeed, the total calculated resistance cannot be greater than the value set for the maximum resistance or less than the value set for the minimum resistance. See `tpm_tracesetminres`. See `tpm_traceshort`.
Sets minimum resistance for the shortest path algorithm.

```c
int tpm_tracesetminres(
    ade_id trace_id,
    ads_real minres);
```

Returns RTNORM or an error code.

- `trace_id` The tracing model ID, returned from `tpm_tracealloc`.
- `minres` The minimum accumulated allowed path resistance.

This function sets an accumulated resistance value for path tracing. For the shortest path trace to succeed, the total calculated resistance cannot be greater than the value set for the maximum resistance or less than the value set for the minimum resistance. See `tpm_tracesetmaxres`. See `tpm_traceshort`.
tpm_traceshort

Network Tracing Functions

Calculates the shortest path between two nodes.

```c
int tpm_traceshort(
    ade_id trace_id,
    ade_id start,
    ade_id end);
```

Returns RTNORM or an error code.

- `trace_id` The tracing model ID, returned by `tpm_tracealloc`.
- `start` The element ID of the start node.
- `end` The element ID of end node.

For the shortest path trace to succeed, the total calculated resistance cannot be greater than the value set for the maximum resistance or less than the value set for the minimum resistance. See `tpm_tracesetminres` and `tpm_tracesetmaxres`. The accumulated resistance value is the total resistance of the nodes and links that make up the shortest path.
tpm_traceshortscan

Network Tracing Functions

Gets the ID of a link or node in the shortest path.

```c
ade_id
tpm_traceshortscan(
    ade_id trace_id,
    int flag);
```

Returns a topology ID or `ADE_NULLID`.

<table>
<thead>
<tr>
<th>trace_id</th>
<th>The tracing model ID, returned by <code>tpm_tracealloc</code>.</th>
</tr>
</thead>
<tbody>
<tr>
<td>flag</td>
<td>Path element code. Values can be:</td>
</tr>
<tr>
<td>0</td>
<td>Current element</td>
</tr>
<tr>
<td>1</td>
<td>First element</td>
</tr>
<tr>
<td>2</td>
<td>Last element</td>
</tr>
<tr>
<td>3</td>
<td>Next element</td>
</tr>
<tr>
<td>4</td>
<td>Previous element</td>
</tr>
</tbody>
</table>

First use `tpm_traceshort` to calculate a shortest path.
Calculates the resistance of the shortest path.

```c
int
tpm_traceshortval(
    ade_id trace_id,
    ads_real *resist);
```

Returns RTNORM or an error code.

- **trace_id** The tracing model ID returned by `tpm_tracealloc`.
- **resist** The shortest path resistance.

Before calling this function, use `tpm_traceshort` to calculate the shortest path.
The functions for accessing topologies begin with *tpm_ac*.

<table>
<thead>
<tr>
<th>Function</th>
<th>Synopsis</th>
</tr>
</thead>
<tbody>
<tr>
<td>tpm_acclose</td>
<td>Closes a topology.</td>
</tr>
<tr>
<td>tpm_acexist</td>
<td>Checks if a topology exists.</td>
</tr>
<tr>
<td>tpm_acload</td>
<td>Loads a topology into memory.</td>
</tr>
<tr>
<td>tpm_acopen</td>
<td>Opens a topology.</td>
</tr>
<tr>
<td>tpm_acqty</td>
<td>Counts topologies.</td>
</tr>
<tr>
<td>tpm_acunload</td>
<td>Unloads a topology from memory.</td>
</tr>
<tr>
<td>tpm_acupgradeopen</td>
<td>Changes access from read only to write.</td>
</tr>
</tbody>
</table>
The functions for analyzing topologies begin with `tpm_ana`.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>tpm_anabuffer</code></td>
<td>Creates a buffer space around a topology.</td>
</tr>
<tr>
<td><code>tpm_anadissolve</code></td>
<td>Merges topology elements with the same value in the specified field.</td>
</tr>
<tr>
<td><code>tpm_anaoverlay</code></td>
<td>Overlays two topologies.</td>
</tr>
</tbody>
</table>
Building and Erasing Functions

Topology Function Synopsis

The functions for topology maintenance begin with `tpm_mnt`.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>tpm_mntbuild</code></td>
<td>Builds a topology.</td>
</tr>
<tr>
<td><code>tpm_mnterase</code></td>
<td>Erases a topology from the project drawing.</td>
</tr>
<tr>
<td><code>tpm_mntrebuild</code></td>
<td>Rebuilds a topology.</td>
</tr>
<tr>
<td><code>tpm_mntrename</code></td>
<td>Renames a topology.</td>
</tr>
</tbody>
</table>
The functions for editing topology elements begin with tpm_edit.

<table>
<thead>
<tr>
<th>Function</th>
<th>Synopsis</th>
</tr>
</thead>
<tbody>
<tr>
<td>tpm_editaddelem</td>
<td>Adds an element to a topology.</td>
</tr>
<tr>
<td>tpm_editdelelem</td>
<td>Deletes an element from a topology.</td>
</tr>
<tr>
<td>tpm_editmodelem</td>
<td>Modifies a topology element.</td>
</tr>
<tr>
<td>tpm_editupdelem</td>
<td>Updates a topology element.</td>
</tr>
</tbody>
</table>
The functions for managing topology elements begin with `tpm_elem`.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>tpm_elemadj</code></td>
<td>Compiles a list of adjacent elements.</td>
</tr>
<tr>
<td><code>tpm_elemfind</code></td>
<td>Finds an element.</td>
</tr>
<tr>
<td><code>tpm_elemget</code></td>
<td>Lists information about an element.</td>
</tr>
<tr>
<td><code>tpm_elemid</code></td>
<td>Gets the ID of an element.</td>
</tr>
<tr>
<td><code>tpm_elemqty</code></td>
<td>Counts topology elements.</td>
</tr>
<tr>
<td><code>tpm_elemss</code></td>
<td>Creates a selection set of elements of a given type.</td>
</tr>
</tbody>
</table>
The functions for iterating through topologies begin with `tpm_iter`.

Many have counterparts in the topology information functions (`tpm_info[xx]`). The iterating functions can query any topology, loaded or unloaded, open or closed. The information functions query only topologies that are open.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>tpm_iterdesc</code></td>
<td>Gets a topology description.</td>
</tr>
<tr>
<td><code>tpm_itername</code></td>
<td>Gets a topology name.</td>
</tr>
<tr>
<td><code>tpm_iternext</code></td>
<td>Moves the iterator to the next topology.</td>
</tr>
<tr>
<td><code>tpm_iterstart</code></td>
<td>Creates a topology iterator.</td>
</tr>
<tr>
<td><code>tpm_iterstop</code></td>
<td>Frees a topology iterator.</td>
</tr>
<tr>
<td><code>tpm_itertype</code></td>
<td>Gets a topology type.</td>
</tr>
<tr>
<td><code>tpm_iterversion</code></td>
<td>Gets the version of a topology.</td>
</tr>
</tbody>
</table>
The functions for network tracing begin with `tpm_trace`.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>tpm_tracealloc</code></td>
<td>Allocates a tracing model.</td>
</tr>
<tr>
<td><code>tpm_tracebestroute</code></td>
<td>Calculates the best round-trip route.</td>
</tr>
<tr>
<td><code>tpm_tracebestroutescan</code></td>
<td>Gets the element ID of a link or node in the best route.</td>
</tr>
<tr>
<td><code>tpm_tracebestrouteval</code></td>
<td>Calculates the resistance of the best route.</td>
</tr>
<tr>
<td><code>tpm_traceelemedit</code></td>
<td>Modifies a tracing element.</td>
</tr>
<tr>
<td><code>tpm_traceelemget</code></td>
<td>Gets information about a tracing element.</td>
</tr>
<tr>
<td><code>tpm_traceelemid</code></td>
<td>Gets the topology ID of a tracing element.</td>
</tr>
<tr>
<td><code>tpm_traceflood</code></td>
<td>Traces a flood path from a specified point.</td>
</tr>
<tr>
<td><code>tpm_tracefree</code></td>
<td>Frees a tracing model.</td>
</tr>
<tr>
<td><code>tpm_traceqty</code></td>
<td>Counts the selected elements after a trace.</td>
</tr>
<tr>
<td><code>tpm_tracesetmaxres</code></td>
<td>Sets maximum resistance for the shortest path algorithm.</td>
</tr>
<tr>
<td><code>tpm_tracesetminres</code></td>
<td>Sets minimum resistance for the shortest path algorithm.</td>
</tr>
<tr>
<td><code>tpm_traceshort</code></td>
<td>Calculates the shortest path between two nodes.</td>
</tr>
<tr>
<td><code>tpm_traceshortscan</code></td>
<td>Gets the topology ID of a link or node in the shortest path.</td>
</tr>
<tr>
<td><code>tpm_traceshortval</code></td>
<td>Calculates the resistance of the shortest path.</td>
</tr>
</tbody>
</table>
The functions for getting information about topologies begin with `tpm_info`.

Many have counterparts in the topology iterating functions (`tpm_iter[xx]`). The information functions can query only topologies that are open. The iterating functions can query any topology, loaded or unloaded, open or closed.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>tpm_infobuildvar</code></td>
<td>Stores the configuration values of a topology.</td>
</tr>
<tr>
<td><code>tpm_infocomplete</code></td>
<td>Tests if a topology is complete.</td>
</tr>
<tr>
<td><code>tpm_infocorrect</code></td>
<td>Tests if a topology is correct.</td>
</tr>
<tr>
<td><code>tpm_infocurrent</code></td>
<td>Checks the source from which a topology was loaded.</td>
</tr>
<tr>
<td><code>tpm_infodesc</code></td>
<td>Gets a topology description.</td>
</tr>
<tr>
<td><code>tpm_infomodified</code></td>
<td>Checks if topology elements have been modified using drawing tools.</td>
</tr>
<tr>
<td><code>tpm_infname</code></td>
<td>Gets a topology name.</td>
</tr>
<tr>
<td><code>tpm_infostatus</code></td>
<td>Checks whether a topology is open for Read or Write.</td>
</tr>
<tr>
<td><code>tpm_infotype</code></td>
<td>Gets a topology type.</td>
</tr>
<tr>
<td><code>tpm_infoversion</code></td>
<td>Gets a topology version.</td>
</tr>
</tbody>
</table>
The functions for querying topologies begin with `tpm_qry`.

<table>
<thead>
<tr>
<th>Function</th>
<th>Synopsis</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>tpm_qrygetresdesc</code></td>
<td>Gets the description of the query result topology.</td>
</tr>
<tr>
<td><code>tpm_qrygetrestopo</code></td>
<td>Gets the name of the query result topology.</td>
</tr>
<tr>
<td><code>tpm_qrygettoponame</code></td>
<td>Gets the name of the query source topology.</td>
</tr>
<tr>
<td><code>tpm_qrysetrestopo</code></td>
<td>Defines or or undefines a query result topology.</td>
</tr>
<tr>
<td><code>tpm_qrysettoponame</code></td>
<td>Defines or undefines a topology query.</td>
</tr>
</tbody>
</table>
The functions for managing configuration variables begin with `tpm_var`.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>tpm_varalloc</code></td>
<td>Allocates a set of configuration variables.</td>
</tr>
<tr>
<td><code>tpm_varfree</code></td>
<td>Frees a set of configuration variables.</td>
</tr>
<tr>
<td><code>tpm_varget</code></td>
<td>Gets the value of a configuration variable.</td>
</tr>
<tr>
<td><code>tpm_varlist</code></td>
<td>Gets all the values in a set of configuration variables.</td>
</tr>
<tr>
<td><code>tpm_varset</code></td>
<td>Sets the value of a configuration variable.</td>
</tr>
</tbody>
</table>
The **AcMapProject** class has new functions added for AutoCAD Map 3D 2005.

**Library**  rxapimap.lib  
**Include**  MapProj.h  

**Members**

[CreatePropertyAlterationDefinition Function 1](#)  [CreatePropertyAlterationDefinition Function 2](#)
New Functions for AcMapAttachedDrawing Class

**Classes and Namespaces**

The [AcMapAttachedDrawing](AcMapAttachedDrawing) class has a new function added for AutoCAD Map 3D 2005.

**Library**  rxapimap.lib

**Include**  MapArxApi.h

**Members**

[ApplyThisQuery Function](ApplyThisQuery Function)
The `AcMapExpression` class has a new function added for AutoCAD Map 3D 2005.

**Library**  rxapimap.lib

**Include**  MapArxApi.h

**Members**

**TopologyExecute Function**
New Functions for AcMapQuery Class

Classes and Namespaces

The AcMapQuery class has new functions added for AutoCAD Map 3D 2005.

Library  rxapimap.lib

Include  MapArxApi.h

Members

Execute Function 1  Execute Function 2
Execute Function 3
FileIn Function
FileOut Function
GetStringToDisplay Function
GetTopologyName Function
IsDefined Function
Select Function
SetTopologyName Function
The **AcMapPropertyAlterationDefinition** class has new functions added for AutoCAD Map 3D 2005.

**Library**  rxapimap.lib

**Include**  MapAlteration.h

**Members**

- **Apply Function 1**
- **Apply Function 2**
- **Erase Function**
- **ObjectId Function**
The **AcMapHatchAlteration** class has new functions added for AutoCAD Map 3D 2005.

**Library**  rxaimap.lib

**Include**  MapAlteration.h

**Inherits** from public AcMapPropertyAlteration

**Members**

- **Apply Function 1**
- **Apply Function 2**
- **Erase Function**
- **GetExcludedObjects Function**
- **SetExcludedObjects Function**
The **AcMapTextAlteration** class has new functions added for AutoCAD Map 3D 2005.

**Library**  rxapimap.lib

**Include**  MapAlteration.h

**Inherits**  from public AcMapPropertyAlteration

**Members**

- **Apply Function 1**  Apply Function 2
- **Erase Function**
ade_projentitybackward

Coordinate Transformation Functions

Transforms an entity from the destination coordinate system to the source coordinate system.

```c
int ade_projentitybackward(
    ads_name ent);
```

Returns RTNORM if successful.
Returns RTEERROR if failed.
Returns ADEMEMERROR if memory allocation failed.
Returns ADEINVERROR if acedInvoke() failed.

```c
ent        Input entity to be transformed.
```

The source and destination coordinate systems must be set previously by the ade_projsetsrc() and ade_projetdest() functions.
ade_projentityforward

Coordinate Transformation Functions

Transforms an entity from the source coordinate system to the destination coordinate system.

```c
int ade_projentityforward(  
    ads_name ent);
```

Returns RTNORM if successful.
Returns RTERROR if failed.
Returns ADEMEMERROR if memory allocation failed.
Returns ADEINVERROR if acedInvoke() failed.

`ent` Input entity to be transformed.

The source and destination coordinate systems must be set previously by the ade_projsetsrc() and ade_projsetdest() functions.
ade_projwsgeodistance
Coordinate Transformation Functions

Measures the geodetic distance between two points.

int ade_projwsgeodistance(
    ads_point pt1,
    ads_point pt2,
    ads_real* pdDistance,
    char** ppchDistUnits,
    int nDistUnitsBuffSize,
    ads_real* pdAzimuthForward,
    ads_real* ppdAzimuthReverse);

Returns RTNORM if successful.
Returns RTERROR if failed.
Returns ADEMEMERROR if memory allocation failed.
Returns ADEINVERROR if acedInvoke() failed.

pt1 Input first point.
pt2 Input second point.
pdDistance Output calculated geodetic distance between the points.
ppchDistUnits Output name of the distance units used in calculations. This argument is a pointer to the buffer (char**). If the buffer was not allocated, call this function with *ppchDistUnits = NULL. This function will allocate the needed buffer, and caller must free the memory that *ppchDistUnits points to. The returned display units depend on the AutoCAD Map menu setting: Map > Options > Coordinate Systems > Geodetic Distance > Units For Display.
nDistUnitsBuffSize Input size of the buffer to use to copy the name of the units if *ppchDistUnits != NULL and memory was allocated when function is called.
pdAzimuthForward Output angle, expressed in radians east of north, of the line from the first point to the second point, measured at the first point.
ppdAzimuthReverse Output angle, expressed in radians east of north, of the line from the second point to the first point, measured at the second point.
If no coordinate system is assigned, this function uses the Pythagorean system for calculations.
Creating Index Operation Expressions

```c
d锪x resbuf* pIndexParamsRb = acutBuildList(RTNIL, 0);
ade_dwgindexdef("", 0, pIndexParamsRb);
```

The following examples demonstrate creating index operation expressions.

Clears all previous index operation expressions.

```c
d锪x resbuf* pIndexParamsRb = acutBuildList(RTNIL, 0);
ade_dwgindexdef("Location", 1, pIndexParamsRb);
```

Specifies that the location index is to be generated.

```c
d锪x resbuf* pIndexParamsRb = acutBuildList(RTNIL, 0);
ade_dwgindexdef("Location", 0, pIndexParamsRb);
```

Specifies that the location index is to be removed.

```c
d亸x resbuf* pIndexParamsRb = acutBuildList(RTNIL, 0);
ade_dwgindexdef("Property", 1, pIndexParamsRb);
```

Specifies that the property index is to be generated.

```c
d亸x resbuf* pIndexParamsRb = acutBuildList(RTNIL, 0);
ade_dwgindexdef("Property", 0, pIndexParamsRb);
```

Specifies that the property index is to be removed.

```c
d亸x resbuf* pIndexParamsRb = acutBuildList(RTNIL, 0);
ade_dwgindexdef("ObjData", 1, pIndexParamsRb);
```

**Object Data Indexes**

If the `indexParams` argument is `RTNIL`, then indexes will be generated or removed for all tables and fields, for example:

Specifies that an object data index be generated on all tables and all fields.

```c
d亸x resbuf* pIndexParamsRb = acutBuildList(RTNIL, 0);
ade_dwgindexdef("ObjData", 1, pIndexParamsRb);
```
Specifies that all object data indexes for all tables and all fields be removed.

```c
struct resbuf* pIndexParamsRb = acutBuildList(RTNIL, 0);
ade_dwgindexdef("ObjData", 0, pIndexParamsRb);
```

If the `indexParams` argument includes only a table name with no field names specified, then indexes will be generated or removed for all fields in the table, for example:

Specifies that an object data index be generated on all fields in table TABLE.

```c
struct resbuf* pOdIndexParamsRb = acutBuildList(
    RTLB,
    RTLB,
    RTSTR, "TABLE",
    RTLE,
    RTLE,
    0);
ade_dwgindexdef("ObjData", 1, pOdIndexParamsRb);
```

Specifies that all object data indexes be removed from on all fields in table TABLE.

```c
struct resbuf* pOdIndexParamsRb = acutBuildList(
    RTLB,
    RTLB,
    RTSTR, "TABLE",
    RTLE,
    RTLE,
    0);
ade_dwgindexdef("ObjData", 0, pOdIndexParamsRb);
```

The list of fields in the `indexParams` argument can contain one or more fields, for example:

Specifies that an object data index be generated on field FIELD1 in table TABLE.

```c
struct resbuf* pOdIndexParamsRb = acutBuildList(
    RTLB,
    RTLB,
    RTSTR, "TABLE",
    RTSTR, "FIELD1",
    RTLE,
    RTLE,
    0);
ade_dwgindexdef("ObjData", 1, pOdIndexParamsRb);
```
Specifies that an object data index be removed from field FIELD1 in table TABLE.

```
struct resbuf* pOdIndexParamsRb = acutBuildList(
    RTLB,
    RTLB,
    RTSTR, "TABLE",
    RTSTR, "FIELD1",
    RTLE,
    RTLE,
    0);
ade_dwgindexdef("ObjData", 0, pOdIndexParamsRb);
```

Specifies that an object data index be generated on fields FIELD1 and FIELD2 in table TABLE.

```
struct resbuf* pOdIndexParamsRb = acutBuildList(
    RTLB,
    RTLB,
    RTSTR, "TABLE",
    RTSTR, "FIELD1",
    RTSTR, "FIELD2",
    RTLE,
    RTLE,
    0);
ade_dwgindexdef("ObjData", 1, pOdIndexParamsRb);
```

Specifies that an object data index be removed from fields FIELD1 and FIELD2 in table TABLE.

```
struct resbuf* pOdIndexParamsRb = acutBuildList(
    RTLB,
    RTLB,
    RTSTR, "TABLE",
    RTSTR, "FIELD1",
    RTSTR, "FIELD2",
    RTLE,
    RTLE,
    0);
ade_dwgindexdef("ObjData", 0, pOdIndexParamsRb);
```

It is also valid to pass multiple table field lists through the `indexParams` parameter, for example:

Specifies creating object data indexes on all fields in TABLE1 and on FIELD1 in table TABLE2.

```
struct resbuf* pOdIndexParamsRb = acutBuildList(
    RTLB,
    RTLB,
    RTSTR, "TABLE1",
    RTSTR, "FIELD1",
    RTLE,
    RTLE,
    0);
ade_dwgindexdef("ObjData", 1, pOdIndexParamsRb);
```
RTLB,
   RTSTR, "TABLE1",
RTLE,
RTLB,
   RTSTR, "TABLE2",
   RTSTR, "FIELD1",
   RTLE,
   RTLE,
   0);
ade_dwgindexdef("ObjData", 1, pOdIndexParamsRb);

Specifies deleting object data indexes from all fields in TABLE1 and from FIELD1 in table TABLE2.

struct resbuf* pOdIndexParamsRb = acutBuildList(
   RTLB,
   RTLB,
      RTSTR, "TABLE1",
      RTLE,
   RTLB,
      RTSTR, "TABLE2",
      RTSTR, "FIELD1",
      RTLE,
      RTLE,
      0);
ade_dwgindexdef("ObjData", 0, pOdIndexParamsRb);
The data types for object data fields are as follows. The type names in the first column are the coltype arguments for use in field definitions.

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>integer</td>
<td>From $-2,147,483,648$ through $2,147,483,647$</td>
</tr>
<tr>
<td>character</td>
<td>Any alphanumeric string, including the empty string, &quot;&quot;</td>
</tr>
<tr>
<td>point</td>
<td>List of three real numbers, separated by commas, that represent a point with an X, Y, and Z value, enclosed in quotes (a string value)</td>
</tr>
<tr>
<td>real</td>
<td>From $-1.7E308$ through $+1.7E308$</td>
</tr>
</tbody>
</table>
Location expressions are used as querycond arguments in ade_querydefine calls that define Location conditions.

There are a number of formats to choose from when writing Location expressions. They are listed below. The format to use depends on the Location type, which is identified in each of the following formats by the first argument. The other arguments are described below.

Location-All

("all")

Location-Bufferfence

("bufferfence" searchtype offset pt1 pt2 ... ptN)

Location-Circle

("circle" searchtype centerpt radius)

Location-Fence

("fence" pt1 pt2 ... ptN)

Location-Point

("point" pt)

Location-Polygon

("polygon" searchtype pt1 pt2 ... ptN)

Location-Polyline-Bufferfence

("polyline" "bufferfence" searchtype offset ename)

Location-Polyline-Fence
("polyline" "fence" ename)

Location-Polyline-Polygon

("polyline" "polygon" searchtype ename)

Location-Polygon-Polygon

("window" searchtype pt1 pt2)

**Location Expression Parameters**

- **searchtype**: Search type keyword (string): "inside" or "crossing".
- **offset**: Buffer offset distance (real).
- **ename**: AutoCAD entity name, or a set of points, or "?". If "?", when the query executes, it prompts the user to click a set of points.
- **pt ptN centerpt**: A 2D or 3D point (a list of reals). If a 3D point, the Z coordinate is ignored.
- **radius**: Radius (real).

**Location Examples**

The following examples define Location conditions. The first is a Location-All condition:

```
(ade_qrydefine
 ("AND" "NOT" "Location" ("All") "")
)
```

The second is a Location-Window condition:

```
(ade_qrydefine
 ("AND" "" "" "Location"
 ("window" "crossing" (1.0 2.0) (3.0 4.0)) "")
)
```

And the third is a Location-Polyline condition.

```
(ade_qrydefine
 (list "AND" "" "" "Location"
 (list "polyline" "bufferfence" "inside" 20.0
 (entlast)) "")
)
```
The following three examples of Location-Polyline conditions specify the `ename` argument in different ways. The first supplies an entity name:

```
(ade_qrydefine
  (list "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" 

And the third prompts the user to click a set of points:
(ade_qrydefine
  ('"" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" 

Property Expressions

ade_qrydefine

Property expressions are used as querycond arguments in ade_querydefine calls that define Property conditions.

They have the following format:

(property operator value [subclasses])

**Property Expression Parameters**

- **property**: Property name (**string**). See the Property and Value Arguments table below.
- **operator**: ",=", ",>", ",<", ",<=", ",>=", ",<>". Note that the only valid operator in a string context is ",=".
- **value**: Depends on the property argument. See the Property and Value Arguments table below.
- **subclasses**: Optional. **T** or **nil**. The default if the argument is omitted is **nil**. This setting has no effect unless property is "feature". **T** means return all objects belonging to the feature class identified by the value argument, including objects belonging to any subclass of that feature. **nil** means do not include objects belonging to such a subclass.

**Property and Value Arguments**

<table>
<thead>
<tr>
<th>Property</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>area</td>
<td>Area value (<strong>string</strong>).</td>
</tr>
<tr>
<td>blockname</td>
<td>Block name (<strong>string</strong>).</td>
</tr>
<tr>
<td>color</td>
<td>Color (<strong>string</strong>).</td>
</tr>
<tr>
<td>elevation</td>
<td>Z coordinate (<strong>string</strong>).</td>
</tr>
<tr>
<td>&quot;feature&quot;</td>
<td>Feature name (<strong>string</strong>).</td>
</tr>
<tr>
<td>group</td>
<td>Group name (<strong>string</strong>).</td>
</tr>
<tr>
<td>layer</td>
<td>Layer name (<strong>string</strong>).</td>
</tr>
</tbody>
</table>
length | Length (string).
linetype | Line type (string).
"lineweight" | Line weight (string).
"plotstyle" | Plot style (string).
style | Text style (string).
thickness | Thickness (string).
objtype | Object type (string), or "unknown".
value | Text value (string).

**Property Expression Examples**

The following examples define Property conditions. The first specifies a layer:

```
(setq qry_id
  (ade_qrydefine
    ('("and" "" "") "property" ("layer" ":=" "WATER") "")
  )
)
```

And the second specifies a color:

```
(setq qry_id
  (ade_qrydefine
    ('("or" "" "") "not" "property" ("color" ":=" "RED") "")
  )
)
```
Data Expressions

ade_qrydefine

Data expressions are used as querycond arguments in ade_querydefine calls that define Data conditions. They have the following format.

(datatype tablename.fieldname operator value [subclasses])

Data Expression Parameters

datatype  Data type to match (string): "objdata", "attrib", "aselink", "EED", or "feature".

tablename  Depends on the datatype argument. See the Tablename And Fieldname Arguments table below.

fieldname  Depends on the datatype argument. See the Tablename And Fieldname Arguments table below.

operator  Comparison operator (string): "=" , ">", "<", "<=>", ">=", or "<>". Note that the only valid operator in a string context is ">=".

value  Value to match.

subclasses  Optional. T or nil. The default if the argument is omitted is T. This setting has no effect unless datatype is "feature". T means return objects belonging to the feature class identified by the tablename argument, including objects belonging to any subclass of that feature. nil means do not include objects belonging to such a subclass.

The tablename and fieldname arguments depend on the datatype argument:

Tablename and Fieldname Arguments

<table>
<thead>
<tr>
<th>datatype</th>
<th>tablename</th>
<th>fieldname</th>
</tr>
</thead>
<tbody>
<tr>
<td>objdata</td>
<td>Table name.</td>
<td>Field name.</td>
</tr>
<tr>
<td>attrib</td>
<td>Block name.</td>
<td>Attribute definition.</td>
</tr>
<tr>
<td>aselink</td>
<td>Link template.</td>
<td>Column name.</td>
</tr>
<tr>
<td>EED</td>
<td>RegApp name.</td>
<td>EED field name.</td>
</tr>
</tbody>
</table>
**Data Expression Examples**

The following examples define Data conditions of various types.

```plaintext
(adef_qrydefine
 ("" " " " "Data"
 ("attrib" ".Type" "=" "c*""))"
)

(adef_qrydefine
 ("" " " " "Data"
 ("objdata" "mytable.fl1" "=" "1")"))"

(adef_qrydefine
 ("" " " " "Data"
 ("aselink" "cpu_lpn.cpu" "=" "MAC2LC")"))"

(adef_qrydefine
 ("" " " " "Data"
 ("EED" "REGAPP.STREET" "=" "Willow")"))"

(adef_qrydefine
 ("" " " " "Data"
 ("EED" "REGAPP.#NUMBER" "=" "512")"))"

(adef_qrydefine
 ("" " " " "Data"
 ("EED" "REGAPP.&SQLLINK," "=" "MAC2LC")"))"
```

Note in the last example that `&SQLLINK` is not treated as a link template key. It is treated just like any other ADE 1.0 EED field. For example, if an object has EED such as

```
(-3 (1000 . "&EEDFIELD;" = "1234"))
```

then the Data condition to retrieve the object is written as follows:

```
("
(EED" "REGAPP.&EEDFIELD;" =" "1234")"
```

If the EED is defined by

```
(-3 (1000 . "&EEDFIELD;" = "'First', 'Last'"))
```
then the Data condition to retrieve the object is written as follows:

```
("EED" "REGAPP.&EEDFIELD;" "=" "'First', 'Last'"
```

In other words, everthing after the equal sign is treated as one string. That way you can use any pattern (for `wcmatch`) in the query.
SQL expressions are used as querycond arguments in ade_querydefine calls that define SQL conditions. They have the following format:

(linkpathname sqlcondition)

**SQL Expression Parameters**

- **linkpathname**: Link template (string).
- **sqlcondition**: SQL condition (string).

The **sqlcondition** argument should contain only the WHERE clause of an SQL statement (for example, "last_name = 'Smith'"). To select the entire table, let the **sqlcondition** argument be the empty string ("").

(ade_qrydefine
  ('"AND""""""Sql"
   ("EMPLN3"""")"
  )
)
You can use the following comparison operators in range table expressions:

\(<\>\) \(\leq\) \(\geq\) \(\neq\) otherwise

When the first element in the range expression is "otherwise", the second element must be the empty string ("""). For example: ("otherwise" "" "red"). Although you do not specify a comparison value when you use the otherwise operator, a second list element is still required.

**Note** The not-equal operator, represented here by "/=", is represented by "<>" in other Data Extension functions.
Destroys an instance of this class.

\texttt{\textasciitilde DMScaleFactorReactor()};

Returns

Returns nothing.

Created with a commercial version of \texttt{Doc-O-Matic}. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at \texttt{support@toolsfactory.com}.
Constructs an instance of this class.

DMScaleFactorReactor();

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Invoked when the scale factor changes.

```cpp
virtual void ScaleFactorChanged() = 0;
```

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

```cpp
virtual ~AcMapContainer();
```

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbTileSet::AcMapContainer Class, AcMapMbTileSet::AcMapContainer Class
AcMapMbTileSet:: AcMapContainer:: AcMapContainer Constructor
AcMapMbTileSet::AcMapContainer Class | AcMapMbTileSet::AcMapContainer Class

Constructs an instance of this class.

AcMapContainer();
Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns the container iterator for this container.

```cpp
virtual Acad::ErrorStatus GetContainerIterator(
    AcMapContainerIterator*& pContIter
) const = 0;
```

**Parameters**

**Description**

- `pContIter`: Output container iterator pointer. The container iterator should be deleted when no longer needed.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code in which case the returned iterator will be NULL.

**Remarks**

Since the container represents a tree node, the container iterator it returns are the child and siblings of the node which do not directly contain tiles, but rather have children.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns the tile iterator for this container.

```cpp
virtual Acad::ErrorStatus GetTileIterator(
    AcMapTileIterator*& pTileIter
) const = 0;
```

**Parameters**

- **pTileIter**
  - Description: Output tile iterator pointer. The tile iterator should be deleted when no longer needed.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code in which case the returned iterator will be NULL.

**Remarks**

Since the container represents a tree node, the tile iterator it returns are the child and siblings of the node which do not directly contain tiles.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the Node name for the node associated with this container.

```cpp
virtual const ACHAR* Name() const = 0;
```

Returns

Returns the node name, or the empty string if the node has no name.

Remarks

Typically a node will have a name if it does not have a tile. For instance in a grid tiling scheme, typically the first level below the root will contain the rows. Each row will contain a child whose siblings contain tiles for each column in the row. The row nodes will typically be named with the Row name.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets a node name for the node associated with this container.

```cpp
virtual Acad::ErrorStatus SetName(
    const ACHAR * pszName
) = 0;
```

### Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pszName</td>
<td>Input tile name. The name should specify a valid node name. Node names follow esn rules.</td>
</tr>
</tbody>
</table>

### Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

### Remarks

Node names are typically used where the node represents the row or column or other subset of the tile name contained somewhere below the level of this container's node.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns the node associated with this container.

```cpp
virtual AcMapTreeNode* TreeNode() const = 0;
```

Returns

Returns the tree node for this container. If it fails for any reason it returns NULL.

Remarks

Any node.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

```
virtual ~AcMapContainerIterator();
```

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Constructs an instance of this class.

`AcMapContainerIterator();`  
Returns

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Links
AcMapMbTileSet::AcMapContainerIterator Class,
AcMapMbTileSet::AcMapContainerIterator Class
AcMapMbTileSet:: AcMapContainerIterator:: Done Method
AcMapMbTileSet::AcMapContainerIterator Class |
AcMapMbTileSet::AcMapContainerIterator Class

Tells the user if we are done.

**virtual bool** Done() **const** = 0;

Returns

Returns true if done; otherwise, returns false.

Created with a commercial version of *Doc-O-Matic*. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns the next node in the chain as a container.

```cpp
virtual bool Get(
    const AcMapContainer*& pContainer
) const = 0;
```

Parameters  
- **pContainer**: Output container.

Returns

Returns true if successful; otherwise, returns false to indicate there are no more containers.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbTileSet::AcMapContainerIterator Class,
AcMapMbTileSet::AcMapContainerIterator Class
AcMapMbTileSet:: AcMapContainerIterator:: Length Method
AcMapMbTileSet::AcMapContainerIterator Class | AcMapMbTileSet::AcMapContainerIterator Class

Returns the number of containers in the iterator.

```cpp
virtual unsigned int Length() const = 0;
```

Returns

Returns the size of the iterator.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Resets the iterator.

```cpp
virtual bool Rewind(  
    bool bReversed = false  
) const = 0;
```

**Parameters**
- **bReversed**: Direction of iteration.

**Returns**
- Returns true if successful; otherwise, returns false.

**Remarks**

Sets the iteration to go either forward or reversed. Default is forward.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Moves to the next node in the chain as a container.

```cpp
virtual bool Step() const = 0;
```

Returns true if successful; otherwise, returns false to indicate there are no more containers.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

**virtual** ~AcMapTileIterator();

Returns

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Constructs an instance of this class.

AcMapTileIterator();
Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Links
AcMapMbTileSet::AcMapTileIterator Class, AcMapMbTileSet::AcMapTileIterator Class
AcMapMbTileSet:: AcMapTileIterator:: Done Method
AcMapMbTileSet::AcMapTileIterator Class | AcMapMbTileSet::AcMapTileIterator Class

Tells the user if we are done.

**virtual bool** Done() const = 0;

Returns

Returns true if done; otherwise, returns false.

Created with a commercial version of [Doc-O-Matic](#). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns the current tile in the chain.

```cpp
virtual bool Get(
    const AcMapMbTile*& pTile
) const = 0;
```

**Parameters**
- `pTile`: Output Tile.

**Returns**

Returns true if successful; otherwise, returns false to indicate there are no more tiles.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns the node associated with the current tile in the chain.

```cpp
virtual bool GetNode(
    AcMapMbTileSet::AcMapTreeNode*& pNode
) const = 0;
```

Parameters

- `pNode`: Output Tile.

Returns

Returns true if successful; otherwise, returns false to indicate there are no more nodes.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns the number of containers in the iterator.

```cpp
virtual unsigned int Length() const = 0;
```

Returns the size of the iterator.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Reset the iterator.

```cpp
virtual bool Rewind(
    bool bReversed = false
) const = 0;
```

**Parameters**

- **bReversed** Direction of iteration.

**Returns**

Returns true if successful; otherwise, returns false.

**Remarks**

Sets the iteration to go either forward or reversed. Default is forward.

Created with a commercial version of [Doc-O-Matic](http://www.doc-o-matic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Sets a node name for the node associated with this tile.

```cpp
virtual Acad::ErrorStatus SetCurrentSecondaryIndexName(
    const ACHAR * pszName
) = 0;
```

### Parameters

- **pszName**: Input tile name. The name should specify a valid node name. Node names follow esn rules.

### Returns

Returns true if successful; otherwise, returns false.

### Remarks

This function is meant to handle grid tiling schemes where there is no place to store the value of the secondary index. The primary index is the container's name for the current tile iterator.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Moves to the next tile in the chain.

```cpp
virtual bool Step() const = 0;
```

Returns true if successful; otherwise, returns false to indicate there are no more tiles.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Destroys an instance of this class.

```cpp
virtual ~AcMapTreeNode();
```

Returns nothing.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Constructs an instance of this class.

```
AcMapTreeNode();
```

Returns

Returns nothing.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets a child for this node.

```cpp
virtual Acad::ErrorStatus AddChild(
    AcMapTreeNode * pChild,
    bool bAtEnd = false
) = 0;
```

**Parameters**

- **pChild**
  - Input tree node.
  - Input indicating where to put the added child. If true, then put the child at the end of the chain of the child of this node. If false, then insert the node as the child of this node and make the current child the first sibling in the chain.

**Returns**

- Returns Acad::eOk if successful; otherwise, returns a different error code.

**Remarks**

If there is already an existing non null child for this node, then this function will place the node at the beginning or end of the chain of siblings of the child of this node.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets a sibling for this node.

```cpp
virtual Acad::ErrorStatus AddSibling(
    AcMapTreeNode * pSibling,
    bool bAtEnd = false
) = 0;
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pSibling</td>
<td>Input sibling. Input indicating where to put the added sibling. If true,</td>
</tr>
<tr>
<td></td>
<td>then put the sibling at the end of the chain of siblings, if false, then</td>
</tr>
<tr>
<td></td>
<td>insert the node as the first sibling in the chain.</td>
</tr>
</tbody>
</table>

Returns

Returns Acad::eOk if successful; otherwise, returns a different error code.

Remarks

If the node already has a sibling, then this node is inserted into the sibling chain.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns the child of this node.

\[ \text{virtual AcMapTreeNode* Child() const = 0; } \]

Returns the tree node if there is a child to this node. otherwise, returns NULL.

Remarks

Any node contains one or zero children. The child may have siblings however.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns the const child of this node.

```cpp
virtual const AcMapTreeNode* ConstChild() const = 0;
```

Returns the tree node if there is a child to this node. otherwise, returns NULL.

Remarks

Any node contains one or zero children. The child may have siblings however.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns the const sibling of this node.

```cpp
virtual const AcMapTreeNode* ConstSibling() const = \0;
```

Returns

Returns the tree node if there is a sibling to this node. otherwise, returns NULL.

Remarks

Once the tile set has been constructed, the tile set should be accessed thru const member functions.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns the const tile of this node.

`virtual const AcMapMbTile* ConstTile() const = 0;`

Returns the tile if this node has a tile. otherwise, returns NULL.

Remarks

A tree node must contain either a child or a tile. But never contains both.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the Node name.

```cpp
virtual const ACHAR* Name() const = 0;
```

Returns the node name, or the empty string if the node has no name.

**Remarks**

Typically a node will have a name indicating the grouping for the tiles below it. For instance in a grid tiling scheme, typically the first level below the root will contain the rows. Each row will contain a child whose siblings contain tiles for each column in the row. The row nodes will typically be named with the Row name.

Created with a commercial version of [Doc-O-Matic](https://www.toolsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets a node name.

```cpp
virtual Acad::ErrorStatus SetName(
    const ACHAR * pszName
) = 0;
```

**Parameters**

- `pszName`: Input tile name. The name should specify a valid node name. Node names follow esn rules.

**Returns**

Returns Acad::eOk if successful; otherwise, returns a different error code.

**Remarks**

Node names may be used where the name indicates the category of the tiles below it in the tree.

Created with a commercial version of [Doc-O-Matic](https://toolsfactory.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets a sibling for this node.

**virtual Acad::ErrorStatus SetSibling(**

  AcMapTreeNode * pSibling **) = 0;**

**Parameters**

- **pSibling**
  - Input sibling.

**Returns**

- Returns true if successful; otherwise, returns false.

**Remarks**

If the node already has a sibling, then this node is inserted into the sibling chain.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Sets the tile for this node.

```cpp
virtual Acad::ErrorStatus SetTile(
    AcMapMbTile * pTile
) = 0;
```

**Parameters**
- `pTile`: Input tile. Current tile if any is discarded.

**Returns**
- Returns Acad::eOk if successful; otherwise, returns a different error code.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at [support@toolsfactory.com](mailto:support@toolsfactory.com).
Returns the sibling of this node.

```c++
virtual AcMapTreeNode* Sibling() const = 0;
```

Returns the tree node if there is a sibling to this node. otherwise, returns NULL.

Remarks

Once the tile set has been constructed, the tile set should be accessed thru const member functions. These functions might be used during construction.

Created with a commercial version of [Doc-O-Matic](https://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns the tile of this node.

```cpp
virtual AcMapMbTile* Tile() const = 0;
```

Returns the tile if this node has a tile. otherwise, returns NULL.

Remarks

A tree node must contain either a child or a tile. But never contains both.

Created with a commercial version of [Doc-O-Matic](http://www.docomatic.com). In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
Returns the owner tile set of this node.

```cpp
virtual const AcMapMbTileSet* TileSet() const = 0;
```

Returns the tile set if successful otherwise, returns NULL.

Created with a commercial version of Doc-O-Matic. In order to make this message disappear you need to register this software. If you have problems registering this software please contact us at support@toolsfactory.com.
This table shows plot set attribute names, descriptions, examples, and default values.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Query name (string), e.g., &quot;plotset2&quot;</td>
<td>&quot;&quot;</td>
</tr>
<tr>
<td>desc</td>
<td>Description (string), e.g., &quot;Complete Plot Set&quot;</td>
<td>&quot;&quot;</td>
</tr>
<tr>
<td>pnam</td>
<td>Plotter name (string), e.g., &quot;HP LaserJet III on LPT1&quot;</td>
<td>&quot;&quot;</td>
</tr>
<tr>
<td>pscr</td>
<td>Plot script (string), e.g., &quot;e, n, y&quot;</td>
<td>&quot;&quot;</td>
</tr>
<tr>
<td>block</td>
<td>Name of plot layout, a block name (string), e.g., &quot;TBLOCK&quot;</td>
<td>&quot;&quot;</td>
</tr>
<tr>
<td>atts</td>
<td>List of attributes for the block (string list), e.g., &quot;TITLE&quot;</td>
<td>nil</td>
</tr>
<tr>
<td>vlayer</td>
<td>Main viewport layer (string), e.g., &quot;VPORT2&quot;</td>
<td>&quot;&quot;</td>
</tr>
<tr>
<td>vdispl</td>
<td>Layers to disable for the main viewport (string), e.g., &quot;0&quot;</td>
<td>nil</td>
</tr>
<tr>
<td>kflg</td>
<td>Enables the reference viewport: T or nil&lt;br&gt;T Enabled  &lt;br&gt;nil Disabled&lt;br&gt;Note that there are two dependent attributes, &quot;klayer&quot; and &quot;kdispl&quot;</td>
<td>T</td>
</tr>
<tr>
<td>klayer</td>
<td>Key viewport layer (string), e.g., &quot;VPORT1&quot;, the layer of the reference viewport, which must be a different layer from the layer used for the main viewport (attribute &quot;vlayer&quot;); if the function map_pltblkvps returns only one viewport layer, it should be used for the main viewport</td>
<td>&quot;&quot;</td>
</tr>
<tr>
<td>kdispl</td>
<td>Key display layers (string list), e.g., &quot;BOUNDARY&quot;, a list of layers to be displayed in the referenced viewport. This should not be nil if &quot;kflg&quot; is enabled. Valid layers are all the layers returned by ade_dsproplist using the option &quot;layer&quot;</td>
<td>nil</td>
</tr>
<tr>
<td>kscl</td>
<td>Key viewport scale factor (real)</td>
<td>0.25</td>
</tr>
<tr>
<td>dwgs</td>
<td>List of source drawings (string list), e.g., &quot;DRAWING1.DWG&quot;&lt;br&gt;&quot;DRAWING2.DWG&quot;</td>
<td>nil</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
<td>Value</td>
</tr>
<tr>
<td>---------</td>
<td>-----------------------------------------------------------------------------</td>
<td>-------</td>
</tr>
<tr>
<td>dact</td>
<td>Force drawing active at plot time: T or nil</td>
<td>T</td>
</tr>
<tr>
<td>qcat</td>
<td>List of query catalogs (string list), paired with qnam list entries, e.g., &quot;Queries&quot;</td>
<td>nil</td>
</tr>
<tr>
<td>qnam</td>
<td>List of query names (string list), paired with qcat list entries, e.g., &quot;Query_One&quot;</td>
<td>nil</td>
</tr>
<tr>
<td>bdwg</td>
<td>Boundary drawing (string), e.g., &quot;BOUNDARY.DWG&quot;</td>
<td>&quot;&quot;</td>
</tr>
<tr>
<td>bnds</td>
<td>Boundaries (string list), the boundary or boundaries to use by providing the value for the field assigned in &quot;bnamf&quot;, e.g., &quot;Boundary #3&quot;</td>
<td>nil</td>
</tr>
<tr>
<td>blyr</td>
<td>Boundary layer (string), the name of the layer the boundary is placed on. It can be any layer of the boundary drawing. For example, &quot;BOUNDARY&quot;</td>
<td>&quot;&quot;</td>
</tr>
<tr>
<td>bnamt</td>
<td>Boundary name table (string), the name of the object data table attached to the boundary object, e.g., &quot;BOUNDARIES&quot;</td>
<td>&quot;&quot;</td>
</tr>
<tr>
<td>bnamf</td>
<td>Boundary name field (string), the name of the field to use (from the table assigned in &quot;bnamt&quot;) for the &quot;bnds&quot; attribute, e.g., &quot;name&quot;</td>
<td>&quot;&quot;</td>
</tr>
<tr>
<td>bodfs</td>
<td>Boundary fields (string list), the list of field names to be mapped to attributes in the title block, e.g., &quot;name&quot; (see details and example following this table); the title block is assigned using the attributes from the block assigned in &quot;block&quot;</td>
<td>nil</td>
</tr>
<tr>
<td>sflg</td>
<td>Plot to specified scale: T or nil</td>
<td>nil</td>
</tr>
<tr>
<td>scale</td>
<td>Plot scale (string), e.g., &quot;1:2000&quot;</td>
<td>1:1</td>
</tr>
<tr>
<td>clip</td>
<td>Clip objects against boundary (flag): T or nil</td>
<td>nil</td>
</tr>
<tr>
<td>pbnd</td>
<td>Plot boundary (flag): T or nil</td>
<td>nil</td>
</tr>
<tr>
<td></td>
<td>T Plot boundary polylines are plotted</td>
<td></td>
</tr>
<tr>
<td></td>
<td>nil Plot boundary polylines are erased before plotting</td>
<td></td>
</tr>
<tr>
<td>bbuf</td>
<td>Buffer boundary: T or nil</td>
<td>nil</td>
</tr>
<tr>
<td>btyp</td>
<td>Buffer type: &quot;true&quot; or &quot;rect&quot;</td>
<td>true</td>
</tr>
<tr>
<td></td>
<td>&quot;true&quot; Offsets the minimum bounding rectangle for a map boundary polyline</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;rect&quot; Specifies an offset of an existing plot boundary line</td>
<td></td>
</tr>
<tr>
<td>bdlist</td>
<td>Buffer distance (real)</td>
<td>0.10</td>
</tr>
</tbody>
</table>
Mapping of object data fields to block attributes is managed by the two plot attributes "bodfs" and "atts". The mapping is best explained by example. Suppose the following conditions:

- The boundary object data table, defined by the "bnamf" attribute, has the following fields: Field1, Field2, Field3, and Field4.
- The layout block, defined by the "block" attribute, has the following attributes: Attr1, Attr2, Attr3, Attr4, and Attr5.
- You want the value of Field1 to appear in place of Attr2, the value of Field2 in place of Attr3, and value of Field4 in place of Attr1.

To define the desired mapping, set the "bodfs" attribute to ("field1" "field2" "field4") and the "atts" attribute to ("attr2" "attr3" "attr1"). When plotting is done, values of the fields from the object data table attached to the boundary's closed polyline will be assigned to the specified block attributes. The number of elements in each list for "bodfs" and "atts" must be the same.
AcMapOSEObject::AddToEditSet

AcMapOSEObject class

Adds the object's AutoCAD counterpart to the EditSet.

bool
AddToEditSet();

Returns True if successful.

If an object is added to this user's EditSet, it is locked for other users.
AcMapOSEObject::FeatureName

AcMapOSEObject class

Gets the object's feature type.

const char *
FeatureName();

Returns the feature type.
AcMapOSEObject::IsErased

AcMapOSEObject class

Tells if the object's AutoCAD counterpart has been erased.

bool
IsErased();

Returns True if the AutoCAD counterpart has been erased.
Tells if the object’s AutoCAD counterpart is in the EditSet.

```cpp
bool IsInEditSet();
```

Returns true if the AutoCAD counterpart is in the EditSet.

If the object is in this user's EditSet, it is locked for other users.
Tells if the object's AutoCAD counterpart has been modified.

```cpp
bool IsModified();
```

Returns true if the AutoCAD counterpart has been modified.
Tells if the object is up to date.

```cpp
bool IsUpToDate();
```

Returns true if the object is up to date.

An object is up to date if its AutoCAD and Oracle counterparts have the same version number.
Removes the object's AutoCAD counterpart from the EditSet.

```cpp
bool RemoveFromEditSet();
```

Returns true if successful.

If an object is removed from this user's EditSet, it is unlocked for other users.
gets the version number of the object's AutoCAD counterpart.

unsigned long
Version();

Returns the version number, an integer starting with 1.
AcMapOSEObject::WhoHasIt

Gets information about the user who has this object in their EditSet.

```cpp
bool
WhoHasIt(
    unsigned long & UserId,
    std::string & strNtUser,
    std::string & strDbUser,
    std::string & strMapUser,
    std::string & strComputer,
    std::string & strLoginTime);
```

Returns True if successful.

- **UserId**: Owner's user ID.
- **strNtUser**: Owner's NT user name.
- **strDbUser**: Owner's Oracle user name.
- **strMapUser**: Owner's AutoCAD Map user name.
- **strComputer**: Owner's computer name.
- **strLoginTime**: Owner's log in time.

If this user cannot add an object to their EditSet, it may be in someone else's EditSet. WhoHasIt() gets information about the object's owner.
Adds drawing objects to the EditSet.

```cpp
bool AddToEditSet(
    const AcDbObjectIdArray & acadIds,
    AcDbObjectIdArray & arrFilteredOut);
```

Returns true if successful.

- `acadIds`: Selection set of objects to add.
- `arrFilteredOut`: Selection set of objects that could not be added. Normally because the objects were in the EditSet already, or they were in some other user's EditSet, or they were not queried objects.
Given a selection set of drawing objects, compiles lists of which are new (and of a certain feature type, or types), and which are not.

```cpp
bool FilterNewObjects(
    const std::vector<_string class="calibre8"> & vFeatures,
    const AcDbObjectIdArray & arrInput,
    AcDbObjectIdArray & arrNew,
    AcDbObjectIdArray & arrFilteredOut);
```

Returns true if successful.

- **vFeatures**: Input list of feature types. The output list, arrNew, will include objects of these feature types only. If the vFeatures list is empty, the output list will include objects of any valid feature type.
- **arrInput**: Input selection set of drawing objects to filter. If the arrInput list is empty, the selection set includes all objects in the drawing.
- **arrNew**: Output list of arrInput objects that are new and of vFeatures feature type(s).
- **arrFilteredOut**: Output list of arrInput objects not included in arrNew.

Objects in the project drawing are considered new if they are valid features and they are not in the project's Oracle database. An object is a valid feature if it resides on a drawing layer associated with a feature type.
Given a selection set of drawing objects, compiles lists of which are queried (and of a certain feature type, or types, and EditSet status), and which are not.

```cpp
bool FilterQueriedObjects(
    const std::vector<_string class="calibre8"> & vFeatures,
    const AcDbObjectIdArray & arrInput,
    AcDbObjectIdArray & arrErased,
    AcDbObjectIdArray & arrModified,
    AcDbObjectIdArray & arrUnchanged,
    AcDbObjectIdArray & arrFilteredOut,
    int nOptions);
```

Returns true if successful.

**vFeatures**
Input list of feature types. The output lists, arrErased, arrModified, and arrUnchanged, will include objects of these feature types only. If the vFeatures list is empty, the output list will include objects without regard to feature type.

**arrInput**
Input selection set of drawing objects to filter. If the arrInput list is empty, the selection set includes all objects in the drawing.

**arrErased**
Output list of arrInput objects that are queried and erased, and of vFeatures feature type(s) and nOptions edit-set status.

**arrModified**
Output list of arrInput objects that are queried and modified, and of vFeatures feature type(s) and nOptions edit-set status.

**arrUnchanged**
Output list of arrInput objects that are queried and unchanged, and of vFeatures feature type(s) and nOptions edit-set status.

**arrFilteredOut**
Output list of arrInput objects not included in arrErased, arrModified, or arrUnchanged.

**nOptions**
Input edit-set status options, a bitwise OR combination of AcMapOSEProject::EProjectOptions enumerators expressing whether output lists should include only objects in the EditSet, only objects not in the EditSet, or both.

Objects in the project drawing are considered queried if they are in the project's Oracle database. Objects in this user's EditSet are locked for other users. Objects not in this user's EditSet are unlocked for other users.

For information about edit-set status options ("project options"), see [AcMapOSEProject::EProjectOptions enumeration](#).
Removes drawing objects from the EditSet.

```c++
bool RemoveFromEditSet(
    const AcDbObjectIdArray & acadIds,
    AcDbObjectIdArray & arrFilteredOut);
```

Returns true if successful.

- **acadIds**: Input selection set of drawing objects to remove.
- **arrFilteredOut**: Output selection set of objects that could not be removed. Normally because they were not in the EditSet to begin with.
### AcMapOSEProject::EProjectOptions enumeration

<table>
<thead>
<tr>
<th>Code</th>
<th>Enum Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x1</td>
<td>kInEditSet</td>
<td>Object is in this user's EditSet (i.e., it is locked). Only this user can edit it.</td>
</tr>
<tr>
<td>0x2</td>
<td>kNotInEditSet</td>
<td>Object is not in this user's EditSet (i.e., it is unlocked). Any user can lock it for editing.</td>
</tr>
</tbody>
</table>
The following sample illustrates importing data from an Oracle Spatial database to a project drawing. It is in two parts. First, it defines a query to specify a condition for the objects to import, and then it imports them.

**Note** The sample assumes that there is an open project drawing, and that an import query has been defined already through the Import dialog box in the UI (choose Map > OracleSpatial > ImportObjectsFromOracle). It also assumes that the connection object is actually connected to a database. Click "Connecting to an Oracle Spatial Database" for sample code.

**Connecting to an Oracle Spatial Database (sample)**  
**Querying and importing data (overview)**

**Related samples**

```cpp
#include "StdAfx.h"
#include "StdArx.h"
#include "AcMapOracleConnection.h"
#include "AcMapOracleQuery.h"
#include "AcMapOracleImport.h"

BOOL ImportFromOracle()
{
    // Get the connection object
    AcMapOracleConnection *pConnection = AcMapOracleGetConnection();

    ********************
    ******* Query *******
    **********************

    // Instantiate a query object
    AcmapOracleQuery *pQuery =
        new AcMapOracleQuery(pConnection);

    // Initialize query object with current import query,
    // which has been defined already through the UI
    pQuery->InitWithCurrent(      acdbHostApplicationServices()->workingDatabase());

    // Or initialize query with specific condition
    pQuery->Init("EntityType = 'AcDbCircle' ");

    // You can check the whole query by
    // using ConvertToSql function
    char *pSql = pQuery->ConvertToSql();
    acutPrintf("\nQuery in SQL format:%s", pSql);

    // Save the query to the Oracle Spatial database
    pQuery->Save("MyQuery");

    // Load the saved query in a new query object
    AcMapOracleQuery *pNewQuery =
        new AcMapOracleQuery(pConnection);
pNewQuery->Load("MyQuery");

// Compare old and new query
char *pNewSql = pNewQuery->ConvertToSql();

if(strcmp(pSql, pNewSql) == 0)
    acutPrintf("Query Load/Save works!!\n");
else
    acutPrintf("Query Load/Save failed!!\n");

/**********************/
/******* Import *******/

// Instantiate an import object
AcMapOracleImport *pImport =
    new AcMapOracleImport(pConnection);

// Use the query object defined above to
// import Oracle Spatial data
if(pImport->CheckImport())
{
    if(pImport->Import(pQuery, false))
        acutPrintf("\nImported successfully\n");
    else
        acutPrintf("\nImport failed\n");
}
else
    acutPrintf("\nCheckImport() failed, can not import!\n");
}
Connecting to an Oracle Spatial Database

The following code sample illustrates connecting to an Oracle Spatial database and adding custom reactors. Custom reactor declarations are in MyMapOracleReactors.h, one of the files included at the beginning of the sample. The “Subclassing Custom Reactors” sample shows what such a file would contain.

Managing the connection (overview) Subclassing Custom Reactors (sample)
Other related samples

```cpp
#include "StdAfx.h"
#include "StdArx.h"
#include "AcMapOracleConnection.h"
#include "MyMapOracleReactors.h"  // Custom reactors

BOOL ConnectToOracle()
{
    // Get the connection object
    AcMapOracleConnection *pConnection = AcMapOracleGetConnection();

    // Initialize reactors
    // Custom reactor types are declared in MyMapOracleReactors.h
    MyConnectionReactor *pConnectReactor = new MyConnectionReactor();
    MyExportReactor *pExportReactor = new MyExportReactor();
    MyImportReactor *pImportReactor = new MyImportReactor();

    // Add reactors to connection class
    pConnection->AddConnectionReactor(pConnectReactor);
    pConnection->AddExportReactor(pExportReactor);
    pConnection->AddImportReactor(pImportReactor);

    BOOL bConnect = FALSE;

    // If no database is connected, connect one
    if (!pConnection->IsConnected())
    {
        bConnect = pConnection->Connect("Scott","TestDB","Scott","Tiger");
    }
    else
    {
        acutPrintf("Already connected to Oracle!!");
        return FALSE;
    }

    // Check if schema is valid
    if(bConnect)
    {
        if (pConnection->IsSchemaValid())
            acutPrintf("\nSchema is valid for this connection\n");
        else
            acutPrintf("\nSchema is not valid\n");
    }
```
// Get the service name
char *pService = 0;
pService = new char[80];
pService = pConnection->Service();
acutPrintf("\nService is %s\n", pService);
delete [] pService;

// Get the schema name
char *pSchema = 0;
pSchema = new char[80];
pSchema = pConnection->Schema();
acutPrintf("\nSchema is %s\n", pSchema);
delete [] pSchema;

// Get the user name
char *pUserName = 0;
pUserName = new char[80];
pUserName = pConnection->UserName();
acutPrintf("\nUserName is %s\n", pUserName);
delete [] pUserName;

// Disconnect the database
if(pConnection->Disconnect())
    acutPrintf("\nDisconnected successfully from Oracle\n");
else
    acutPrintf("\nError disconnecting from Oracle\n");

// Remove reactors from connection interface
pConnection->RemoveConnectionReactor(pConnectReactor);
pConnection->RemoveExportReactor(pExportReactor);
pConnection->RemoveImportReactor(pImportReactor);
The following code sample illustrates exporting data from a project drawing to an Oracle Spatial database.

**Note** The sample assumes an open project drawing containing some objects. It also assumes that the connection object is actually connected to a database. Click "Connecting to an Oracle Spatial Database" for sample code.

**Connecting to an Oracle Spatial Database**
**Exporting data (overview)**
**Related samples**

```cpp
#include "StdAfx.h"
#include "StdArx.h"
#include "AcMapOracleConnection.h"
#include "AcMapOracleExport.h"

BOOL ExportToOracle()
{
    // Get the connection object
    AcMapOracleConnection *pConnection = AcMapOracleGetConnection();

    // Instantiate an export object
    AcMapOracleExport *pExport = new AcMapOracleExport(pConnection);

    bool bExport = false;

    // Export 1
    // Default options...
    //   From all layers, export
    //   object data, block attributes, link templates
    // Erase drawing objects after export
    bExport = pExport->ExportObjectsAll();

    if(bExport)
        acutPrintf("\nExport 1 successful!!\n");
    else
        acutPrintf("\nExport 1 failed\n");

    // Export 2
    //   From selected layers 0, Line, and Arc, export
    //   object data, block attributes, link templates
    // Erase drawing objects after export
    bExport = pExport->ExportObjectsAll("0,Line,Arc",
                                       AcMapOracleExport::kBlockAttributes |
                                       AcMapOracleExport::kObjectData |
                                       AcMapOracleExport::kLinkTemplates |
                                       AcMapOracleExport::kEraseExported);

    if(bExport)
        acutPrintf("\nExport 2 successful!!\n");
```
else
    acutPrintf("\nExport 2 failed!\n");

// Export 3
// From all layers export all objects, but not
// object data, block attributes, link templates
// With erased objects, delete corresponding record
// Erase drawing objects after export

bExport = pExport->ExportObjectsAll(
    "*",
    AcMapPracleExport::kUpdateErased |
    AcMapPracleExport::kEraseExported );

if(bExport)
    acutPrintf("\nExport 3 successful!!\n");
else
    acutPrintf("\nExport 3 failed!\n");

// Export 4
// Export objects in an ObjectIdArray

AcDbObjectIdArray idArray;

// idArray can be filled by manually
// selecting Objects from Model space
// or iterating thorough Model Space
// and selecting all

bExport = pExport->ExportObjects(
    idArray,
    AcMapPracleExport::kBlockAttributes |
    AcMapPracleExport::kObjectData |
    AcMapPracleExport::kLinkTemplates |
    AcMapPracleExport::kEraseExported );

if(bExport)
    acutPrintf("\nExport 4 successful!!\n");
else
    acutPrintf("\nExport 4 failed!\n");

// Export 5
// Insert new rows in database (objects are new)
// From all layers export all objects, and include
// object data, block attributes, link templates
// Erase drawing objects after export

bExport = pExport->ExportObjects(
    "*",
    AcMapPracleExport::kDisregardIDs |
    AcMapPracleExport::kBlockAttributes |
    AcMapPracleExport::kObjectData |
    AcMapPracleExport::kLinkTemplates |
    AcMapPracleExport::kEraseExported );

if(bExport)
acutPrintf("\nExport 5 successful!!\n");
else
    acutPrintf("\nExport 5 failed!\n");
}
The following sample demonstrates getting an entity's Oracle ID if its AutoCAD ID is known, and vice versa. It also demonstrates that round trips with corresponding IDs are reliable — that if you use an AutoCAD ID to get an Oracle ID, and then use the Oracle ID to get an AutoCAD ID, the two AutoCAD IDs are the same.

**Note** The sample assumes an open project drawing containing imported entities. It also assumes that the connection object is actually connected to a database. Click "Connecting to an Oracle Spatial Database" for sample code.

```cpp
#include "StdAfx.h"
#include "StdArx.h"
#include "AcMapOracleConnection.h"
#include "AcMapOracleIdentification.h"

BOOL Identification()
{
    // Get the connection object
    AcMapOracleConnection *pConnection = AcMapOracleGetConnection();

    // Create an identification object
    AcmapOracleIdentification *pId = new AcMapOracleIdentification(pConnection);

    AcDbObjectId AcadId;
    AcDbEntity *pEntity;
    ads_name en;
    ads_point pt;
    unsigned long ulOracleID;

    // Get an entity from the user
    if((acedEntSel("Select an entity: ", en, pt) == RTNORM))
    {
        // Get the entity's AutoCAD ID
        acdbGetObjectId(AcadId, en);

        // Initialize the identification object with the AutoCAD ID
        bInit = pId->Init(AcadId);

        // Get the corresponding Oracle ID
        ulOracleID = pId->GetOracleID();

        char pBuf[100];

        // Get the entity's handle and print it to the AutoCAD Map text window
        if (Acad::eOk == acdbOpenAcDbEntity( pEntity, AcadId, AcDb::kForRead ))
        {
            AcadId.handle().getIntoAsciiBuffer(pBuf);
        }
    }
}
```
acutPrintf("\nAcad handle sel:%s", pBuf);
pEntity->close();
}

// Initialize the identification object with the Oracle ID
bInit = pId->Init(ulOracleID);

// Get the corresponding AutoCAD ID
AcadId = pId->GetAcadID(
    acdbHostApplicationServices()->workingDatabase());

// Get the entity's handle and print it to the AutoCAD Map text window
if (Acad::eOk == acdbOpenAcDbEntity( pEntity, AcadId, AcDb::kForRead ))
{
    AcadId.handle().getIntoAsciiBuffer(pBuf);
    acutPrintf("\nAcad handle sel:%s", pBuf);
    pEntity->close();
}

acadSSFree(en);
}
Allocates new AcMapPropertyAlterationDefinition object.

virtual AcMap::EErrCode
CreatePropertyAlterationDefinition(
    AcMapPropertyAlterationDefinition*& pAlteration) const;

Returns AcMap::kOk in case of success, error code otherwise.

pAlteration Output new AcMapPropertyAlteration object. It is application’s responsibility to erase it.

This object represents a stand alone property alteration. An application is responsible to delete the object allocated via this method.
Allocates new AcMapPropertyAlteration object.

virtual AcMap::EErrCode CreatePropertyAlterationDefinition(
    AcMapPropertyAlteration*& pAlteration,
    AcMap::EAlterationType kType) const;

Returns AcMap::kOk in case of success, error code otherwise.

**pAlteration**  Output new AcMapPropertyAlteration object. It is application’s responsibility to erase it.

**kType**  Input the type of object allocated by this method, where

- AcMap::kAlterationTextEntity = AcMapTextAlteration object
- AcMap::kAlterationHatch = AcMapHatchAlteration object
- AcMap::kAlterationAnnotate = AcMapAnnotAlteration object
- Other kType arguments = AcMapPropertyAlteration object.

This object represents a stand alone property alteration. An application is responsible for removing the object allocated via this method calling Erase().

**NOTE**  AcMap::kAlterationText does not get you an AcMapTextAlteration object! The only correct kType for AcMapTextAlteration objects is AcMap::kAlterationTextEntity. When this method creates AcMapTextAlteration or AcMapHatchAlteration objects, it uses default values for their parameters.

Unlike similar methods, this method always allocates a new pObj object, which the application is responsible for deleting.
AcMapAttachedDrawing:: ApplyThisQuery Function

AcMapAttachedDrawing Class

Executes the specified query against the attached drawing to select objects within the drawing.

virtual AcMap::EErrCode ApplyThisQuery(
    AcDbObjectIdArray& aObjectSelection,
    AcMapQuery* pQuery) = 0;

Returns AcMap::kOk in case of success, error code otherwise.

aObjectSelection: Output IDs of the objects in the drawing that match the query. These IDs have valid values until the drawing has a heavy lock in place.

pQuery: Input query definition.
**AcMapExpression:: TopologyExecute Function**

**AcMapExpression Class**

Evaluates the expression against the specified topology element.

```
virtual AcMap::EErrCode
TopologyExecute(
    AcMapValue& tValue,
    const char* pcTopologyName,
    long topoElementId,
    Adesk::Boolean fromCurDwg) = 0;
```

Returns nothing.

- **tValue**  
  Output expression value.

- **pcTopologyName**  
  Input topology name of an object to modify.

- **topoElementId**  
  Input topology element ID of an entity to modify.

- **fromCurDwg**  
  Input true if expression has to be executed against a topology in the current drawing or false if against a topology in a source drawing.

Topology must be loaded. If the Val type is not initialized, the type of the value is set automatically.
AcMapQuery:: Execute Function 1 of 3

**AcMapQuery Class**

Executes the query against the given drawing set.

```cpp
virtual AcMap::EErrCode
Execute(
    AcDbObjectIdArray& resultIds, 
    const AcMapDrawingSet* kpDwgSet) = 0;
```

Returns AcMap::kOk in case of success, AcMap::kErrUsrBreak if the user cancels the process, or another error code.

- **resultIds**: Output selection of entities in the project drawing that meet the query conditions.
- **kpDwgSet**: Input the drawing set pointer.

Objects that meet the query conditions are cloned from the source drawings in the drawing set to their associated project drawing. The result is a selection of objects in the project drawing. If an object has already been queried, it is not cloned. It is just added to the selection.
Executes the query against the drawing set of the given project.

```cpp
virtual AcMap::EErrCode Execute(
   AcDbObjectIdArray& resultIds,
   const AcMapProject* kpProject) = 0;
```

Returns AcMap::kOk in case of success, AcMap::kErrUsrBreak if the user cancels the process, or another error code.

- **resultIds**: Output selection of entities in the project drawing that meet the query conditions.
- **kpProject**: Input the project pointer.

The result is a selection of objects in the project drawing.
AcMapQuery:: Execute Function 3 of 3

AcMapQuery Class

Executes query against the given topology.

virtual AcMap::EErrCode
Execute(
    AcDbObjectIdArray& resultIds,
    AcArray<AcDbObjectIdArray*>& resultEntityIdsPerTopoElement,
    AcArray<long>& resultTopoElementIds,
    const char* pszTopoName) = 0;

Returns AcMap::kOk in case of success, AcMap::kErrUsrBreak if the user cancels the process, or another error code.

**resultIds**  
Output selection of entities in the current drawing that meet the query conditions.

**resultEntityIdsPerTopoElement**  
Output array of *AcDbObjectIdArrays for each member of resultTopoElementIds. These are parallel arrays. The AcDbObjectIdArray pointers are allocated by this object and will be cleaned up in the destructor.

**resultTopoElementIds**  
Output selection of topology elements in the current drawing that meet the query conditions.

**pszTopoName**  
Input the topology name. The topology has to be loaded.

The result is a selection of objects in the current drawing. If an object has already been queried, it is not cloned. It is just added to the selection.
AcMapQuery:: FileIn Function
AcMapQuery Class

Restores the query definition from the result buffer.

virtual AcMap::EErrCode
FileIn(
    const struct resbuf* pResBuffer) = 0;

Returns AcMap::kOk in case of success, appropriate error code otherwise.

pResBuffer Input result buffer.
Stores the query definition in a result buffer.

`virtual AcMap::EErrCode FileOut(
    struct resbuf*& pResBuffer) const = 0;`

Returns `AcMap::kOk` in case of success, appropriate error code otherwise.

`pResBuffer` Output result buffer containing the query definition. Your application is responsible for deleting the result buffer. Use `acutRelRb()`.
AcMapQuery:: GetStringToDisplay Function

Retrieves the string representation of the query.

```
virtual AcMap::EErrCode
GetStringToDisplay(
    char*& pszString) const = 0;
```

Returns AcMap::kOk in case of success, appropriate error code otherwise.

`pszString` Output query string. Your application is responsible for releasing the memory allocated for it.
AcMapQuery:: GetTopologyName Function

AcMapQuery Class

Retrieves topology name from a query definition.

virtual AcMap::EErrCode
GetTopologyName(
    const char*& kpszTopoName) const = 0;

Returns AcMap::kOk in case of success, AcMap::kErrObjectNotFound if query is not a topology query, or another error code.

kpszTopoName Output the topology name if any has been assigned. Otherwise NULL.
AcMapQuery:: IsDefined Function

Checks if any query conditions have been defined.

virtual bool
IsDefined();

Returns true if the query has at least one condition.
Executes the query against the given topology.

```c++
virtual AcMap::EErrCode Select(
    AcArray<long>& resultTopoElementIds,
    const char* pszTopoName) = 0;
```

Returns AcMap::kOk in case of success, AcMap::kErrUsrBreak if the user cancels the process, or another error code.

- **resultTopoElementIds**: Output selection of topology elements that meet the query conditions: polygons, links, or nodes, depending on the topology type.
- **pszTopoName**: Input the topology name. The topology has to be loaded.

This function selects topological elements only. There is no cloning.
AcMapQuery:: SetTopologyName Function

Sets topology name for a query definition.

```cpp
template virtual AcMap::EErrCode AcMapQuery::SetTopologyName(const char* kpszTopoName) = 0;
```

Returns AcMap::kOk in case of success, appropriate error code otherwise.

**kpszTopoName**

Input the topology name.

The specified topology has to exist.
Applies property alteration to the entity specified by the ID.

```cpp
virtual AcMap::EErrCode
Apply(
    const AcDbObjectId& Id) = 0;
```

Returns AcMap::kOk in case of success, error code otherwise.

`Id` Input object Id of an entity to modify.
Applies property alteration to the entity specified by the ID.

virtual AcMap::EErrCode Apply(
    const char* topologyName,
    long topoElementId) = 0;

Returns AcMap::kOk in case of success, error code otherwise.

topologyName       Input name of the topology containing the entity to modify.
topoElementId      Input topology element ID of the entity to modify.
AcMapPropertyAlterationDefinition:: Erase Function

AcMapPropertyAlterationDefinition Class

Destroys the property alteration data.

    virtual AcMap::EErrCode
    Erase();

Returns AcMap::kOk in case of success, error code otherwise.

An application is still responsible for calling delete for the erased object to free the memory.
AcMapPropertyAlterationDefinition:: ObjectId Function

Accesses the ID of the associated internal object.

virtual AcMapId
ObjectId();

Returns the ID.
Applies property alteration to the entity specified by the ID.

```cpp
virtual AcMap::EErrorCode
Apply(
    AcDbObjectIdArray& createdObjects,
    const AcDbObjectId& Id) = 0;
```

Returns AcMap::kOk in case of success, error code otherwise.

- **createdObjects**: Output IDs of the objects created. No objects get created in the case of a simple entity alteration.
- **Id**: Input object ID of the entity to modify.
Applies property alteration to the entity specified by the ID.

```
virtual AcMap::EErrCode
Apply(
    AcDbObjectIdArray& createdObjects,
    const char* topologyName,
    long topoElementId) = 0;
```

Returns AcMap::kOk in case of success, error code otherwise.

- `createdObjects`: Output IDs of the objects created. No objects get created in the case of a simple entity alteration.
- `topologyName`: Input topology name of the entity to modify.
- `topoElementId`: Input topology element ID of the entity to modify.
AcMapHatchAlteration:: Erase Function

**AcMapHatchAlteration Class**

Destroys the Property Alteration data.

```cpp
virtual AcMap::EErrCode Erase();
```

Returns AcMap::kOk in case of success, error code otherwise.

An application is still responsible for calling delete for the erased object to free the memory.
Gets an array of objects to exclude from the hatch.

```cpp
virtual AcMap::EErrCode
GetExcludedObjects(
    AcDbObjectIdArray& excludedObjects) = 0;
```

Returns AcMap::kOk in case of success, error code otherwise.

`excludedObjects` Output Ids of the objects to be excluded from this hatch.
Sets an array of objects to exclude from the hatch.

```
virtual AcMap::EErrCode
SetExcludedObjects(
    AcDbObjectIdArray& excludedObjects) = 0;
```

Returns AcMap::kOk in case of success, error code otherwise.

`excludedObjects`     Input IDs of the objects to be excluded from this hatch.
AcMapTextAlteration:: Apply Function 1 of 2

AcMapTextAlteration Class

Applies property alteration to the entity specified via the Id.

virtual AcMap::EErrCode
Apply(
    AcDbObjectIdArray& createdObjects,
    const AcDbObjectId& Id) = 0;

Returns AcMap::kOk in case of success, error code otherwise.

createdObjects  Output Ids of the objects created. No objects get created in the case of a simple entity alteration.
Id  Input object Id of an entity to modify.
Applies property alteration to the entity specified via the Id.

```cpp
virtual AcMap::EErrCode Apply(
    AcDbObjectIdArray& createdObjects,
    const char* topologyName,
    long topoElementId) = 0;
```

Returns AcMap::kOk in case of success, error code otherwise.

- **createdObjects**: Output Ids of the objects created. No objects get created in the case of a simple entity alteration.
- **topologyName**: Input topology name of an object to modify.
- **topoElementId**: Input topology element Id of an entity to modify.
Destroys the Property Alteration data.

```cpp
virtual AcMap::EErrCode Erase();
```

Returns AcMap::kOk in case of success, error code otherwise.

An application is still responsible for calling delete for the erased object to free the memory.
Subclassing Custom Reactors

The following sample code illustrates subclassing custom reactors from AcMapOracleReactor classes. Classes and functions are declared, but function definitions, which would be highly implementation specific, are not shown. This is an example of what MyMapOracleReactors.h, one of the include files in the "Connecting to an Oracle Spatial Database" sample, would contain.

Managing reactors (overview) Connecting to an Oracle Spatial Database (sample)
Other related samples

```cpp
#include "AcMapOracleReactor.h"

class MyConnectionReactor : public AcMapOracleConnectionReactor
{
public:
    MyConnectionReactor();
    virtual ~MyConnectionReactor();

    virtual void Connected();
    virtual void Disconnected();
    virtual void BeforeConnect();
    virtual void BeforeDisconnect();

};

class MyExportReactor : public AcMapOracleExportReactor
{
public:
    MyExportReactor();
    virtual ~MyExportReactor();

    virtual void BeforeObjectCached(AcDbEntity *);
    virtual void ObjectCached(AcDbEntity *,unsigned long);
    virtual void ObjectRejected(AcDbEntity *);
    virtual void BeforeObjectsExported(std::vector &);
    virtual void ObjectsExported(std::vector &);

};

class MyImportReactor : public AcMapOracleImportReactor
{
public:
    MyImportReactor();
    virtual ~MyImportReactor();

    virtual void BeforeRecordImport(const ODynaset &);
```
virtual void RecordImported(const ODynaset &, AcDbEntity *);
virtual void RecordRejected(const ODynaset &);

};