Dynamic Data Exchange Library for .NET
**NDde Namespace**

This namespace contains classes for using Dynamic Data Exchange (DDE) in .NET.

**Namespace hierarchy**

**Classes**

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>DdeEventArgs</code></td>
<td>This is the base class for all NDde event argument classes.</td>
</tr>
<tr>
<td><code>DdeException</code></td>
<td>This is thrown when a DDE exception occurs.</td>
</tr>
</tbody>
</table>

**Send comments on this topic.**

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Dynamic Data Exchange Library for .NET
**DdeEventArgs Class**

This is the base class for all NDde event argument classes.

For a list of all members of this type, see [DdeEventArgs Members](#).

```csharp
System.Object EventArgs DdeEventArgs
    DdeRegistrationEventArgs
    DdeActivityEventArgs
    DdeAdviseEventArgs
    DdeDisconnectedEventArgs
```

**public class DdeEventArgs : EventArgs**

**Thread Safety**

Public static *(Shared in Visual Basic)* members of this type are safe for multithreaded operations. Instance members are **not** guaranteed to be thread-safe.

**Requirements**

- **Namespace:** NDde
- **Assembly:** NDde (in NDde.dll)

**See Also**

[DdeEventArgs Members](#) | [NDde Namespace](#)

---

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
## DdeEventArgs Members

### DdeEventArgs overview

### Public Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <strong>Object</strong> is equal to the current <strong>Object</strong>.</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. <strong>GetHashCode</strong> is suitable for use in hashing algorithms and data structures like a hash table.</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <strong>Type</strong> of the current instance.</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>This returns a string containing the current values of all properties.</td>
</tr>
</tbody>
</table>

### Protected Instance Constructors

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>DdeEventArgs Constructor</strong></td>
<td>Initializes a new instance of the <strong>DdeEventArgs</strong> class.</td>
</tr>
</tbody>
</table>

### Protected Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <strong>Object</strong> to attempt to free resources and perform other cleanup operations before the <strong>Object</strong> is reclaimed by garbage collection.</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <strong>Object</strong>.</td>
</tr>
</tbody>
</table>

### See Also

[**DdeEventArgs Class**](#) | [**NDde Namespace**](#)
Dynamic Data Exchange Library for .NET
DdeEventArgs Constructor

Initializes a new instance of the **DdeEventArgs** class.

```csharp
protected DdeEventArgs();
```

See Also

[DdeEventArgs Class](#) | [NDde Namespace](#)

---

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeEventArgs Methods

The methods of the **DdeEventArgs** class are listed below. For a complete list of **DdeEventArgs** class members, see the **DdeEventArgs Members** topic.

Public Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <strong>Object</strong> is equal to the current <strong>Object</strong>.</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. <strong>GetHashCode</strong> is suitable for use in hashing algorithms and data structures like a hash table.</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <strong>Type</strong> of the current instance.</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>This returns a string containing the current values of all properties.</td>
</tr>
</tbody>
</table>

Protected Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <strong>Object</strong> to attempt to free resources and perform other cleanup operations before the <strong>Object</strong> is reclaimed by garbage collection.</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <strong>Object</strong>.</td>
</tr>
</tbody>
</table>

See Also

[DdeEventArgs Class] | [NDde Namespace]  

---

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeEventArgs.ToString Method

This returns a string containing the current values of all properties.

```csharp
public override string ToString();
```

Return Value

A string containing the current values of all properties.

See Also

[DdeEventArgs Class] [NDde Namespace]

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeException Class

This is thrown when a DDE exception occurs.

For a list of all members of this type, see DdeException Members.

System.Object Exception
DdeException

public class DdeException : Exception

Thread Safety

Public static (Shared in Visual Basic) members of this type are safe for multithreaded operations. Instance members are not guaranteed to be thread-safe.

Requirements

Namespace: NDde
Assembly: NDde (in NDde.dll)

See Also

DdeException Members | NDde Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
# DdeException Members

## DdeException overview

### Public Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="Code" alt="Code" /></td>
<td>This gets an error code returned by the DDEML.</td>
</tr>
<tr>
<td><img src="Data" alt="Data" /></td>
<td>(inherited from Exception) Gets a collection of key/value pairs that provide additional, user-defined information about the exception.</td>
</tr>
<tr>
<td><img src="HelpLink" alt="HelpLink" /></td>
<td>(inherited from Exception) Gets or sets a link to the help file associated with this exception.</td>
</tr>
<tr>
<td><img src="InnerException" alt="InnerException" /></td>
<td>(inherited from Exception) Gets the Exception instance that caused the current exception.</td>
</tr>
<tr>
<td><img src="Message" alt="Message" /></td>
<td>(inherited from Exception) Gets a message that describes the current exception.</td>
</tr>
<tr>
<td><img src="Source" alt="Source" /></td>
<td>(inherited from Exception) Gets or sets the name of the application or the object that causes the error.</td>
</tr>
<tr>
<td><img src="StackTrace" alt="StackTrace" /></td>
<td>(inherited from Exception) Gets a string representation of the frames on the call stack at the time the current exception was thrown.</td>
</tr>
<tr>
<td><img src="TargetSite" alt="TargetSite" /></td>
<td>(inherited from Exception) Gets the method that throws the current exception.</td>
</tr>
</tbody>
</table>

### Public Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="Equals" alt="Equals" /></td>
<td>(inherited from Object) Determines whether the specified Object is equal to the current Object.</td>
</tr>
<tr>
<td><img src="GetBaseException" alt="GetBaseException" /></td>
<td>(inherited) When overridden in a derived</td>
</tr>
</tbody>
</table>
from Exception) class, returns the Exception that is the root cause of one or more subsequent exceptions.

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>GetHashCode</strong> (inherited from Object)</td>
<td>Serves as a hash function for a particular type. GetHashCode is suitable for use in hashing algorithms and data structures like a hash table.</td>
</tr>
<tr>
<td><strong>GetObjectData</strong></td>
<td></td>
</tr>
<tr>
<td><strong>GetType</strong> (inherited from Exception)</td>
<td>Gets the runtime type of the current instance.</td>
</tr>
<tr>
<td><strong>ToString</strong> (inherited from Exception)</td>
<td>Creates and returns a string representation of the current exception.</td>
</tr>
</tbody>
</table>

**Protected Instance Constructors**

- DdeException Constructor

**Protected Instance Properties**

- HResult (inherited from Exception) Gets or sets HRESULT, a coded numerical value that is assigned to a specific exception.

**Protected Instance Methods**

- Finalize (inherited from Object) Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection.
- MemberwiseClone (inherited from Object) Creates a shallow copy of the current Object.

**See Also**

DdeException Class | NDde Namespace
Dynamic Data Exchange Library for .NET
DdeException Constructor

```csharp
protected DdeException(
    SerializationInfo info,
    StreamingContext context
);
```

Parameters

- `info`
- `context`

See Also

- [DdeException Class](#) | [NDde Namespace](#)

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
The properties of the **DdeException** class are listed below. For a complete list of **DdeException** class members, see the [DdeException Members](#) topic.

### Public Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Code</strong></td>
<td>This gets an error code returned by the DDEML.</td>
</tr>
<tr>
<td><strong>Data</strong> (inherited from Exception)</td>
<td>Gets a collection of key/value pairs that provide additional, user-defined information about the exception.</td>
</tr>
<tr>
<td><strong>HelpLink</strong> (inherited from Exception)</td>
<td>Gets or sets a link to the help file associated with this exception.</td>
</tr>
<tr>
<td><strong>InnerException</strong> (inherited from Exception)</td>
<td>Gets the <strong>Exception</strong> instance that caused the current exception.</td>
</tr>
<tr>
<td><strong>Message</strong> (inherited from Exception)</td>
<td>Gets a message that describes the current exception.</td>
</tr>
<tr>
<td><strong>Source</strong> (inherited from Exception)</td>
<td>Gets or sets the name of the application or the object that causes the error.</td>
</tr>
<tr>
<td><strong>StackTrace</strong> (inherited from Exception)</td>
<td>Gets a string representation of the frames on the call stack at the time the current exception was thrown.</td>
</tr>
<tr>
<td><strong>TargetSite</strong> (inherited from Exception)</td>
<td>Gets the method that throws the current exception.</td>
</tr>
</tbody>
</table>

### Protected Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>HResult</strong> (inherited from Exception)</td>
<td>Gets or sets HRESULT, a coded numerical value that is assigned</td>
</tr>
</tbody>
</table>
See Also

DdeException Class | NDde Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
This gets an error code returned by the DDEML.

```
public Int32 Code {get;}
```

Remarks

The value is zero if the exception was not thrown because of the DDEML.

- `0x0000` - DMLERR_NO_DMLERROR
- `0x4000` - DMLERR_ADVACKTIMEOUT
- `0x4001` - DMLERR_BUSY
- `0x4002` - DMLERR_DATAACKTIMEOUT
- `0x4003` - DMLERR_DLL_NOT_INITIALIZED
- `0x4004` - DMLERR_DLL_USAGE
- `0x4005` - DMLERR_EXECACKTIMEOUT
- `0x4006` - DMLERR_INVALIDPARAMETER
- `0x4007` - DMLERR_LOW_MEMORY
- `0x4008` - DMLERR_MEMORY_DMLERROR
- `0x4009` - DMLERR_NOTPROCESSED
- `0x400A` - DMLERR_NO_CONV_ESTABLISHED
- `0x400B` - DMLERR_POKEACKTIMEOUT
- `0x400C` - DMLERR_POSTMSG_FAILED
- `0x400D` - DMLERR_REENTRANCY
- `0x400E` - DMLERR_SERVER_DIED
- `0x400F` - DMLERR_SYS_DMLERROR
- `0x4010` - DMLERR_UNADVACKTIMEOUT
- `0x4011` - DMLERR_UNFOUND_QUEUE_ID

See Also
Dynamic Data Exchange Library for .NET
DdeException Methods

The methods of the DdeException class are listed below. For a complete list of DdeException class members, see the DdeException Members topic.

### Public Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong> (inherited from Object)</td>
<td>Determines whether the specified Object is equal to the current Object.</td>
</tr>
<tr>
<td><strong>GetBaseException</strong> (inherited from Exception)</td>
<td>When overridden in a derived class, returns the Exception that is the root cause of one or more subsequent exceptions.</td>
</tr>
<tr>
<td><strong>GetHashCode</strong> (inherited from Object)</td>
<td>Serves as a hash function for a particular type. GetHashCode is suitable for use in hashing algorithms and data structures like a hash table.</td>
</tr>
<tr>
<td><strong>GetObjectData</strong></td>
<td></td>
</tr>
<tr>
<td><strong>GetType</strong> (inherited from Exception)</td>
<td>Gets the runtime type of the current instance.</td>
</tr>
<tr>
<td><strong>ToString</strong> (inherited from Exception)</td>
<td>Creates and returns a string representation of the current exception.</td>
</tr>
</tbody>
</table>

### Protected Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Finalize</strong> (inherited from Object)</td>
<td>Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection.</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong> (inherited from Object)</td>
<td>Creates a shallow copy of the current Object.</td>
</tr>
</tbody>
</table>
Dynamic Data Exchange Library for .NET
public override void GetObjectData(
    SerializationInfo info,
    StreamingContext context);

Parameters

    info
    context

Implements

    ISerializable.

See Also

    DdeException Class | NDde Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
**NDde.Advanced Namespace**

This namespace contains classes for using advanced features of the library.

**Namespace hierarchy**

**Classes**

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>DdeContext</code></td>
<td>This provides an execution context for <code>DdeClient</code> and <code>DdeServer</code>.</td>
</tr>
<tr>
<td><code>DdeMessageLoop</code></td>
<td>This is a synchronizing object that can run a message loop on any thread.</td>
</tr>
<tr>
<td><code>DdeRegistrationEventArgs</code></td>
<td>This contains information about the <code>Register</code> and <code>Unregister</code> events.</td>
</tr>
<tr>
<td><code>DdeTransaction</code></td>
<td>This contains the parameters of the DDEML callback function.</td>
</tr>
</tbody>
</table>

**Interfaces**

<table>
<thead>
<tr>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>IDdeTransactionFilter</code></td>
<td>This defines a transaction filter.</td>
</tr>
</tbody>
</table>

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Dynamic Data Exchange Library for .NET
DdeContext Class

This provides an execution context for DdeClient and DdeServer.

For a list of all members of this type, see DdeContext Members.

System.Object DdeContext

```csharp
public class DdeContext : IDisposable, ISynchronizeInvoke
```

Thread Safety

This type is safe for multithreaded operations.

Remarks

This class provides a context for DDE activity. All DdeClient and DdeServer objects must be associated with an instance of this class. If one is not specified in their constructors then a default instance of this class is used. This class must be initialized before it can begin sending and receiving DDE messages. This happens automatically upon its first use by a DdeClient or DdeServer. An application can call Initialize to make the initialization process occur immediately. This is useful when a calling application expects this class to raise the Register and Unregister events or invoke the ITransactionFilter.PreFilterTransaction method before being used by a DdeClient or DdeServer.

Since forms and controls implement ISynchronizeInvoke they can be used as the synchronizing object for this class. When an instance of this class is created to use a form or control as the synchronizing object it will use the UI thread for execution. This is the preferred way of creating an instance of this class when used in a windows application since it avoids multithreaded synchronization issues and cross thread marshaling. When an instance of this class is created without specifying a synchronizing object it will create and manage its own thread for execution. This is convenient if you wish to use this library in a console or service application, but with the
added cost of cross thread marshaling and the potential for deadlocking application threads.

Events are invoked on the thread hosting the `DdeContext`. All operations must be marshaled onto the thread hosting the `DdeContext`. Method calls will block until that thread becomes available. An exception will be generated if the thread does not become available in a timely manner.

**Example**

The following example demonstrates how to instantiate a `DdeContext` in a console application.

```csharp
using System;
using NDde.Advanced;

public class Example
{
    public static void Main()
    {
        // Create a context that uses a dedicated thread for DDE message pumping.
        DdeContext context = new DdeContext();
    }
}
```

```visualbasic
Imports NDde.Advanced

Public Class Example
    Public Shared Sub Main()
        ' Create a context that uses a dedicated thread for DDE message pumping.
        Dim context As DdeContext = New DdeContext()
    End Sub
End Class
```

The following example demonstrates how to instantiate a `DdeContext` in a windows application.
using System;
using NDde.Advanced;

public class Example : Form
{
    // Standard Form code omitted for brevity.
    private DdeContext context = null;

    private void Form1_Load(object sender, System.EventArgs e)
    {
        // Create a context that uses the UI thread
        context = new DdeContext(this);
    }
}

[Visual Basic]
Imports NDde.Advanced

Public Class Example
    Inherits Form
    
    Private context As DdeContext = Nothing
    
    Private Sub Form1_Load(ByVal sender As Object, ByVal e As System.EventArgs)
        ' Create a context that uses the UI thread
        context = New DdeContext(Me)
    End Sub
End Class

Requirements

Namespace: NDde.Advanced

Assembly: NDde (in NDde.dll)

See Also
Dynamic Data Exchange Library for .NET
## DdeContext Members

### DdeContext overview

### Public Instance Constructors

<table>
<thead>
<tr>
<th>Constructor</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DdeContext</td>
<td>Overloaded.</td>
</tr>
</tbody>
</table>

### Public Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Encoding</td>
<td>This gets or sets the default encoding that is used.</td>
</tr>
<tr>
<td>Instanceld</td>
<td>This gets the DDEML instance identifier.</td>
</tr>
<tr>
<td>InvokeRequired</td>
<td>This gets a bool indicating whether the caller must use Invoke.</td>
</tr>
<tr>
<td>IsInitialized</td>
<td>This gets a bool indicating whether the context is initialized.</td>
</tr>
</tbody>
</table>

### Public Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AddTransactionFilter</td>
<td>This adds a transaction filter to monitor DDE transactions.</td>
</tr>
<tr>
<td>BeginInvoke</td>
<td>This begins an asynchronous operation to execute a delegate on the thread hosting this object.</td>
</tr>
<tr>
<td>Dispose</td>
<td>This releases all resources held by this instance.</td>
</tr>
<tr>
<td>EndInvoke</td>
<td>This returns the object that the delegate returned in the operation.</td>
</tr>
<tr>
<td>Equals</td>
<td>Determines whether the specified Object is equal to the current Object.</td>
</tr>
<tr>
<td><strong>GetHashCode</strong> (inherited from Object)</td>
<td>Serves as a hash function for a particular type. <strong>GetHashCode</strong> is suitable for use in hashing algorithms and data structures like a hash table.</td>
</tr>
<tr>
<td>----------------------------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>GetType</strong> (inherited from Object)</td>
<td>Gets the <strong>Type</strong> of the current instance.</td>
</tr>
<tr>
<td><strong>Initialize</strong></td>
<td>This initializes the context.</td>
</tr>
<tr>
<td><strong>Invoke</strong></td>
<td>This executes a delegate on the thread hosting this object.</td>
</tr>
<tr>
<td><strong>RemoveTransactionFilter</strong></td>
<td>This removes a transaction filter and stops it from monitoring DDE transactions.</td>
</tr>
<tr>
<td><strong>ToString</strong> (inherited from Object)</td>
<td>Returns a <strong>String</strong> that represents the current <strong>Object</strong>.</td>
</tr>
</tbody>
</table>

**Public Instance Events**

| **Register**                           | This is raised when a service name has been registered by a server using the DDEML.                                                                                                               |
| **Unregister**                          | This is raised when a service name has been unregistered by a server using the DDEML.                                                                                                               |

**See Also**

[DdeContext Class] | [NDde.Advanced Namespace]

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeContext Constructor

Overload List

This initializes a new instance of the DdeContext class that uses a dedicated thread for execution.

    public DdeContext();

This initializes a new instance of the DdeContext class that uses the specified synchronizing object for execution.

    public DdeContext(ISynchronizeInvoke);

See Also

DdeContext Class | NDde.Advanced Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
This initializes a new instance of the DdeContext class that uses a dedicated thread for execution.

```csharp
public DdeContext();
```

Remarks

This constructor is used when you want the context to create and manage its own thread for DDE message pumping.

See Also

DdeContext Class | NDde.Advanced Namespace | DdeContext Constructor Overload List

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
**DdeContext Constructor (ISynchronizeInvoke)**

This initializes a new instance of the `DdeContext` class that uses the specified synchronizing object for execution.

```csharp
public DdeContext(
    ISynchronizeInvoke synchronizingObject
);
```

**Parameters**

`synchronizingObject`  
The synchronizing object to use for execution.

**Remarks**

This constructor is used when you want the context to use the specified synchronizing object for DDE message pumping. Since forms and controls implement `ISynchronizeInvoke` they can be used as the synchronizing object. In that case the windows application UI thread that is hosting the form or control is used.

**Exceptions**

<table>
<thead>
<tr>
<th>Exception Type</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ArgumentNullException</code></td>
<td>This is thrown when synchronizer is a null reference.</td>
</tr>
</tbody>
</table>

**See Also**

[DdeContext Class](#) | [NDde.Advanced Namespace](#) | [DdeContext Constructor Overload List](#)

---

*Send comments on this topic.*

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeContext Properties

The properties of the **DdeContext** class are listed below. For a complete list of **DdeContext** class members, see the [DdeContext Members](#) topic.

### Public Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Encoding</strong></td>
<td>This gets or sets the default encoding that is used.</td>
</tr>
<tr>
<td><strong>InstanceId</strong></td>
<td>This gets the DDEML instance identifier.</td>
</tr>
<tr>
<td><strong>InvokeRequired</strong></td>
<td>This gets a bool indicating whether the caller must use Invoke.</td>
</tr>
<tr>
<td><strong>IsInitialized</strong></td>
<td>This gets a bool indicating whether the context is initialized.</td>
</tr>
</tbody>
</table>

See Also

[DdeContext Class](#) | [NDde.Advanced Namespace](#)

---

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeContext.Encoding Property

This gets or sets the default encoding that is used.

```csharp
public Encoding Encoding {get; set;}
```

See Also

[DdeContext Class] | [NDde.Advanced Namespace]

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
**DdeContext.InstancesId Property**

This gets the DDEML instance identifier.

```csharp
public Int32 InstancesId {get;}
```

**Remarks**

This can be used in any DDEML function requiring an instance identifier.

**CAUTION** Incorrect usage of the DDEML can cause this library to function incorrectly and can lead to resource leaks.

**See Also**

- [DdeContext Class](#) | [NDde.Advanced Namespace](#)

---

**Send comments on this topic.**

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeContext.InvokeRequired Property

This gets a bool indicating whether the caller must use Invoke.

```csharp
public Boolean InvokeRequired {get;}
```

Implements

- ISynchronizeInvoke.

See Also

- DdeContext Class | NDde.Advanced Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeContext.IsInitialized Property

This gets a bool indicating whether the context is initialized.

```
public Boolean IsInitialized {get;}
```

See Also

DdeContext Class | NDde.Advanced Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
## DdeContext Methods

The methods of the **DdeContext** class are listed below. For a complete list of **DdeContext** class members, see the **DdeContext Members** topic.

### Public Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>AddTransactionFilter</strong></td>
<td>This adds a transaction filter to monitor DDE transactions.</td>
</tr>
<tr>
<td><strong>BeginInvoke</strong></td>
<td>This begins an asynchronous operation to execute a delegate on the thread hosting this object.</td>
</tr>
<tr>
<td><strong>Dispose</strong></td>
<td>This releases all resources held by this instance.</td>
</tr>
<tr>
<td><strong>EndInvoke</strong></td>
<td>This returns the object that the delegate returned in the operation.</td>
</tr>
<tr>
<td><strong>Equals</strong> (inherited from <strong>Object</strong>)</td>
<td>Determines whether the specified <strong>Object</strong> is equal to the current <strong>Object</strong>.</td>
</tr>
<tr>
<td><strong>GetHashCode</strong> (inherited from <strong>Object</strong>)</td>
<td>Serves as a hash function for a particular type. <strong>GetHashCode</strong> is suitable for use in hashing algorithms and data structures like a hash table.</td>
</tr>
<tr>
<td><strong>GetType</strong> (inherited from <strong>Object</strong>)</td>
<td>Gets the <strong>Type</strong> of the current instance.</td>
</tr>
<tr>
<td><strong>Initialize</strong></td>
<td>This initializes the context.</td>
</tr>
<tr>
<td><strong>Invoke</strong></td>
<td>This executes a delegate on the thread hosting this object.</td>
</tr>
<tr>
<td><strong>RemoveTransactionFilter</strong></td>
<td>This removes a transaction filter and stops it from monitoring DDE transactions.</td>
</tr>
<tr>
<td><strong>ToString</strong> (inherited from Object)</td>
<td>Returns a <strong>String</strong> that represents the current <strong>Object</strong>.</td>
</tr>
</tbody>
</table>

**See Also**

- [DdeContext Class](#) | [NDde.Advanced Namespace](#)

*Send comments on this topic.*

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
This adds a transaction filter to monitor DDE transactions.

```csharp
public void AddTransactionFilter(IDdeTransactionFilter filter);
```

**Parameters**

*filter*

The implementation of `ITransactionFilter` that you want to add.

**Remarks**

Transaction filters can be used to intercept the DDEML callback.

**CAUTION** Incorrect usage of the DDEML can cause this library to function incorrectly and can lead to resource leaks.

**Exceptions**

<table>
<thead>
<tr>
<th>Exception Type</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ArgumentNullException</code></td>
<td>This is thrown when filter is a null reference.</td>
</tr>
<tr>
<td><code>InvalidOperationException</code></td>
<td>This is thrown when the filter was already added.</td>
</tr>
</tbody>
</table>

**See Also**

[DdeContext Class] | [NDde.Advanced Namespace]

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeContext.BeginInvoke Method

This begins an asynchronous operation to execute a delegate on the thread hosting this object.

```csharp
public IAsyncResult BeginInvoke(
    Delegate method,
    Object[] args
);
```

Parameters

- **method**
  - The delegate to execute.

- **args**
  - The arguments to pass to the delegate.

Return Value

An **IAsyncResult** object for this operation.

Implements

**ISynchronizeInvoke**.

See Also

**DdeContext Class** | **NDde.Advanced Namespace**

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeContext.Dispose Method

This releases all resources held by this instance.

```csharp
public void Dispose();
```

Implements

IDisposable.

See Also

DdeContext Class | NDde.Advanced Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeContext.EndInvoke Method

This returns the object that the delegate returned in the operation.

```csharp
public object EndInvoke(IAsyncResult asyncResult);
```

Parameters

asyncResult

The IAsyncResult object returned by a call to BeginInvoke.

Return Value

The object returned by the delegate.

Implements

ISynchronizeInvoke.

See Also

DdeContext Class | NDde.Advanced Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeContext.Initialize Method

This initializes the context.

```csharp
public void Initialize();
```

Remarks

This class must be initialized before it can begin sending and receiving DDE messages. This happens automatically upon its first use by a `DdeClient` or `DdeServer`. An application can call `Initialize` to make the initialization process occur immediately. This is useful when a calling application expects this class to raise the `Register` and `Unregister` events or invoke the `ITransactionFilter.PreFilterTransaction` method before being used by a `DdeClient` or `DdeServer`.

If you attempt to use a synchronizer that is not hosted on a thread running a windows message loop an exception will be thrown.

Explicitly calling this method will allow added `ITransactionFilter` objects to begin intercepting the DDEML callback function.

Exceptions

<table>
<thead>
<tr>
<th>Exception Type</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>InvalidOperationException</code></td>
<td>This is thrown when the context is already initialized.</td>
</tr>
<tr>
<td><code>DdeException</code></td>
<td>This is thrown when the context could not be initialized.</td>
</tr>
</tbody>
</table>

See Also

[DdeContext Class] | [NDde.Advanced Namespace]

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeContext.Invoke Method

This executes a delegate on the thread hosting this object.

```csharp
public object Invoke(
    Delegate method,
    Object[] args
);
```

Parameters

* method
  The delegate to execute.

* args
  The arguments to pass to the delegate.

Return Value

The object returned by the delegate.

Implements

ISynchronizeInvoke.

See Also

DdeContext Class | NDde.Advanced Namespace

---

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeContext.RemoveTransactionFilter Method

This removes a transaction filter and stops it from monitoring DDE transactions.

```csharp
public void RemoveTransactionFilter(
    IDdeTransactionFilter filter
);
```

Parameters

filter
The implementation of ITransactionFilter that you want to remove.

Remarks

Transaction filters can be used to intercept the DDEML callback.

**CAUTION** Incorrect usage of the DDEML can cause this library to function incorrectly and can lead to resource leaks.

Exceptions

<table>
<thead>
<tr>
<th>Exception Type</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ArgumentNullException</td>
<td>This is thrown when filter is a null reference.</td>
</tr>
<tr>
<td>InvalidOperationException</td>
<td>This is thrown when the filter was not previously added.</td>
</tr>
</tbody>
</table>

See Also

DdeContext Class | NDde.Advanced Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
DdeContext Events

The events of the DdeContext class are listed below. For a complete list of DdeContext class members, see the DdeContext Members topic.

Public Instance Events

<table>
<thead>
<tr>
<th>Event</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Register</td>
<td>This is raised when a service name has been registered by a server using the DDEML.</td>
</tr>
<tr>
<td>Unregister</td>
<td>This is raised when a service name has been unregistered by a server using the DDEML.</td>
</tr>
</tbody>
</table>

See Also

DdeContext Class | NDde.Advanced Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeContext.EventHandler<TEventArgs> Event

This is raised when a service name has been registered by a server using the DDEML.

public event EventHandler<TEventArgs> Register;

Remarks
This event will not be raised by servers that do not use the DDEML.

See Also
DdeContext Class | NDde.Advanced Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
This is raised when a service name has been unregistered by a server using the DDEML.

```csharp
public event EventHandler<TEventArgs> Unregister;
```

Remarks

This event will not be raised by servers that do not use the DDEML.

See Also

[DdeContext Class] | [NDde.Advanced Namespace]

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
This is a synchronizing object that can run a message loop on any thread.

For a list of all members of this type, see DdeMessageLoop Members.

System.Object  DdeMessageLoop

**Thread Safety**

Public static (Shared in Visual Basic) members of this type are safe for multithreaded operations. Instance members are not guaranteed to be thread-safe.

**Requirements**

Namespace: NDde.Advanced

Assembly: NDde (in NDde.dll)

**See Also**

DdeMessageLoop Members | NDde.Advanced Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
**DdeMessageLoop Members**

**DdeMessageLoop overview**

**Public Instance Constructors**

<table>
<thead>
<tr>
<th>Constructor</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>DdeMessageLoop Constructor</strong></td>
<td>This initializes a new instance of the DdeMessageLoop class.</td>
</tr>
</tbody>
</table>

**Public Instance Methods**

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Dispose</strong></td>
<td>This releases all resources held by this instance.</td>
</tr>
<tr>
<td><strong>Equals</strong> (inherited from Object)</td>
<td>Determines whether the specified Object is equal to the current Object.</td>
</tr>
<tr>
<td><strong>GetHashCode</strong> (inherited from Object)</td>
<td>Serves as a hash function for a particular type. GetHashCode is suitable for use in hashing algorithms and data structures like a hash table.</td>
</tr>
<tr>
<td><strong>GetType</strong> (inherited from Object)</td>
<td>Gets the Type of the current instance.</td>
</tr>
<tr>
<td><strong>Run</strong></td>
<td>Overloaded. This starts a message loop on the current thread.</td>
</tr>
<tr>
<td><strong>ToString</strong> (inherited from Object)</td>
<td>Returns a String that represents the current Object.</td>
</tr>
</tbody>
</table>

**See Also**

DdeMessageLoop Class | NDde.Advanced Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeMessageLoop Constructor

This initializes a new instance of the DdeMessageLoop class.

```plaintext
public DdeMessageLoop();
```

See Also

DdeMessageLoop Class | NDde.Advanced Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeMessageLoop Methods

The methods of the **DdeMessageLoop** class are listed below. For a complete list of **DdeMessageLoop** class members, see the **DdeMessageLoop Members** topic.

### Public Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Dispose</strong></td>
<td>This releases all resources held by this instance.</td>
</tr>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <strong>Object</strong> is equal to the current <strong>Object</strong>.</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. <strong>GetHashCode</strong> is suitable for use in hashing algorithms and data structures like a hash table.</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <strong>Type</strong> of the current instance.</td>
</tr>
<tr>
<td><strong>Run</strong></td>
<td>Overloaded. This starts a message loop on the current thread.</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <strong>String</strong> that represents the current <strong>Object</strong>.</td>
</tr>
</tbody>
</table>

See Also

[DdeMessageLoop Class](#) | [NDde.Advanced Namespace](#)

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
**DdeMessageLoop.Dispose Method**

This releases all resources held by this instance.

```csharp
public void Dispose();
```

**Implements**

`IDisposable`.

**See Also**

[DdeMessageLoop Class](#) | [NDde.Advanced Namespace](#)

---

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeMessageLoop.Run Method

This starts a message loop on the current thread.

Overload List

This starts a message loop on the current thread.

public void Run();

This starts a message loop on the current thread and shows the specified form.

public void Run(Form);

See Also

DdeMessageLoop Class | NDde.Advanced Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
This starts a message loop on the current thread.

```csharp
public void Run();
```

See Also

[DdeMessageLoop Class] | [NDde.Advanced Namespace] | [DdeMessageLoop.Run Overload List]

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeMessageLoop.Run Method (Form)

This starts a message loop on the current thread and shows the specified form.

```csharp
public void Run(
    Form form
);
```

Parameters

form

The Form to display.

See Also

DdeMessageLoop Class | NDde.Advanced Namespace | DdeMessageLoop.Run Overload List

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
This contains information about the Register and Unregister events.

For a list of all members of this type, see DdeRegistrationEventArgs Members.

System.Object EventArgs DdeEventArgs DdeRegistrationEventArgs

public class DdeRegistrationEventArgs : 
    EventArgs

Thread Safety

Public static (Shared in Visual Basic) members of this type are safe for multithreaded operations. Instance members are not guaranteed to be thread-safe.

Requirements

Namespace: NDde.Advanced
Assembly: NDde (in NDde.dll)

See Also

DdeRegistrationEventArgs Members | NDde.Advanced Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
## DdeRegistrationEventArgs Members

### DdeRegistrationEventArgs overview

#### Public Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Service</strong></td>
<td>This gets the service name associated with this event.</td>
</tr>
</tbody>
</table>

#### Public Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <strong>Object</strong> is equal to the current <strong>Object</strong>.</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. <strong>GetHashCode</strong> is suitable for use in hashing algorithms and data structures like a hash table.</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <strong>Type</strong> of the current instance.</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>This returns a string containing the current values of all properties.</td>
</tr>
</tbody>
</table>

### See Also

[DdeRegistrationEventArgs Class] | [NDde.Advanced Namespace]

---

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
The properties of the `DdeRegistrationEventArgs` class are listed below. For a complete list of `DdeRegistrationEventArgs` class members, see the `DdeRegistrationEventArgs Members` topic.

### Public Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>Service</code></td>
<td>This gets the service name associated with this event.</td>
</tr>
</tbody>
</table>

See Also

- [DdeRegistrationEventArgs Class](#) | [NDde.Advanced Namespace](#)

---

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeRegistrationEventArgs.Service Property

This gets the service name associated with this event.

```java
public String Service {get;}
```

See Also

DdeRegistrationEventArgs Class | NDde.Advanced Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeTransaction Class

This contains the parameters of the DDEML callback function. For a list of all members of this type, see DdeTransaction Members.

System.Object  DdeTransaction

public class DdeTransaction

Thread Safety

Public static (Shared in Visual Basic) members of this type are safe for multithreaded operations. Instance members are not guaranteed to be thread-safe.

Remarks

The dwRet property contains the value returned by the DDEML callback function and is the only member that can be modified. See the MSDN documentation for more information about the members of this class.

CAUTION Incorrect usage of the DDEML can cause this library to function incorrectly and can lead to resource leaks.

Requirements

Namespace: NDde.Advanced
Assembly: NDde (in NDde.dll)

See Also

DdeTransaction Members  |  NDde.Advanced Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
# DdeTransaction Members

## DdeTransaction overview

### Public Instance Properties

<table>
<thead>
<tr>
<th>Member</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dwData1</td>
<td>See the MSDN documentation for information about this member.</td>
</tr>
<tr>
<td>dwData2</td>
<td>See the MSDN documentation for information about this member.</td>
</tr>
<tr>
<td>dwRet</td>
<td>This gets the return value of the DDEML callback function. See the MSDN documentation for information about this member.</td>
</tr>
<tr>
<td>hConv</td>
<td>See the MSDN documentation for information about this member.</td>
</tr>
<tr>
<td>hData</td>
<td>See the MSDN documentation for information about this member.</td>
</tr>
<tr>
<td>hsz1</td>
<td>See the MSDN documentation for information about this member.</td>
</tr>
<tr>
<td>hsz2</td>
<td>See the MSDN documentation for information about this member.</td>
</tr>
<tr>
<td>uFmt</td>
<td>See the MSDN documentation for information about this member.</td>
</tr>
<tr>
<td>uType</td>
<td>See the MSDN documentation for information about this member.</td>
</tr>
</tbody>
</table>
**Public Instance Methods**

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong> (inherited from Object)</td>
<td>Determines whether the specified Object is equal to the current Object.</td>
</tr>
<tr>
<td><strong>GetHashCode</strong> (inherited from Object)</td>
<td>Serves as a hash function for a particular type. GetHashCode is suitable for use in hashing algorithms and data structures like a hash table.</td>
</tr>
<tr>
<td><strong>GetType</strong> (inherited from Object)</td>
<td>Gets the Type of the current instance.</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td></td>
</tr>
</tbody>
</table>

**See Also**

[DdeTransaction Class] | [NDde.Advanced Namespace]

[Send comments on this topic.]

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
## DdeTransaction Properties

The properties of the **DdeTransaction** class are listed below. For a complete list of **DdeTransaction** class members, see the [DdeTransaction Members](#) topic.

### Public Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>dwData1</strong></td>
<td>See the MSDN documentation for information about this member.</td>
</tr>
<tr>
<td><strong>dwData2</strong></td>
<td>See the MSDN documentation for information about this member.</td>
</tr>
<tr>
<td><strong>dwRet</strong></td>
<td>This gets the return value of the DDEML callback function. See the MSDN documentation for information about this member.</td>
</tr>
<tr>
<td><strong>hConv</strong></td>
<td>See the MSDN documentation for information about this member.</td>
</tr>
<tr>
<td><strong>hData</strong></td>
<td>See the MSDN documentation for information about this member.</td>
</tr>
<tr>
<td><strong>hsz1</strong></td>
<td>See the MSDN documentation for information about this member.</td>
</tr>
<tr>
<td><strong>hsz2</strong></td>
<td>See the MSDN documentation for information about this member.</td>
</tr>
<tr>
<td><strong>uFmt</strong></td>
<td>See the MSDN documentation for information about this member.</td>
</tr>
<tr>
<td><strong>uType</strong></td>
<td>See the MSDN documentation for information about this member.</td>
</tr>
</tbody>
</table>
See Also

DdeTransaction Class | NDde.Advanced Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeTransaction.dwData1 Property

See the MSDN documentation for information about this member.

```csharp
public IntPtr dwData1 {get;}
```

See Also

DdeTransaction Class | NDde.Advanced Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeTransaction.dwData2 Property

See the MSDN documentation for information about this member.

```csharp
public IntPtr dwData2 {get;}
```

See Also

[DdeTransaction Class] | [NDde.Advanced Namespace]

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeTransaction.dwRet Property

This gets the return value of the DDEML callback function. See the MSDN documentation for information about this member.

```csharp
public IntPtr dwRet {get; set;}
```

Remarks

This will be ignored if the PreFilterTransaction method returns false.

See Also

DdeTransaction Class | NDde.Advanced Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeTransaction.hConv Property

See the MSDN documentation for information about this member.

public IntPtr hConv {get;}

See Also

DdeTransaction Class | NDde.Advanced Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeTransaction.hData Property

See the MSDN documentation for information about this member.

```
public IntPtr hData {get;}
```

See Also

[DdeTransaction Class] | [NDde.Advanced Namespace]

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeTransaction.hsz1 Property

See the MSDN documentation for information about this member.

```csharp
public IntPtr hsz1 {get;}
```

See Also

DdeTransaction Class | NDde.Advanced Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeTransaction.hsz2 Property

See the MSDN documentation for information about this member.

```csharp
public IntPtr hsz2 {get;}
```

See Also

DdeTransaction Class | NDde.Advanced Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
DdeTransaction.uFmt Property

See the MSDN documentation for information about this member.

```csharp
public Int32 uFmt {get;}
```

See Also

- [DdeTransaction Class](#) | [NDde.Advanced Namespace](#)

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeTransaction.uType Property

See the MSDN documentation for information about this member.

```csharp
public Int32 uType {get;}
```

See Also

DdeTransaction Class | NDde.Advanced Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeTransaction Methods

The methods of the **DdeTransaction** class are listed below. For a complete list of **DdeTransaction** class members, see the **DdeTransaction Members** topic.

**Public Instance Methods**

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
</table>
| **Equals**      | (inherited from **Object**)
|                 | Determines whether the specified **Object** is equal to the current **Object**. |
| **GetHashCode** | (inherited from **Object**)
|                 | Serves as a hash function for a particular type. **GetHashCode** is suitable for use in hashing algorithms and data structures like a hash table. |
| **GetType**     | (inherited from **Object**)
|                 | Gets the **Type** of the current instance.                                  |
| **ToString**    |                                                                             |

**See Also**

[DdeTransaction Class] | [NDde.Advanced Namespace]

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
public override string ToString();

Return Value

See Also

DdeTransaction Class | NDde.Advanced Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
This defines a transaction filter.

For a list of all members of this type, see IDdeTransactionFilter Members.

**public interface IDdeTransactionFilter**

***Remarks***

Use a transaction filter to intercept the DDEML callback function. The PreFilterTransaction method will be called every time the DDEML callback function executes. The Transaction object passed into the method contains the parameters of the DDE callback function. By using a transaction filter the developer has complete control over the DDEML. See the MSDN documentation for more information on using the DDEML.

**CAUTION** Incorrect usage of the DDEML can cause this library to function incorrectly and can lead to resource leaks.

***Requirements***

**Namespace:** NDde.Advanced

**Assembly:** NDde (in NDde.dll)

***See Also***

IDdeTransactionFilter Members | NDde.Advanced Namespace

__________________________________________________

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
IDdeTransactionFilter Members

IDdeTransactionFilter overview

Public Instance Methods

| PreFilterTransaction | This filters a transaction before it is dispatched. |

See Also

IDdeTransactionFilter Interface | NDde.Advanced Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
IDdeTransactionFilter Methods

The methods of the IDdeTransactionFilter interface are listed below. For a complete list of IDdeTransactionFilter interface members, see the IDdeTransactionFilter Members topic.

Public Instance Methods

| PreFilterTransaction | This filters a transaction before it is dispatched. |

See Also

IDdeTransactionFilter Interface | NDde.Advanced Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
IDdeTransactionFilter.PreFilterTransaction Method

This filters a transaction before it is dispatched.

```csharp
bool PreFilterTransaction(DdeTransaction t);
```

Parameters

t
   The transaction to be dispatched.

Return Value

True to filter the transaction and stop it from being dispatched, false otherwise.

Remarks

This method is called everytime the DDEML callback function executes.

   CAUTION   Incorrect usage of the DDEML can cause this library to function incorrectly and can lead to resource leaks.

See Also

IDdeTransactionFilter Interface | NDde.Advanced Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
**NDde.Advanced.Monitor Namespace**

This namespace contains classes for creating DDE monitors.

**Namespace hierarchy**

**Classes**

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DdeActivityEventArgs</td>
<td>This contains information about events on DdeMonitor.</td>
</tr>
<tr>
<td>DdeCallbackActivityEventArgs</td>
<td>This contains information about the CallbackActivity event.</td>
</tr>
<tr>
<td>DdeConversationActivityEventArgs</td>
<td>This contains information about the ConversationActivity event.</td>
</tr>
<tr>
<td>DdeErrorActivityEventArgs</td>
<td>This contains information about the ErrorActivity event.</td>
</tr>
<tr>
<td>DdeLinkActivityEventArgs</td>
<td>This contains information about the LinkActivity event.</td>
</tr>
<tr>
<td>DdeMessageActivityEventArgs</td>
<td>This contains information about the MessageActivity event.</td>
</tr>
<tr>
<td>DdeMonitor</td>
<td>This is used to monitor DDE activity.</td>
</tr>
</tbody>
</table>

**Enumerations**

<table>
<thead>
<tr>
<th>Enumeration</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DdeMessageActivityKind</td>
<td>This represents the kind of message</td>
</tr>
<tr>
<td><strong>DdeMonitorFlags</strong></td>
<td>This specifies the different kinds of DDE activity that can be monitored.</td>
</tr>
</tbody>
</table>

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Dynamic Data Exchange Library for .NET
This contains information about events on `DdeMonitor`.

For a list of all members of this type, see `DdeActivityEventArgs Members`.

**System.Object**  **EventArgs**  **DdeEventArgs**  **DdeActivityEventArgs**  **DdeCallbackActivityEventArgs**  **DdeConversationActivityEventArgs**  **DdeErrorActivityEventArgs**  **DdeLinkActivityEventArgs**  **DdeMessageActivityEventArgs**

```csharp
public class DdeActivityEventArgs : EventArgs

Thread Safety

Public static (Shared in Visual Basic) members of this type are safe for multithreaded operations. Instance members are **not** guaranteed to be thread-safe.

Requirements

**Namespace:** `NDde.Advanced.Monitor`  
**Assembly:** NDde (in NDde.dll)

See Also

`DdeActivityEventArgs Members`  |  `NDde.Advanced.Monitor Namespace`

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
# DdeActivityEventArgs Members

## DdeActivityEventArgs overview

### Public Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>TaskHandle</strong></td>
<td>This gets the task handle of the application associated with this event.</td>
</tr>
</tbody>
</table>

### Public Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong> (inherited from Object)</td>
<td>Determines whether the specified Object is equal to the current Object.</td>
</tr>
<tr>
<td><strong>GetHashCode</strong> (inherited from Object)</td>
<td>Serves as a hash function for a particular type. GetHashCode is suitable for use in hashing algorithms and data structures like a hash table.</td>
</tr>
<tr>
<td><strong>GetType</strong> (inherited from Object)</td>
<td>Gets the <strong>Type</strong> of the current instance.</td>
</tr>
<tr>
<td><strong>ToString</strong> (inherited from DdeEventArgs)</td>
<td>This returns a string containing the current values of all properties.</td>
</tr>
</tbody>
</table>

### Protected Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Finalize</strong> (inherited from Object)</td>
<td>Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection.</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong> (inherited from Object)</td>
<td>Creates a shallow copy of the current Object.</td>
</tr>
</tbody>
</table>

See Also
Dynamic Data Exchange Library for .NET
DdeActivityEventArgs Properties

The properties of the DdeActivityEventArgs class are listed below. For a complete list of DdeActivityEventArgs class members, see the DdeActivityEventArgs Members topic.

Public Instance Properties

<table>
<thead>
<tr>
<th>TaskHandle</th>
<th>This gets the task handle of the application associated with this event.</th>
</tr>
</thead>
</table>

See Also

DdeActivityEventArgs Class  | NDde.Advanced.Monitor Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeActivityEventArgs.TaskHandle Property

This gets the task handle of the application associated with this event.

```csharp
public IntPtr TaskHandle {get;}
```

See Also

DdeActivityEventArgs Class | NDde.Advanced.Monitor Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
This contains information about the CallbackActivity event.

For a list of all members of this type, see DdeCallbackActivityEventArgs Members.

System.Object EventArgs DdeEventArgs DdeActivityEventArgs DdeCallbackActivityEventArgs

public class DdeCallbackActivityEventArgs : DdeActivityEventArgs

Thread Safety

Public static (Shared in Visual Basic) members of this type are safe for multithreaded operations. Instance members are not guaranteed to be thread-safe.

Requirements

Namespace: NDde.Advanced.Monitor
Assembly: NDde (in NDde.dll)

See Also

DdeCallbackActivityEventArgs Members | NDde.Advanced.Monitor Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
# DdeCallbackActivityEventArgs Members

## DdeCallbackActivityEventArgs overview

### Public Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dwData1</td>
<td>See the MSDN documentation for information about this member.</td>
</tr>
<tr>
<td>dwData2</td>
<td>See the MSDN documentation for information about this member.</td>
</tr>
<tr>
<td>dwRet</td>
<td>This gets the return value of the DDEML callback function. See the MSDN documentation for information about this member.</td>
</tr>
<tr>
<td>hConv</td>
<td>See the MSDN documentation for information about this member.</td>
</tr>
<tr>
<td>hData</td>
<td>See the MSDN documentation for information about this member.</td>
</tr>
<tr>
<td>hsz1</td>
<td>See the MSDN documentation for information about this member.</td>
</tr>
<tr>
<td>hsz2</td>
<td>See the MSDN documentation for information about this member.</td>
</tr>
<tr>
<td>TaskHandle</td>
<td>(inherited from DdeActivityEventArgs) This gets the task handle of the application associated with this event.</td>
</tr>
<tr>
<td>uFmt</td>
<td>See the MSDN documentation for information about this member.</td>
</tr>
<tr>
<td>uType</td>
<td>See the MSDN documentation for information about this member.</td>
</tr>
</tbody>
</table>
for information about this member.

**Public Instance Methods**

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong> (inherited from <strong>Object</strong>)</td>
<td>Determines whether the specified <strong>Object</strong> is equal to the current <strong>Object</strong>.</td>
</tr>
<tr>
<td><strong>GetHashCode</strong> (inherited from <strong>Object</strong>)</td>
<td>Serves as a hash function for a particular type. <strong>GetHashCode</strong> is suitable for use in hashing algorithms and data structures like a hash table.</td>
</tr>
<tr>
<td><strong>GetType</strong> (inherited from <strong>Object</strong>)</td>
<td>Gets the <strong>Type</strong> of the current instance.</td>
</tr>
<tr>
<td><strong>ToString</strong> (inherited from <strong>DdeEventArgs</strong>)</td>
<td>This returns a string containing the current values of all properties.</td>
</tr>
</tbody>
</table>

**See Also**

- [DdeCallbackActivityEventArgs Class](#) | [NDde.Advanced.Monitor Namespace](#)

---

**Send comments on this topic.**

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
The properties of the `DdeCallbackActivityEventArgs` class are listed below. For a complete list of `DdeCallbackActivityEventArgs` class members, see the [DdeCallbackActivityEventArgs Members](#) topic.

### Public Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>dwData1</code></td>
<td>See the MSDN documentation for information about this member.</td>
</tr>
<tr>
<td><code>dwData2</code></td>
<td>See the MSDN documentation for information about this member.</td>
</tr>
<tr>
<td><code>dwRet</code></td>
<td>This gets the return value of the DDEML callback function. See the MSDN documentation for information about this member.</td>
</tr>
<tr>
<td><code>hConv</code></td>
<td>See the MSDN documentation for information about this member.</td>
</tr>
<tr>
<td><code>hData</code></td>
<td>See the MSDN documentation for information about this member.</td>
</tr>
<tr>
<td><code>hsz1</code></td>
<td>See the MSDN documentation for information about this member.</td>
</tr>
<tr>
<td><code>hsz2</code></td>
<td>See the MSDN documentation for information about this member.</td>
</tr>
<tr>
<td><code>TaskHandle</code></td>
<td>(inherited from <code>DdeActivityEventArgs</code>) This gets the task handle of the application associated with this event.</td>
</tr>
<tr>
<td><code>uFmt</code></td>
<td>See the MSDN documentation for information about this member.</td>
</tr>
<tr>
<td><strong>uType</strong></td>
<td>See the MSDN documentation for information about this member.</td>
</tr>
</tbody>
</table>

**See Also**

[DdeCallbackActivityEventArgs Class] | [NDde.Advanced.Monitor Namespace]

---

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeCallbackActivityEventArgs.dwData1 Property

See the MSDN documentation for information about this member.

public IntPtr dwData1 {get;}

See Also

DdeCallbackActivityEventArgs Class | NDde.Advanced.Monitor Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeCallbackActivityEventArgs.dwData2 Property

See the MSDN documentation for information about this member.

```csharp
public IntPtr dwData2 {get;}
```

See Also

- [DdeCallbackActivityEventArgs Class](#)
- [NDde.Advanced.Monitor Namespace](#)

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
DdeCallbackActivityEventArgs.dwRet Property

This gets the return value of the DDEML callback function. See the MSDN documentation for information about this member.

```csharp
public IntPtr dwRet {get;}
```

See Also

- DdeCallbackActivityEventArgs Class | NDde.Advanced.Monitor Namespace

---

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeCallbackActivityEventArgs.hConv Property

See the MSDN documentation for information about this member.

```csharp
public IntPtr hConv {get;}
```

See Also

DdeCallbackActivityEventArgs Class | NDde.Advanced.Monitor Namespace

---

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
DdeCallbackActivityEventArgs.hData Property

See the MSDN documentation for information about this member.

```csharp
public IntPtr hData {get;}
```

See Also

[DdeCallbackActivityEventArgs Class | NDde.Advanced.Monitor Namespace]

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeCallbackActivityEventArgs.hsz1 Property

See the MSDN documentation for information about this member.

```csharp
public IntPtr hsz1 {get;}
```

See Also

DdeCallbackActivityEventArgs Class | NDde.Advanced.Monitor Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeCallbackActivityEventArgs.hsz2 Property

See the MSDN documentation for information about this member.

```csharp
public IntPtr hsz2 {get;}
```

See Also

DdeCallbackActivityEventArgs Class | NDde.Advanced.Monitor Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
DdeCallbackActivityEventArgs.uFmt Property

See the MSDN documentation for information about this member.

```csharp
public Int32 uFmt {get;}
```

See Also

DdeCallbackActivityEventArgs Class | NDde.Advanced.Monitor Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
See the MSDN documentation for information about this member.

```
public Int32 uType {get;}
```

See Also

[DdeCallbackActivityEventArgs Class](#)  |  [NDde.Advanced.Monitor Namespace](#)

[Send comments on this topic.](#)

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
**DdeConversationActivityEventArgs Class**

This contains information about the `ConversationActivity` event.

For a list of all members of this type, see `DdeConversationActivityEventArgs Members`.

```
public class DdeConversationActivityEventArgs : DdeActivityEventArgs
```

**Thread Safety**

Public static *(Shared in Visual Basic)* members of this type are safe for multithreaded operations. Instance members are not guaranteed to be thread-safe.

**Requirements**

**Namespace:** `NDde.Advanced.Monitor`

**Assembly:** NDde (in NDde.dll)

**See Also**

`DdeConversationActivityEventArgs Members` | `NDde.Advanced.Monitor Namespace`

---

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
# DdeConversationActivityEventArgs Members

## DdeConversationActivityEventArgs overview

### Public Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ClientHandle</strong></td>
<td>This gets the handle to the client application associated with the conversation.</td>
</tr>
<tr>
<td><strong>IsEstablished</strong></td>
<td>This gets a bool indicating whether the conversation is being established.</td>
</tr>
<tr>
<td><strong>ServerHandle</strong></td>
<td>This gets the handle to the server application associated with the conversation.</td>
</tr>
<tr>
<td><strong>Service</strong></td>
<td>This gets the service name associated with the conversation.</td>
</tr>
<tr>
<td><strong>TaskHandle</strong> (inherited from DdeActivityEventArgs)</td>
<td>This gets the task handle of the application associated with this event.</td>
</tr>
<tr>
<td><strong>Topic</strong></td>
<td>This gets the topic name associated with the conversation.</td>
</tr>
</tbody>
</table>

### Public Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong> (inherited from Object)</td>
<td>Determines whether the specified Object is equal to the current Object.</td>
</tr>
<tr>
<td><strong>GetHashCode</strong> (inherited from Object)</td>
<td>Serves as a hash function for a particular type. GetHashCode is suitable for use in hashing algorithms and data structures like a hash table.</td>
</tr>
<tr>
<td><strong>GetType</strong> (inherited from Object)</td>
<td>Gets the Type of the current</td>
</tr>
</tbody>
</table>
###ToString (inherited from DdeEventArgs)

<table>
<thead>
<tr>
<th><strong>Object)</strong></th>
<th>instance.</th>
</tr>
</thead>
</table>

This returns a string containing the current values of all properties.

###See Also

[DdeConversationActivityEventArgs Class] | [NDde.Advanced.Monitor Namespace]

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
The properties of the `DdeConversationActivityEventArgs` class are listed below. For a complete list of `DdeConversationActivityEventArgs` class members, see the `DdeConversationActivityEventArgs Members` topic.

### Public Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ClientHandle</code></td>
<td>This gets the handle to the client application associated with the conversation.</td>
</tr>
<tr>
<td><code>IsEstablished</code></td>
<td>This gets a bool indicating whether the conversation is being established.</td>
</tr>
<tr>
<td><code>ServerHandle</code></td>
<td>This gets the handle to the server application associated with the conversation.</td>
</tr>
<tr>
<td><code>Service</code></td>
<td>This gets the service name associated with the conversation.</td>
</tr>
<tr>
<td><code>TaskHandle</code> (inherited from <code>DdeActivityEventArgs</code>)</td>
<td>This gets the task handle of the application associated with this event.</td>
</tr>
<tr>
<td><code>Topic</code></td>
<td>This gets the topic name associated with the conversation.</td>
</tr>
</tbody>
</table>

See Also

- `DdeConversationActivityEventArgs Class` | `NDde.Advanced.Monitor Namespace`

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
This gets the handle to the client application associated with the conversation.

```csharp
public IntPtr ClientHandle {get;}
```

See Also

[DdeConversationActivityEventArgs Class] | [NDde.Advanced.Monitor Namespace]

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
This gets a bool indicating whether the conversation is being established.

```csharp
public Boolean IsEstablished {get;}
```

**Remarks**

The value returned by this property will be true if the conversation is being established. If the conversation is being terminated then the value will be false.

**See Also**

[DdeConversationActivityEventArgs Class](#) | [NDde.Advanced.Monitor Namespace](#)

---

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
This gets the handle to the server application associated with the conversation.

```csharp
public IntPtr ServerHandle {get;}
```

See Also

- DdeConversationActivityEventArgs Class | NDde.Advanced.Monitor Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeConversationActivityEventArgs.Service Property

This gets the service name associated with the conversation.

```
public String Service {get;}
```

See Also

[DdeConversationActivityEventArgs Class] | [NDde.Advanced.Monitor Namespace]

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeConversationActivityEventArgs.Topic Property

This gets the topic name associated with the conversation.

```csharp
public String Topic {get;}
```

See Also

*DdeConversationActivityEventArgs Class* | *NDde.Advanced.Monitor Namespace*

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeErrorActivityEventArgs Class

This contains information about the ErrorActivity event.

For a list of all members of this type, see DdeErrorActivityEventArgs Members.

System.Object EventArgs
    DdeEventArgs
        DdeActivityEventArgs
            DdeErrorActivityEventArgs

public class DdeErrorActivityEventArgs : DdeActivityEventArgs

Thread Safety

Public static (Shared in Visual Basic) members of this type are safe for multithreaded operations. Instance members are not guaranteed to be thread-safe.

Requirements

Namespace: NDde.Advanced.Monitor
Assembly: NDde (in NDde.dll)

See Also

DdeErrorActivityEventArgs Members | NDde.Advanced.Monitor Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeErrorActivityEventArgs Members

DdeErrorActivityEventArgs overview

Public Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Code</td>
<td>This gets an error code returned by the DDEML.</td>
</tr>
<tr>
<td>TaskHandle (inherited from DdeActivityEventArgs)</td>
<td>This gets the task handle of the application associated with this event.</td>
</tr>
</tbody>
</table>

Public Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Equals (inherited from Object)</td>
<td>Determines whether the specified Object is equal to the current Object.</td>
</tr>
<tr>
<td>GetHashCode (inherited from Object)</td>
<td>Serves as a hash function for a particular type. GetHashCode is suitable for use in hashing algorithms and data structures like a hash table.</td>
</tr>
<tr>
<td>GetType (inherited from Object)</td>
<td>Gets the Type of the current instance.</td>
</tr>
<tr>
<td>ToString (inherited from DdeEventArgs)</td>
<td>This returns a string containing the current values of all properties.</td>
</tr>
</tbody>
</table>

See Also

DdeErrorActivityEventArgs Class | NDde.Advanced.Monitor Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeErrorActivityEventArgs Properties

The properties of the DdeErrorActivityEventArgs class are listed below. For a complete list of DdeErrorActivityEventArgs class members, see the DdeErrorActivityEventArgs Members topic.

Public Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Code</strong></td>
<td>This gets an error code returned by the DDEML.</td>
</tr>
<tr>
<td><strong>TaskHandle</strong></td>
<td>This gets the task handle of the application associated with this event.</td>
</tr>
</tbody>
</table>

See Also

DdeErrorActivityEventArgs Class | NDde.Advanced.Monitor Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeErrorActivityEventArgs.Code Property

This gets an error code returned by the DDEML.

```csharp
public Int32 Code {get;}
```

See Also

DdeErrorActivityEventArgs Class | NDde.Advanced.Monitor Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeLinkActivityEventArgs Class

This contains information about the LinkActivity event.

For a list of all members of this type, see DdeLinkActivityEventArgs Members.

System.Object EventArgs DdeEventArgs DdeActivityEventArgs DdeLinkActivityEventArgs

```
public class DdeLinkActivityEventArgs : DdeActivityEventArgs
```

Thread Safety

Public static (Shared in Visual Basic) members of this type are safe for multithreaded operations. Instance members are not guaranteed to be thread-safe.

Requirements

Namespace: NDde.Advanced.Monitor
Assembly: NDde (in NDde.dll)

See Also

DdeLinkActivityEventArgs Members | NDde.Advanced.Monitor Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
## DdeLinkActivityEventArgs Members

### DdeLinkActivityEventArgs overview

#### Public Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ClientHandle</strong></td>
<td>This gets the handle to the client application associated with the link.</td>
</tr>
<tr>
<td><strong>Format</strong></td>
<td>This gets the format of the data associated with the link.</td>
</tr>
<tr>
<td><strong>IsEstablished</strong></td>
<td>This gets a bool indicating whether the link is being established.</td>
</tr>
<tr>
<td><strong>IsHot</strong></td>
<td>This gets a bool indicating whether the link is hot.</td>
</tr>
<tr>
<td><strong>IsServerInitiated</strong></td>
<td>This gets a bool indicating whether the link was terminated by the server.</td>
</tr>
<tr>
<td><strong>Item</strong></td>
<td>This gets the item name associated with the link.</td>
</tr>
<tr>
<td><strong>ServerHandle</strong></td>
<td>This gets the handle to the server application associated with the link.</td>
</tr>
<tr>
<td><strong>Service</strong></td>
<td>This gets the service name associated with the link.</td>
</tr>
<tr>
<td><strong>TaskHandle</strong></td>
<td>(inherited from DdeActivityEventArgs) This gets the task handle of the</td>
</tr>
<tr>
<td></td>
<td>application associated with this event.</td>
</tr>
<tr>
<td><strong>Topic</strong></td>
<td>This gets the topic name associated with the link.</td>
</tr>
</tbody>
</table>

#### Public Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong></td>
<td>(inherited from Object) Determines whether the</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>.AppendLine()</td>
<td>Serves as a hash function for a particular type. <strong>GetHashCode</strong> is suitable for use in hashing algorithms and data structures like a hash table.</td>
</tr>
<tr>
<td>GetType() (inherited from Object)</td>
<td>Gets the <strong>Type</strong> of the current instance.</td>
</tr>
<tr>
<td>ToString() (inherited from DdeEventArgs)</td>
<td>This returns a string containing the current values of all properties.</td>
</tr>
</tbody>
</table>

**See Also**

[DdeLinkActivityEventArgs Class](NDde_Advanced_MonitorNamespace) | [NDde.Advanced.Monitor Namespace](NDde_Advanced_MonitorNamespace)

[Send comments on this topic.](Send comments on this topic)
Dynamic Data Exchange Library for .NET
The properties of the **DdeLinkActivityEventArgs** class are listed below. For a complete list of **DdeLinkActivityEventArgs** class members, see the [DdeLinkActivityEventArgs Members](#) topic.

### Public Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ClientHandle</td>
<td>This gets the handle to the client application associated with the link.</td>
</tr>
<tr>
<td>Format</td>
<td>This gets the format of the data associated with the link.</td>
</tr>
<tr>
<td>IsEstablished</td>
<td>This gets a bool indicating whether the link is being established.</td>
</tr>
<tr>
<td>IsHot</td>
<td>This gets a bool indicating whether the link is hot.</td>
</tr>
<tr>
<td>IsServerInitiated</td>
<td>This gets a bool indicating whether the link was terminated by the server.</td>
</tr>
<tr>
<td>Item</td>
<td>This gets the item name associated with the link.</td>
</tr>
<tr>
<td>ServerHandle</td>
<td>This gets the handle to the server application associated with the link.</td>
</tr>
<tr>
<td>Service</td>
<td>This gets the service name associated with the link.</td>
</tr>
<tr>
<td>TaskHandle (inherited from DdeActivityEventArgs)</td>
<td>This gets the task handle of the application associated with this event.</td>
</tr>
<tr>
<td>Topic</td>
<td>This gets the topic name associated with the link.</td>
</tr>
</tbody>
</table>

**See Also**
Dynamic Data Exchange Library for .NET
**DdeLinkActivityEventArgs.ClientHandle Property**

This gets the handle to the client application associated with the link.

```csharp
public IntPtr ClientHandle {get;}
```

See Also

[DdeLinkActivityEventArgs Class](#) | [NDde.Advanced.Monitor Namespace](#)

________________________________________

**Send comments on this topic.**

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeLinkActivityEventArgs.Format Property

This gets the format of the data associated with the link.

```csharp
public Int32 Format {get;}
```

See Also

DdeLinkActivityEventArgs Class | NDde.Advanced.Monitor Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeLinkActivityEventArgs.IsEstablished Property

This gets a bool indicating whether the link is being established.

```csharp
public Boolean IsEstablished {get;}
```

Remarks

The value returned by this property will be true if the conversation is being established. If the conversation is being terminated then the value will be false.

See Also

DdeLinkActivityEventArgs Class | NDde.Advanced.Monitor Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
**DdeLinkActivityEventArgs.IsHot Property**

This gets a bool indicating whether the link is hot.

```csharp
public Boolean IsHot {get;}
```

**See Also**

[DdeLinkActivityEventArgs Class](#) | [NDde.Advanced.Monitor Namespace](#)

---

**Send comments on this topic.**

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeLinkActivityEventArgs.IsServerInitiated Property

This gets a bool indicating whether the link was terminated by the server.

```csharp
public Boolean IsServerInitiated {get;}
```

See Also

DdeLinkActivityEventArgs Class | NDde.Advanced.Monitor Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeLinkActivityEventArgs.Item Property

This gets the item name associated with the link.

```
public String Item {get;}
```

See Also

*DdeLinkActivityEventArgs Class* | *NDde.Advanced.Monitor Namespace*

---

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeLinkActivityEventArgs.ServerHandle Property

This gets the handle to the server application associated with the link.

```csharp
public IntPtr ServerHandle {get;}
```

See Also

- DdeLinkActivityEventArgs Class | NDde.Advanced.Monitor Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeLinkActivityEventArgs.Service Property

This gets the service name associated with the link.

```csharp
public String Service {get;}
```

See Also

DdeLinkActivityEventArgs Class | NDde.Advanced.Monitor Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
### DdeLinkActivityEventArgs.Topic Property

This gets the topic name associated with the link.

```csharp
public String Topic {get;}
```

**See Also**

[DdeLinkActivityEventArgs Class](#) | [NDde.Advanced.Monitor Namespace](#)

---

**Send comments on this topic.**

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeMessageActivityEventArgs Class

This contains information about the MessageActivity event.

For a list of all members of this type, see DdeMessageActivityEventArgs Members.

System.Object   EventArgs
                DdeEventArgs
                DdeActivityEventArgs
                DdeMessageActivityEventArgs

public class DdeMessageActivityEventArgs :
    DdeActivityEventArgs

Thread Safety

Public static (Shared in Visual Basic) members of this type are safe for multithreaded operations. Instance members are not guaranteed to be thread-safe.

Requirements

Namespace: NDde.Advanced.Monitor

Assembly: NDde (in NDde.dll)

See Also

DdeMessageActivityEventArgs Members | NDde.Advanced.Monitor
Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeMessageActivityEventArgs overview

Public Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Kind</strong></td>
<td>This gets the kind of message associated with this event.</td>
</tr>
<tr>
<td><strong>Message</strong></td>
<td>This gets the message associated with this event.</td>
</tr>
<tr>
<td><strong>TaskHandle</strong> (inherited from DdeActivityEventArgs)</td>
<td>This gets the task handle of the application associated with this event.</td>
</tr>
</tbody>
</table>

Public Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong> (inherited from Object)</td>
<td>Determines whether the specified Object is equal to the current Object.</td>
</tr>
<tr>
<td><strong>GetHashCode</strong> (inherited from Object)</td>
<td>Serves as a hash function for a particular type. GetHashCode is suitable for use in hashing algorithms and data structures like a hash table.</td>
</tr>
<tr>
<td><strong>GetType</strong> (inherited from Object)</td>
<td>Gets the Type of the current instance.</td>
</tr>
<tr>
<td><strong>ToString</strong> (inherited from DdeEventArgs)</td>
<td>This returns a string containing the current values of all properties.</td>
</tr>
</tbody>
</table>

See Also

DdeMessageActivityEventArgs Class | NDde.Advanced.Monitor Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
DdeMessageActivityEventArgs Properties

The properties of the `DdeMessageActivityEventArgs` class are listed below. For a complete list of `DdeMessageActivityEventArgs` class members, see the `DdeMessageActivityEventArgs Members` topic.

Public Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kind</td>
<td>This gets the kind of message associated with this event.</td>
</tr>
<tr>
<td>Message</td>
<td>This gets the message associated with this event.</td>
</tr>
<tr>
<td><code>TaskHandle</code> (inherited from <code>DdeActivityEventArgs</code>)</td>
<td>This gets the task handle of the application associated with this event.</td>
</tr>
</tbody>
</table>

See Also

- `DdeMessageActivityEventArgs Class` | `NDde.Advanced.Monitor Namespace`

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeMessageActivityEventArgs.Kind Property

This gets the kind of message associated with this event.

```csharp
public DdeMessageActivityKind Kind {get;}
```

See Also

[DdeMessageActivityEventArgs Class] | [NDde.Advanced.Monitor Namespace]

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeMessageActivityEventArgs.Message Property

This gets the message associated with this event.

```csharp
public Message Message {get;}
```

See Also

[DdeMessageActivityEventArgs Class] | [NDde.Advanced.Monitor Namespace]

---

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeMessageActivityKind Enumeration

This represents the kind of message contained in DdeMessageActivityEventArgs.

```csharp
public enum DdeMessageActivityKind
```

### Members

<table>
<thead>
<tr>
<th>Member Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Post</td>
<td>The message was posted by a DDE application.</td>
</tr>
<tr>
<td>Send</td>
<td>The message was sent by a DDE application.</td>
</tr>
</tbody>
</table>

### Requirements

- **Namespace:** NDde.Advanced.Monitor
- **Assembly:** NDde (in NDde.dll)

### See Also

- [NDde.Advanced.Monitor Namespace](#)

---

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET


**DdeMonitor Class**

This is used to monitor DDE activity.

For a list of all members of this type, see [DdeMonitor Members](#).

```csharp
System.Object  DdeMonitor
```

```csharp
public class DdeMonitor : IDisposable
```

**Thread Safety**

Public static *(Shared in Visual Basic)* members of this type are safe for multithreaded operations. Instance members are *not* guaranteed to be thread-safe.

**Requirements**

**Namespace:** NDde.Advanced.Monitor

**Assembly:** NDde (in NDde.dll)

**See Also**

[DdeMonitor Members](#) | [NDde.Advanced.Monitor Namespace](#)

---

*Send comments on this topic.*

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
# DdeMonitor Members

## DdeMonitor overview

### Public Instance Constructors

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>DdeMonitor Constructor</strong></td>
<td>This initializes a new instance of the DdeMonitor class.</td>
</tr>
</tbody>
</table>

### Public Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Context</strong></td>
<td>This gets the context associated with this instance.</td>
</tr>
</tbody>
</table>

### Public Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Dispose</strong></td>
<td>This releases all resources held by this instance.</td>
</tr>
<tr>
<td><strong>Equals</strong> (inherited from Object)</td>
<td>Determines whether the specified Object is equal to the current Object.</td>
</tr>
<tr>
<td><strong>GetHashCode</strong> (inherited from Object)</td>
<td>Serves as a hash function for a particular type. GetHashCode is suitable for use in hashing algorithms and data structures like a hash table.</td>
</tr>
<tr>
<td><strong>GetType</strong> (inherited from Object)</td>
<td>Gets the Type of the current instance.</td>
</tr>
<tr>
<td><strong>Start</strong></td>
<td>This starts monitoring the system for DDE activity.</td>
</tr>
<tr>
<td><strong>ToString</strong> (inherited from Object)</td>
<td>Returns a String that represents the current Object.</td>
</tr>
</tbody>
</table>

### Public Instance Events

<table>
<thead>
<tr>
<th>Event</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CallbackActivity</strong></td>
<td>This is raised anytime a DDEML callback is executed.</td>
</tr>
<tr>
<td>Activity</td>
<td>Description</td>
</tr>
<tr>
<td>----------------</td>
<td>---------------------------------------------------------------</td>
</tr>
<tr>
<td>ConversationActivity</td>
<td>This is raised anytime a conversation is established or terminated.</td>
</tr>
<tr>
<td>ErrorActivity</td>
<td>This is raised anytime there is an error.</td>
</tr>
<tr>
<td>LinkActivity</td>
<td>This is raised anytime an advise loop is established or terminated.</td>
</tr>
<tr>
<td>MessageActivity</td>
<td>This is raised anytime a DDE message is sent or posted.</td>
</tr>
</tbody>
</table>

See Also

[NDde.Advanced.Monitor Namespace](#)

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeMonitor Constructor

This initializes a new instance of the DdeMonitor class.

```java
public DdeMonitor();
```

See Also

DdeMonitor Class | NDde.Advanced.Monitor Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
The properties of the **DdeMonitor** class are listed below. For a complete list of **DdeMonitor** class members, see the [DdeMonitor Members topic](#).

### Public Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Context</strong></td>
<td>This gets the context associated with this instance.</td>
</tr>
</tbody>
</table>

**See Also**

[DdeMonitor Class](#) | [NDde.Advanced.Monitor Namespace](#)

---

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeMonitor.Context Property

This gets the context associated with this instance.

```
public DdeContext Context {get; set;}
```

See Also

DdeMonitor Class | NDde.Advanced.Monitor Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
**DdeMonitor Methods**

The methods of the **DdeMonitor** class are listed below. For a complete list of **DdeMonitor** class members, see the **DdeMonitor Members** topic.

### Public Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Dispose</strong></td>
<td>This releases all resources held by this instance.</td>
</tr>
<tr>
<td><strong>Equals</strong> (inherited from <strong>Object</strong>)</td>
<td>Determines whether the specified <strong>Object</strong> is equal to the current <strong>Object</strong>.</td>
</tr>
<tr>
<td><strong>GetHashCode</strong> (inherited from <strong>Object</strong>)</td>
<td>Serves as a hash function for a particular type. <strong>GetHashCode</strong> is suitable for use in hashing algorithms and data structures like a hash table.</td>
</tr>
<tr>
<td><strong>GetType</strong> (inherited from <strong>Object</strong>)</td>
<td>Gets the <strong>Type</strong> of the current instance.</td>
</tr>
<tr>
<td><strong>Start</strong></td>
<td>This starts monitoring the system for DDE activity.</td>
</tr>
<tr>
<td><strong>ToString</strong> (inherited from <strong>Object</strong>)</td>
<td>Returns a <strong>String</strong> that represents the current <strong>Object</strong>.</td>
</tr>
</tbody>
</table>

**See Also**

[DdeMonitor Class] | [NDde.Advanced.Monitor Namespace]

---

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeMonitor.Dispose Method

This releases all resources held by this instance.

```csharp
public void Dispose();
```

Implements

IDisposable.

See Also

DdeMonitor Class | NDde.Advanced.Monitor Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeMonitor.Start Method

This starts monitoring the system for DDE activity.

```csharp
public void Start(DdeMonitorFlags flags);
```

Parameters

`flags`  
A bitwise combination of `DdeMonitorFlags` that indicate what DDE activity will be monitored.

See Also

DdeMonitor Class | NDde.Advanced.Monitor Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
# DdeMonitor Events

The events of the **DdeMonitor** class are listed below. For a complete list of **DdeMonitor** class members, see the **DdeMonitor Members** topic.

## Public Instance Events

<table>
<thead>
<tr>
<th>Event</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CallbackActivity</td>
<td>This is raised anytime a DDEML callback is executed.</td>
</tr>
<tr>
<td>ConversationActivity</td>
<td>This is raised anytime a conversation is established or terminated.</td>
</tr>
<tr>
<td>ErrorActivity</td>
<td>This is raised anytime there is an error.</td>
</tr>
<tr>
<td>LinkActivity</td>
<td>This is raised anytime an advise loop is established or terminated.</td>
</tr>
<tr>
<td>MessageActivity</td>
<td>This is raised anytime a DDE message is sent or posted.</td>
</tr>
</tbody>
</table>

## See Also

[DdeMonitor Class](#) | [NDde.Advanced.Monitor Namespace](#)

---

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
**DdeMonitor.EventHandler<TEventArgs> Event**

This is raised anytime a DDEML callback is executed.

```csharp
public event EventHandler<TEventArgs> Callback;
```

**See Also**

[DdeMonitor Class](#) | [NDde.Advanced.Monitor Namespace](#)

*Send comments on this topic.*

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
**DdeMonitor.EventHandler<TEventArgs> Event**

This is raised anytime a conversation is established or terminated.

```csharp
public event EventHandler<TEventArgs> Conversation;
```

See Also

[DdeMonitor Class] | [NDde.Advanced.Monitor Namespace]

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeMonitor.EventHandler<TEventArgs> Event

This is raised anytime there is an error.

public event EventHandler<TEventArgs> ErrorActivity;

See Also

DdeMonitor Class | NDde.Advanced.Monitor Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
**DdeMonitor.EventHandler<TEventArgs> Event**

This is raised anytime an advise loop is established or terminated.

```csharp
public event EventHandler<TEventArgs> LinkActivity;
```

**See Also**

[DdeMonitor Class] [NDde.Advanced.Monitor Namespace]

---

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeMonitor.EventHandler<TEventArgs> Event

This is raised anytime a DDE message is sent or posted.

```
public event EventHandler<TEventArgs> Message;
```

See Also

DdeMonitor Class | NDde.Advanced.Monitor Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeMonitorFlags Enumeration

This specifies the different kinds of DDE activity that can be monitored.

This enumeration has a FlagsAttribute attribute that allows a bitwise combination of its member values.

```
public enum DdeMonitorFlags
```

Members

<table>
<thead>
<tr>
<th>Member Name</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Callback</td>
<td>This indicates activity caused by the execution of a DDEML callback.</td>
<td>134217728</td>
</tr>
<tr>
<td>Conversation</td>
<td>This indicates activity caused by conversation.</td>
<td>1073741824</td>
</tr>
<tr>
<td>Error</td>
<td>This indicates activity caused by an error.</td>
<td>268435456</td>
</tr>
<tr>
<td>Link</td>
<td>This indicates activity caused by an advise loop.</td>
<td>536870912</td>
</tr>
<tr>
<td>Message</td>
<td>This indicates activity caused by DDE messages.</td>
<td>100663296</td>
</tr>
</tbody>
</table>

Requirements

**Namespace:** NDde.Advanced.Monitor

**Assembly:** NDde (in NDde.dll)

See Also

NDde.Advanced.Monitor Namespace

Send comments on this topic.
Dynamic Data Exchange Library for .NET
NDde.Client Namespace

This namespace contains classes for creating DDE client applications.

Namespace hierarchy

Classes

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DdeAdviseEventArgs</td>
<td>This contains information about the Advise event.</td>
</tr>
<tr>
<td>DdeClient</td>
<td>This represents the client side of a DDE conversation.</td>
</tr>
<tr>
<td>DdeDisconnectedEventArgs</td>
<td>This contains information about the Disconnected event.</td>
</tr>
</tbody>
</table>

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Dynamic Data Exchange Library for .NET
This contains information about the Advise event.

For a list of all members of this type, see DdeAdviseEventArgs Members.

System.Object   EventArgs
    DdeEventArgs
    DdeAdviseEventArgs

public class DdeAdviseEventArgs : DdeEventArgs

Thread Safety

Public static (Shared in Visual Basic) members of this type are safe for multithreaded operations. Instance members are not guaranteed to be thread-safe.

Requirements

Namespace: NDde.Client

Assembly: NDde (in NDde.dll)

See Also

DdeAdviseEventArgs Members | NDde.Client Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
# DdeAdviseEventArgs Members

## DdeAdviseEventArgs overview

### Public Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>This gets the data associated with this notification or null if this is not a hot advise loop.</td>
</tr>
<tr>
<td>Format</td>
<td>This gets the format of the data included in this notification.</td>
</tr>
<tr>
<td>Item</td>
<td>This gets the item name associated with this notification.</td>
</tr>
<tr>
<td>State</td>
<td>This gets an application defined data object associated with this advise loop.</td>
</tr>
<tr>
<td>Text</td>
<td>This gets the text associated with this notification or null if this is not a hot advise loop.</td>
</tr>
</tbody>
</table>

### Public Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Equals (inherited from Object)</td>
<td>Determines whether the specified Object is equal to the current Object.</td>
</tr>
<tr>
<td>GetHashCode (inherited from Object)</td>
<td>Serves as a hash function for a particular type. GetHashCode is suitable for use in hashing algorithms and data structures like a hash table.</td>
</tr>
<tr>
<td>GetType (inherited from Object)</td>
<td>Gets the Type of the current instance.</td>
</tr>
<tr>
<td>ToString (inherited from DdeEventArgs)</td>
<td>This returns a string containing the current values of all properties.</td>
</tr>
</tbody>
</table>
See Also

DdeAdviseEventArgs Class | NDde.Client Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeAdviseEventArgs Properties

The properties of the **DdeAdviseEventArgs** class are listed below. For a complete list of **DdeAdviseEventArgs** class members, see the **DdeAdviseEventArgs Members** topic.

**Public Instance Properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Data</strong></td>
<td>This gets the data associated with this notification or null if this is not a hot advise loop.</td>
</tr>
<tr>
<td><strong>Format</strong></td>
<td>This gets the format of the data included in this notification.</td>
</tr>
<tr>
<td><strong>Item</strong></td>
<td>This gets the item name associated with this notification.</td>
</tr>
<tr>
<td><strong>State</strong></td>
<td>This gets an application defined data object associated with this advise loop.</td>
</tr>
<tr>
<td><strong>Text</strong></td>
<td>This gets the text associated with this notification or null if this is not a hot advise loop.</td>
</tr>
</tbody>
</table>

See Also

[DdeAdviseEventArgs Class] | [NDde.Client Namespace]

---

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
DdeAdviseEventArgs.Data Property

This gets the data associated with this notification or null if this is not a hot advise loop.

```csharp
public Byte[] Data {get;}
```

See Also

[DdeAdviseEventArgs Class] | [NDde.Client Namespace]

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
**DdeAdviseEventArgs.Format Property**

This gets the format of the data included in this notification.

```csharp
public Int32 Format {get;}
```

**See Also**

[DdeAdviseEventArgs Class] | [NDde.Client Namespace]

[Send comments on this topic.]

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeAdviseEventArgs.Item Property

This gets the item name associated with this notification.

```csharp
public String Item {get;}
```

See Also

DdeAdviseEventArgs Class | NDde.Client Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeAdviseEventArgs.State Property

This gets an application defined data object associated with this advise loop.

```csharp
public Object State {get;}
```

See Also

DdeAdviseEventArgs Class | NDde.Client Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
**DdeAdviseEventArgs.Text Property**

This gets the text associated with this notification or null if this is not a hot advise loop.

```java
public String Text {get;}
```

**See Also**

[DdeAdviseEventArgs Class] | [NDde.Client Namespace]

[Send comments on this topic.]

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
**DdeClient Class**

This represents the client side of a DDE conversation. For a list of all members of this type, see DdeClient Members.

**System.Object**   **DdeClient**

```java
public class DdeClient : IDisposable
```

**Thread Safety**

This type is safe for multithreaded operations.

**Remarks**

DDE conversations are established by specifying a service name and topic name pair. The service name is usually the name of the application acting as a DDE server. A DDE server can respond to multiple service names, but most servers usually only respond to one. The topic name is a logical context for data and is defined by the server application. A server can and usually does support many topic names.

After a conversation has been established by calling Connect an application can read and write data using the Request and Poke methods respectively by specifying an item name supported by the active conversation. An item name identifies a unit of data. An application can also be notified of changes by initiating an advise loop on an item name using the StartAdvise method. Advise loops can either be warm or hot. A hot advise loop returns the data associated with an item name when it changes whereas a warm advise loop only notifies the application without sending any data. Commands can be sent to the server using the Execute method.

Callbacks and events are invoked on the thread hosting the DdeContext. All operations must be marshaled onto the thread hosting the DdeContext associated with this object. Method calls will block until that thread becomes available. An exception will be generated if the thread does not become available in a timely manner.
Example

The following example demonstrates how to use a DdeClient.

[C#]
using System;
using System.Text;
using NDde.Client;

public sealed class Client
{
    public static void Main(string[] args)
    {
        // Wait for the user to press ENTER before proceeding.
        Console.WriteLine("The Server sample must be running before the client can connect.");
        Console.WriteLine("Press ENTER to continue..."个百分"
        Console.ReadLine();
        try
        {
            // Create a client that connects to 'myapp|mytopic'.
            using (DdeClient client = new DdeClient("myapp", "mytopic"))
            {
                // Subscribe to the Disconnected event. This event will notify the application when a conversation has been terminated.
                client.Disconnected += OnDisconnected;
                // Connect to the server. It must be running or an exception will be thrown.
                client.Connect();

                // Synchronous Execute Operation
                client.Execute("mycommand", 60000);

                // Synchronous Poke Operation
                client.Poke("myitem", DateTime.Now.ToString(), 60000);

                // Synchronous Request Operation
                Console.WriteLine("Request: " + client.Request("myitem", 60000));

                // Asynchronous Execute Operation
            }
        }
    }
}
client.BeginExecute("mycommand", OnExecuteComplete, client);

// Asynchronous Poke Operation
client.BeginPoke("myitem", Encoding.ASCII.GetBytes(DateTime.Now.ToString() + ",0"), 1, OnPokeComplete, client);

// Asynchronous Request Operation
client.BeginRequest("myitem", 1, OnRequestComplete, client);

// Advise Loop
client.StartAdvise("myitem", 1, true, 60000);
client.Advise += OnAdvise;

// Wait for the user to press ENTER before proceeding.
Console.WriteLine("Press ENTER to quit...");
Console.ReadLine();

} } catch (Exception e) {
    Console.WriteLine(e.ToString());
    Console.WriteLine("Press ENTER to quit...");
    Console.ReadLine();
}

private static void OnExecuteComplete(IAsyncResult ar) {
    try {
        DdeClient client = (DdeClient)ar.AsyncState;
        client.EndExecute(ar);
        Console.WriteLine("OnExecuteComplete");
    } catch (Exception e) {
        Console.WriteLine("OnExecuteComplete: " + e.Message);
    }
}
private static void OnPokeComplete(IAsyncResult ar)
{
    try
    {
        DdeClient client = (DdeClient)ar.AsyncState;
        client.EndPoke(ar);
        Console.WriteLine("OnPokeComplete");
    }
    catch (Exception e)
    {
        Console.WriteLine("OnPokeComplete: "+e.Message);
    }
}

private static void OnRequestComplete(IAsyncResult ar)
{
    try
    {
        DdeClient client = (DdeClient)ar.AsyncState;
        byte[] data = client.EndRequest(ar);
        Console.WriteLine("OnRequestComplete: "+Encoding.ASCII.GetString(data));
    }
    catch (Exception e)
    {
        Console.WriteLine("OnRequestComplete: "+e.Message);
    }
}

private static void OnStartAdviseComplete(IAsyncResult ar)
{
    try
    {
        DdeClient client = (DdeClient)ar.AsyncState;
        client.EndStartAdvise(ar);
        Console.WriteLine("OnStartAdviseComplete");
    }
    catch (Exception e)
    {
        Console.WriteLine("OnStartAdviseComplete: "+e.Message);
    }
}
} catch (Exception e) {
    Console.WriteLine("OnStartAdviseComplete: " + e.Message);
}

private static void OnStopAdviseComplete(IAsyncResult ar) {
    try {
        DdeClient client = (DdeClient)ar.AsyncState;
        client.EndStopAdvise(ar);
    } catch (Exception e) {
        Console.WriteLine("OnStopAdviseComplete: " + e.Message);
    }
}

private static void OnAdvise(object sender, DdeAdviseEventArgs args) {
    Console.WriteLine("OnAdvise: " + args.Text);
}

private static void OnDisconnected(object sender, DdeDisconnectedEventArgs args) {
}

} // class
Imports System.Text
Imports NDde.Client

Module Program

Sub Main()

' Wait for the user to press ENT
Console.WriteLine("The Server sa
Console.WriteLine("Press ENTER t
Console.ReadLine()

Try

' Create a client that conne
Using client As DdeClient =

' Subscribe to the Disco
AddHandler client.Disconn

' Connect to the server.
client.Connect()

' Synchronous Execute Op
client.Execute("mycommand"

' Synchronous Poke Opera
client.Poke("myitem", Da

' Synchronous Request Ope
Console.WriteLine("Reque

' Asynchronous Execute O
client.BeginExecute("myc

' Asynchronous Poke Oper
client.BeginPoke("myitem"
' Asynchronous Request
client.BeginRequest("myitem", 1, AddressOf OnRequestComplete, client)

' Advise Loop
client.StartAdvise("myitem", 1, True, 60000)
AddHandler client.Advise, AddressOf OnAdvise

' Wait for the user to press ENTER before proceeding.
Console.WriteLine("Press ENTER to quit...")
Console.ReadLine()
End Using

Catch e As Exception
    Console.WriteLine(e.ToString())
    Console.WriteLine("Press ENTER to quit...")
    Console.ReadLine()
End Try
End Sub

Private Sub OnExecuteComplete(ByVal ar As IAsyncResult)
    Try
        Dim client As DdeClient = DirectCast(ar.AsyncState, DdeClient)
        client.EndExecute(ar)
        Console.WriteLine("OnExecuteComplete")
    Catch e As Exception
        Console.WriteLine("OnExecuteComplete: "+ e.Message)
    End Try
End Sub

Private Sub OnPokeComplete(ByVal ar As IAsyncResult)
    Try
        Dim client As DdeClient = DirectCast(ar.AsyncState, DdeClient)
Private Sub OnRequestComplete(ByVal ar As IAsyncResult)
    Try
        Dim client As DdeClient = DirectCast(ar.AsyncState, DdeClient)
        client.EndRequest(ar)
        Console.WriteLine("OnRequestComplete")
    Catch e As Exception
        Console.WriteLine("OnRequestComplete: "+e.Message)
    End Try
End Sub

Private Sub OnStartAdviseComplete(ByVal ar As IAsyncResult)
    Try
        Dim client As DdeClient = DirectCast(ar.AsyncState, DdeClient)
        client.EndStartAdvise(ar)
        Console.WriteLine("OnStartAdviseComplete")
    Catch e As Exception
        Console.WriteLine("OnStartAdviseComplete: "+e.Message)
    End Try
End Sub

Private Sub OnStopAdviseComplete(ByVal ar As IAsyncResult)
    Try
        Dim client As DdeClient = DirectCast(ar.AsyncState, DdeClient)
        client.EndStopAdvise(ar)
        Console.WriteLine("OnStopAdviseComplete")
    Catch e As Exception
        Console.WriteLine("OnStopAdviseComplete: "+e.Message)
    End Try
End Sub
Private Sub OnAdvise(ByVal sender As Object, ByVal args As DdeAdviseEventArgs)
Console.WriteLine("OnAdvise: " + args.Text)
End Sub

Private Sub OnDisconnected(ByVal sender As Object, ByVal args As DdeDisconnectedEventArgs)
Console.WriteLine(_
"OnDisconnected: " + _
"IsServerInitiated=" + args.IsServerInitiated.ToString() + _
"IsDisposed=" + args.IsDisposed.ToString())
End Sub

End Module

Requirements

Namespace: NDde.Client
Assembly: NDde (in NDde.dll)

See Also

DdeClient Members | NDde.Client Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
# DdeClient Members

## DdeClient overview

### Public Instance Constructors

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>DdeClient</code></td>
<td>Overloaded.</td>
</tr>
</tbody>
</table>

### Public Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>Context</code></td>
<td>This gets the context associated with this instance.</td>
</tr>
<tr>
<td><code>Handle</code></td>
<td>This gets the DDEML handle associated with this conversation.</td>
</tr>
<tr>
<td><code>IsConnected</code></td>
<td>This gets a bool indicating whether the conversation is established.</td>
</tr>
<tr>
<td><code>IsPaused</code></td>
<td>This gets a bool indicating whether this conversation is paused.</td>
</tr>
<tr>
<td><code>Service</code></td>
<td>This gets the service name associated with this conversation.</td>
</tr>
<tr>
<td><code>Topic</code></td>
<td>This gets the topic name associated with this conversation.</td>
</tr>
</tbody>
</table>

### Public Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>Abandon</code></td>
<td>This terminates an asynchronous operation.</td>
</tr>
<tr>
<td><code>BeginExecute</code></td>
<td>This begins an asynchronous operation to send a command to the server application.</td>
</tr>
<tr>
<td><code>BeginPoke</code></td>
<td>This begins an asynchronous operation to send data to the</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>BeginRequest</td>
<td>This begins an asynchronous operation to request data using the specified item name.</td>
</tr>
<tr>
<td>BeginStartAdvise</td>
<td>Overloaded.</td>
</tr>
<tr>
<td>BeginStopAdvise</td>
<td>This begins an asynchronous operation to terminate the advise loop for the specified item name.</td>
</tr>
<tr>
<td>Connect</td>
<td>This establishes a conversation with a server that supports the specified service name and topic name pair.</td>
</tr>
<tr>
<td>Disconnect</td>
<td>This terminates the current conversation.</td>
</tr>
<tr>
<td>Dispose</td>
<td>Overloaded. This terminates the current conversation and releases all resources held by this instance.</td>
</tr>
<tr>
<td>EndExecute</td>
<td>This throws any exception that occurred during the asynchronous operation.</td>
</tr>
<tr>
<td>EndPoke</td>
<td>This throws any exception that occurred during the asynchronous operation.</td>
</tr>
<tr>
<td>EndRequest</td>
<td>This gets the data returned by the server application for the operation.</td>
</tr>
<tr>
<td>EndStartAdvise</td>
<td>This throws any exception that occurred during the operation.</td>
</tr>
<tr>
<td>EndStopAdvise</td>
<td>This throws any exception that occurred during the operation.</td>
</tr>
<tr>
<td>Equals (inherited from Object)</td>
<td>Determines whether the</td>
</tr>
<tr>
<td>Method/Property</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>specified <strong>Object</strong> is equal to the current <strong>Object</strong>.</td>
<td><strong>Execute</strong> This sends a command to the server application.</td>
</tr>
<tr>
<td><strong>GetHashCode</strong> (inherited from <strong>Object</strong>)</td>
<td>Serves as a hash function for a particular type. <strong>GetHashCode</strong> is suitable for use in hashing algorithms and data structures like a hash table.</td>
</tr>
<tr>
<td><strong>GetType</strong> (inherited from <strong>Object</strong>)</td>
<td>Gets the <strong>Type</strong> of the current instance.</td>
</tr>
<tr>
<td><strong>Pause</strong></td>
<td>This pauses the current conversation.</td>
</tr>
<tr>
<td><strong>Poke</strong></td>
<td>Overloaded.</td>
</tr>
<tr>
<td><strong>Request</strong></td>
<td>Overloaded.</td>
</tr>
<tr>
<td><strong>Resume</strong></td>
<td>This resumes the current conversation.</td>
</tr>
<tr>
<td><strong>StartAdvise</strong></td>
<td>Overloaded.</td>
</tr>
<tr>
<td><strong>StopAdvise</strong></td>
<td>This terminates the advise loop for the specified item name.</td>
</tr>
<tr>
<td><strong>ToString</strong> (inherited from <strong>Object</strong>)</td>
<td>Returns a <strong>String</strong> that represents the current <strong>Object</strong>.</td>
</tr>
<tr>
<td><strong>TryConnect</strong></td>
<td>This establishes a conversation with a server that supports the specified service name and topic name pair.</td>
</tr>
<tr>
<td><strong>TryExecute</strong></td>
<td>This sends a command to the server application.</td>
</tr>
<tr>
<td><strong>TryPoke</strong></td>
<td>This sends data to the server application.</td>
</tr>
<tr>
<td><strong>TryRequest</strong></td>
<td>This requests data using the specified item name.</td>
</tr>
</tbody>
</table>
Public Instance Events

<table>
<thead>
<tr>
<th>Event</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Advise</td>
<td>This is raised when the data has changed for an item name that has an advise loop.</td>
</tr>
<tr>
<td>Disconnected</td>
<td>This is raised when the client has been disconnected.</td>
</tr>
</tbody>
</table>

Protected Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dispose</td>
<td>Overloaded. This contains the implementation to release all resources held by this instance.</td>
</tr>
<tr>
<td>Finalize (inherited from Object)</td>
<td>Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection.</td>
</tr>
<tr>
<td>MemberwiseClone (inherited from Object)</td>
<td>Creates a shallow copy of the current Object.</td>
</tr>
</tbody>
</table>

See Also

DdeClient Class | NDde.Client Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
DdeClient Constructor

Overload List

This initializes a new instance of the DdeClient class that can connect to a server that supports the specified service name and topic name pair.

    public DdeClient(String,String);

This initializes a new instance of the DdeClient class that can connect to a server that supports the specified service name and topic name pair and uses the specified context.

    public DdeClient(String,String,DdeContext);

This initializes a new instance of the DdeClient class that can connect to a server that supports the specified service name and topic name pair using the specified synchronizing object.

    public DdeClient(String,String,ISynchronizeInvoke);

See Also

DdeClient Class | NDde.Client Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
**DdeClient Constructor (String, String)**

This initializes a new instance of the `DdeClient` class that can connect to a server that supports the specified service name and topic name pair.

```csharp
public DdeClient(
    String service,
    String topic
);
```

**Parameters**

*service*

A service name supported by a server application.

*topic*

A topic name support by a server application.

**Exceptions**

<table>
<thead>
<tr>
<th>Exception Type</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ArgumentException</code></td>
<td>This is thown when servic or topic exceeds 255 characters.</td>
</tr>
<tr>
<td><code>ArgumentNullException</code></td>
<td>This is thrown when service or topic is a null reference.</td>
</tr>
</tbody>
</table>

**See Also**

[DdeClient Class] | [NDde.Client Namespace] | [DdeClient Constructor Overload List]

---

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
This initializes a new instance of the DdeClient class that can connect to a server that supports the specified service name and topic name pair using the specified synchronizing object.

```java
public DdeClient(
    String service,
    String topic,
    ISynchronizeInvoke synchronizingObject
);
```

Parameters

- **service**
  - A service name supported by a server application.

- **topic**
  - A topic name support by a server application.

- **synchronizingObject**
  - The synchronizing object to use for this instance.

Exceptions

<table>
<thead>
<tr>
<th>Exception Type</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ArgumentException</td>
<td>This is thrown when service or topic exceeds 255 characters.</td>
</tr>
<tr>
<td>ArgumentNullException</td>
<td>This is thrown when service or topic is a null reference.</td>
</tr>
</tbody>
</table>

See Also

- [DdeClient Class](#)
- [NDde.Client Namespace](#)
- [DdeClient Constructor Overload List](#)

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeClient Constructor (String, String, DdeContext)

This initializes a new instance of the DdeClient class that can connect to a server that supports the specified service name and topic name pair and uses the specified context.

```java
public DdeClient(
    String service,
    String topic,
    DdeContext context
);
```

Parameters

- **service**
  A service name supported by a server application.

- **topic**
  A topic name support by a server application.

- **context**
  The context to use for execution.

Exceptions

<table>
<thead>
<tr>
<th>Exception Type</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ArgumentException</td>
<td>This is thrown when service or topic exceeds 255 characters.</td>
</tr>
<tr>
<td>ArgumentNullException</td>
<td>This is thrown when service or topic is a null reference.</td>
</tr>
</tbody>
</table>

See Also

DdeClient Class | NDde.Client Namespace | DdeClient Constructor

Overload List

---

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
The properties of the **DdeClient** class are listed below. For a complete list of **DdeClient** class members, see the [DdeClient Members](#) topic.

### Public Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Context</strong></td>
<td>This gets the context associated with this instance.</td>
</tr>
<tr>
<td><strong>Handle</strong></td>
<td>This gets the DDEML handle associated with this conversation.</td>
</tr>
<tr>
<td><strong>IsConnected</strong></td>
<td>This gets a bool indicating whether the conversation is established.</td>
</tr>
<tr>
<td><strong>IsPaused</strong></td>
<td>This gets a bool indicating whether this conversation is paused.</td>
</tr>
<tr>
<td><strong>Service</strong></td>
<td>This gets the service name associated with this conversation.</td>
</tr>
<tr>
<td><strong>Topic</strong></td>
<td>This gets the topic name associated with this conversation.</td>
</tr>
</tbody>
</table>

See Also

[DdeClient Class](#) | [NDde.Client Namespace](#)

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeClient.Context Property

This gets the context associated with this instance.

```csharp
public virtual DdeContext Context {get; set;}
```

See Also

DdeClient Class | NDde.Client Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeClient.Handle Property

This gets the DDEML handle associated with this conversation.

```csharp
public virtual IntPtr Handle {get;}
```

Remarks

This can be used in any DDEML function requiring a conversation handle.

**CAUTION** Incorrect usage of the DDEML can cause this object to function incorrectly and can lead to resource leaks.

See Also

[DdeClient Class](#) | [NDde.Client Namespace](#)

---

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
DdeClient.IsConnected Property

This gets a bool indicating whether the conversation is established.

```csharp
public virtual Boolean IsConnected {get;}
```

Remarks

**CAUTION** Do not assume that the conversation is still established after checking this property. The conversation can terminate at any time.

See Also

DdeClient Class | NDde.Client Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeClient.IsPaused Property

This gets a bool indicating whether this conversation is paused.

```csharp
public virtual Boolean IsPaused {get;}
```

See Also

DdeClient Class | NDde.Client Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeClient.Service Property

This gets the service name associated with this conversation.

```csharp
public virtual String Service {get; set;}
```

See Also

[DdeClient Class | NDde.Client Namespace]

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeClient.Topic Property

This gets the topic name associated with this conversation.

```csharp
public virtual String Topic {get; set;}
```

See Also

DdeClient Class | NDde.Client Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
The methods of the **DdeClient** class are listed below. For a complete list of **DdeClient** class members, see the [DdeClient Members](#) topic.

### Public Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Abandon</strong></td>
<td>This terminates an asynchronous operation.</td>
</tr>
<tr>
<td><strong>BeginExecute</strong></td>
<td>This begins an asynchronous operation to send a command to the server application.</td>
</tr>
<tr>
<td><strong>BeginPoke</strong></td>
<td>This begins an asynchronous operation to send data to the server application.</td>
</tr>
<tr>
<td><strong>BeginRequest</strong></td>
<td>This begins an asynchronous operation to request data using the specified item name.</td>
</tr>
<tr>
<td><strong>BeginStartAdvise</strong></td>
<td>Overloaded.</td>
</tr>
<tr>
<td><strong>BeginStopAdvise</strong></td>
<td>This begins an asynchronous operation to terminate the advise loop for the specified item name.</td>
</tr>
<tr>
<td><strong>Connect</strong></td>
<td>This establishes a conversation with a server that supports the specified service name and topic name pair.</td>
</tr>
<tr>
<td><strong>Disconnect</strong></td>
<td>This terminates the current conversation.</td>
</tr>
<tr>
<td><strong>Dispose</strong></td>
<td>Overloaded. This terminates the current conversation and releases all resources held by this instance.</td>
</tr>
<tr>
<td><strong>EndExecute</strong></td>
<td>This throws any exception that</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td><strong>EndPoke</strong></td>
<td>This throws any exception that occurred during the asynchronous operation.</td>
</tr>
<tr>
<td><strong>EndRequest</strong></td>
<td>This gets the data returned by the server application for the operation.</td>
</tr>
<tr>
<td><strong>EndStartAdvise</strong></td>
<td>This throws any exception that occurred during the operation.</td>
</tr>
<tr>
<td><strong>EndStopAdvise</strong></td>
<td>This throws any exception that occurred during the operation.</td>
</tr>
<tr>
<td><strong>Equals</strong> (inherited from <strong>Object</strong>)</td>
<td>Determines whether the specified <strong>Object</strong> is equal to the current <strong>Object</strong>.</td>
</tr>
<tr>
<td><strong>Execute</strong></td>
<td>This sends a command to the server application.</td>
</tr>
<tr>
<td><strong>GetHashCode</strong> (inherited from <strong>Object</strong>)</td>
<td>Serves as a hash function for a particular type. <strong>GetHashCode</strong> is suitable for use in hashing algorithms and data structures like a hash table.</td>
</tr>
<tr>
<td><strong>GetType</strong> (inherited from <strong>Object</strong>)</td>
<td>Gets the <strong>Type</strong> of the current instance.</td>
</tr>
<tr>
<td><strong>Pause</strong></td>
<td>This pauses the current conversation.</td>
</tr>
<tr>
<td><strong>Poke</strong></td>
<td>Overloaded.</td>
</tr>
<tr>
<td><strong>Request</strong></td>
<td>Overloaded.</td>
</tr>
<tr>
<td><strong>Resume</strong></td>
<td>This resumes the current conversation.</td>
</tr>
<tr>
<td><strong>StartAdvise</strong></td>
<td>Overloaded.</td>
</tr>
<tr>
<td><strong>StopAdvise</strong></td>
<td>This terminates the advise loop.</td>
</tr>
</tbody>
</table>
for the specified item name.

<table>
<thead>
<tr>
<th><strong>ToString</strong> (inherited from Object)</th>
<th>Returns a String that represents the current Object.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>TryConnect</strong></td>
<td>This establishes a conversation with a server that supports the specified service name and topic name pair.</td>
</tr>
<tr>
<td><strong>TryExecute</strong></td>
<td>This sends a command to the server application.</td>
</tr>
<tr>
<td><strong>TryPoke</strong></td>
<td>This sends data to the server application.</td>
</tr>
<tr>
<td><strong>TryRequest</strong></td>
<td>This requests data using the specified item name.</td>
</tr>
</tbody>
</table>

**Protected Instance Methods**

<table>
<thead>
<tr>
<th><strong>Dispose</strong></th>
<th>Overloaded. This contains the implementation to release all resources held by this instance.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Finalize</strong> (inherited from Object)</td>
<td>Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection.</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong> (inherited from Object)</td>
<td>Creates a shallow copy of the current Object.</td>
</tr>
</tbody>
</table>

**See Also**

[NDde.Client Namespace](NDde.Client.Namespace)
Dynamic Data Exchange Library for .NET
DdeClient.Abandon Method

This terminates an asynchronous operation.

```csharp
public virtual void Abandon(IAsyncResult asyncResult);
```

Parameters

- `asyncResult`:
The `IAsyncResult` object returned by a call that begins an asynchronous operation.

Remarks

This method does nothing if the asynchronous operation has already completed.

Exceptions

<table>
<thead>
<tr>
<th>Exception Type</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ArgumentException</code></td>
<td>This is thrown when <code>asyncResult</code> is an invalid <code>IAsyncResult</code>.</td>
</tr>
<tr>
<td><code>ArgumentNullException</code></td>
<td>This is thrown when <code>asyncResult</code> is a null reference.</td>
</tr>
<tr>
<td><code>InvalidOperationException</code></td>
<td>This is thrown when the client is not connected.</td>
</tr>
<tr>
<td><code>DdeException</code></td>
<td>This is thrown when the asynchronous operation could not be abandoned.</td>
</tr>
</tbody>
</table>

See Also

- [DdeClient Class](#) | [NDde.Client Namespace](#)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeClient.BeginExecute Method

This begins an asynchronous operation to send a command to the server application.

```csharp
public virtual IAsyncResult BeginExecute(
    String command,
    AsyncCallback callback,
    Object state
);
```

Parameters

- `command`: The command to be sent to the server application.
- `callback`: The delegate to invoke when this operation completes.
- `state`: An application defined data object to associate with this operation.

Return Value

An `IAsyncResult` object for this operation.

Exceptions

<table>
<thead>
<tr>
<th>Exception Type</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ArgumentException</td>
<td>This is thrown when command exceeds 255 characters.</td>
</tr>
<tr>
<td>ArgumentNullException</td>
<td>This is thrown when command is a null reference.</td>
</tr>
<tr>
<td>InvalidOperationException</td>
<td>This is thrown when the client is not connected.</td>
</tr>
<tr>
<td>DdeException</td>
<td>This is thrown when the asynchronous operation could not begin.</td>
</tr>
</tbody>
</table>
See Also

DdeClient Class | NDde.Client Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
This begins an asynchronous operation to send data to the server application.

```csharp
public virtual IAsyncResult BeginPoke(
    String item,
    Byte[] data,
    Int32 format,
    AsyncCallback callback,
    Object state
);
```

**Parameters**

*item*  
An item name supported by the current conversation.

*data*  
The data to send.

*format*  
The format of the data.

*callback*  
The delegate to invoke when this operation completes.

*state*  
An application defined data object to associate with this operation.

**Return Value**

An IAsyncResult object for this operation.

**Exceptions**

<table>
<thead>
<tr>
<th>Exception Type</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ArgumentException</td>
<td>This is thrown when item exceeds 255 characters.</td>
</tr>
<tr>
<td>ArgumentNullException</td>
<td>This is thrown when item or</td>
</tr>
<tr>
<td>Exception</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------</td>
</tr>
<tr>
<td>InvalidOperationException</td>
<td>This is thrown when the client is not connected.</td>
</tr>
<tr>
<td>DdeException</td>
<td>This is thrown when the asynchronous operation could not begin.</td>
</tr>
</tbody>
</table>

See Also

DdeClient Class | NDde.Client Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeClient.BeginRequest Method

This begins an asynchronous operation to request data using the specified item name.

```csharp
public virtual IAsyncResult BeginRequest(
    String item,
    Int32 format,
    AsyncCallback callback,
    Object state
);
```

Parameters

- **item**
  - An item name supported by the current conversation.

- **format**
  - The format of the data to return.

- **callback**
  - The delegate to invoke when this operation completes.

- **state**
  - An application defined data object to associate with this operation.

Return Value

An IAsyncResult object for this operation.

Exceptions

<table>
<thead>
<tr>
<th>Exception Type</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ArgumentException</td>
<td>This is thrown when item exceeds 255 characters.</td>
</tr>
<tr>
<td>ArgumentNullException</td>
<td>This is thrown when item is a null reference.</td>
</tr>
<tr>
<td>InvalidOperationException</td>
<td>This is thrown when the client is not connected.</td>
</tr>
</tbody>
</table>
DdeException

This is thrown when the asynchronous operation could not begin.

See Also

DdeClient Class | NDde.Client Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeClient.BeginStartAdvise Method

Overload List

This begins an asynchronous operation to initiate an advise loop on the specified item name.

    public virtual IAsyncResult BeginStartAdvise(String,Int32,Boolean,A

This begins an asynchronous operation to initiate an advise loop on the specified item name.

    public virtual IAsyncResult BeginStartAdvise(String,Int32,Boolean,B

See Also

DdeClient Class | NDde.Client Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
This begins an asynchronous operation to initiate an advise loop on the specified item name.

```csharp
public virtual IAsyncResult BeginStartAdvise(
    String item,
    Int32 format,
    Boolean hot,
    AsyncCallback callback,
    Object asyncState
);
```

### Parameters

**item**

An item name supported by the current conversation.

**format**

The format of the data to be returned.

**hot**

A bool indicating whether data should be included with the notification.

**callback**

The delegate to invoke when this operation completes.

**asyncState**

An application defined data object to associate with this operation.

### Return Value

An `IAsyncResult` object for this operation.

### Events

<table>
<thead>
<tr>
<th>Event Type</th>
<th>Reason</th>
</tr>
</thead>
<tbody>
<tr>
<td>Advise</td>
<td></td>
</tr>
</tbody>
</table>
Exceptions

<table>
<thead>
<tr>
<th>Exception Type</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ArgumentException</strong></td>
<td>This is thrown when item exceeds 255 characters.</td>
</tr>
<tr>
<td><strong>ArgumentNullException</strong></td>
<td>This is thrown when item is a null reference.</td>
</tr>
<tr>
<td><strong>InvalidOperationException</strong></td>
<td>This is thrown when the item is already being advised or when the client is not connected.</td>
</tr>
<tr>
<td><strong>DdeException</strong></td>
<td>This is thrown when the asynchronous operation could not begin.</td>
</tr>
</tbody>
</table>

See Also

- [DdeClient Class](#) | [NDde.Client Namespace](#) | [DdeClient.BeginStartAdvise Overload List](#)

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
This begins an asynchronous operation to initiate an advise loop on the specified item name.

```csharp
public virtual IAsyncResult BeginStartAdvise(
    String item,
    Int32 format,
    Boolean hot,
    Boolean acknowledge,
    AsyncCallback callback,
    Object asyncState,
    Object adviseState
);
```

Parameters

- **item**
  An item name supported by the current conversation.

- **format**
  The format of the data to be returned.

- **hot**
  A bool indicating whether data should be included with the notification.

- **acknowledge**
  A bool indicating whether the client should acknowledge each advisory before the server will send another.

- **callback**
  The delegate to invoke when this operation completes.

- **asyncState**
  An application defined data object to associate with this operation.

- **adviseState**
  An application defined data object to associate with this advise loop.
Return Value

An `IAasyncResult` object for this operation.

Events

<table>
<thead>
<tr>
<th>Event Type</th>
<th>Reason</th>
</tr>
</thead>
<tbody>
<tr>
<td>Advise</td>
<td></td>
</tr>
</tbody>
</table>

Exceptions

<table>
<thead>
<tr>
<th>Exception Type</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ArgumentException</code></td>
<td>This is thrown when item exceeds 255 characters.</td>
</tr>
<tr>
<td><code>ArgumentNullException</code></td>
<td>This is thrown when item is a null reference.</td>
</tr>
<tr>
<td><code>InvalidOperationException</code></td>
<td>This is thrown when the item is already being advised or when the client is not connected.</td>
</tr>
<tr>
<td><code>DdeException</code></td>
<td>This is thrown when the asynchronous operation could not begin.</td>
</tr>
</tbody>
</table>

See Also

[DdeClient Class](#) | [NDde.Client Namespace](#) | [DdeClient.BeginStartAdvise Overload List](#)

---

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
This begins an asynchronous operation to terminate the advise loop for the specified item name.

```java
public virtual IAsyncResult BeginStopAdvise(
    String item,
    AsyncCallback callback,
    Object state
);
```

Parameters

- **item**
  An item name that has an active advise loop.

- **callback**
  The delegate to invoke when this operation completes.

- **state**
  An application defined data object to associate with this operation.

Return Value

An `IAsyncResult` object for this operation.

Exceptions

<table>
<thead>
<tr>
<th>Exception Type</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ArgumentException</td>
<td>This is thrown when item exceeds 255 characters.</td>
</tr>
<tr>
<td>ArgumentNullException</td>
<td>This is thrown when item is a null reference.</td>
</tr>
<tr>
<td>InvalidOperationException</td>
<td>This is thrown when the item is not being advised or when the client is not connected.</td>
</tr>
<tr>
<td>DdeException</td>
<td>This is thrown when the asynchronous operation could</td>
</tr>
</tbody>
</table>
See Also

DdeClient Class | NDde.Client Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeClient.Connect Method

This establishes a conversation with a server that supports the specified service name and topic name pair.

```csharp
public virtual void Connect();
```

Exceptions

<table>
<thead>
<tr>
<th>Exception Type</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>InvalidOperationException</td>
<td>This is thrown when the client is already connected.</td>
</tr>
<tr>
<td>DdeException</td>
<td>This is thrown when the client could not connect to the server.</td>
</tr>
</tbody>
</table>

See Also

DdeClient Class | NDde.Client Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
**DdeClient.Disconnect Method**

This terminates the current conversation.

```csharp
public virtual void Disconnect();
```

**Events**

<table>
<thead>
<tr>
<th>Event Type</th>
<th>Reason</th>
</tr>
</thead>
<tbody>
<tr>
<td>Disconnected</td>
<td></td>
</tr>
</tbody>
</table>

**Exceptions**

<table>
<thead>
<tr>
<th>Exception Type</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>InvalidOperationException</td>
<td>This is thrown when the client was not previously connected.</td>
</tr>
<tr>
<td>DdeException</td>
<td>This is thrown when the client could not disconnect from the server.</td>
</tr>
</tbody>
</table>

**See Also**

[DdeClient Class] | [NDde.Client Namespace]

*Send comments on this topic.*

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeClient.Dispose Method

This terminates the current conversation and releases all resources held by this instance.

Overload List

This terminates the current conversation and releases all resources held by this instance.

    public void Dispose();

This contains the implementation to release all resources held by this instance.

    protected virtual void Dispose(Boolean);

See Also

DdeClient Class | NDde.Client Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
This terminates the current conversation and releases all resources held by this instance.

```
public void Dispose();
```

Implements

`IDisposable`.

See Also

[DdeClient Class] | [NDde.Client Namespace] | [DdeClient.Dispose Overload List]

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
This contains the implementation to release all resources held by this instance.

```csharp
protected virtual void Dispose(
    Boolean disposing
);
```

Parameters

- **disposing**
  True if called by Dispose, false otherwise.

See Also

[DdeClient Class](#) | [NDde.Client Namespace](#) | [DdeClient.Dispose Overload List](#)

---

[Send comments on this topic.](#)

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeClient.EndExecute Method

This throws any exception that occurred during the asynchronous operation.

```csharp
public virtual void EndExecute(IAsyncResult asyncResult);
```

Parameters

`asyncResult`  
The `IAsyncResult` object returned by a call to `BeginExecute`.

Exceptions

<table>
<thead>
<tr>
<th>Exception Type</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ArgumentException</td>
<td>This is thown when asyncResult is an invalid IAsyncResult.</td>
</tr>
<tr>
<td>ArgumentException</td>
<td>This is thrown when asyncResult is a null reference.</td>
</tr>
<tr>
<td>ArgumentException</td>
<td>This is thrown when the server does not process the command.</td>
</tr>
</tbody>
</table>

See Also

DdeClient Class | NDde.Client Namespace

---

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
**DdeClient.EndPoke Method**

This throws any exception that occurred during the asynchronous operation.

```csharp
public virtual void EndPoke(IAsyncResult asyncResult);
```

**Parameters**

*asyncResult*

The `IAsyncResult` object returned by a call to `BeginPoke`.

**Exceptions**

<table>
<thead>
<tr>
<th>Exception Type</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ArgumentException</code></td>
<td>This is thrown when <code>asyncResult</code> is an invalid <code>IAsyncResult</code>.</td>
</tr>
<tr>
<td><code>ArgumentNullException</code></td>
<td>This is thrown when <code>asyncResult</code> is a null reference.</td>
</tr>
<tr>
<td><code>DdeException</code></td>
<td>This is thrown when the server does not process the data.</td>
</tr>
</tbody>
</table>

**See Also**

[DdeClient Class] | [NDde.Client Namespace]

*Send comments on this topic.*

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
This gets the data returned by the server application for the operation.

```csharp
public virtual byte[] EndRequest(IAsyncResult asyncResult);
```

**Parameters**

`asyncResult`

The `IAsyncResult` object returned by a call to `BeginRequest`.

**Return Value**

The data returned by the server application.

**Exceptions**

<table>
<thead>
<tr>
<th>Exception Type</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ArgumentException</code></td>
<td>This is thrown when <code>asyncResult</code> is an invalid <code>IAsyncResult</code>.</td>
</tr>
<tr>
<td><code>ArgumentNullException</code></td>
<td>This is thrown when <code>asyncResult</code> is a null reference.</td>
</tr>
<tr>
<td><code>DdeException</code></td>
<td>This is thrown when the server does not process the request.</td>
</tr>
</tbody>
</table>

**See Also**

- [DdeClient Class](#) | [NDde.Client Namespace](#)

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeClient.EndStartAdvise Method

This throws any exception that occurred during the operation.

```csharp
public virtual void EndStartAdvise(IAsyncResult asyncResult);
```

Parameters

`asyncResult`  
The `IAsyncResult` object returned by a call to `BeginPoke`.

Exceptions

<table>
<thead>
<tr>
<th>Exception Type</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ArgumentException</td>
<td>This is thrown when <code>asyncResult</code> is an invalid <code>IAsyncResult</code>.</td>
</tr>
<tr>
<td>ArgumentNullException</td>
<td>This is thrown when <code>asyncResult</code> is a null reference.</td>
</tr>
<tr>
<td>DdeException</td>
<td>This is thrown when the server does not initiate the advise loop.</td>
</tr>
</tbody>
</table>

See Also

DdeClient Class | NDde.Client Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeClient.EndStopAdvise Method

This throws any exception that occurred during the operation.

```csharp
public virtual void EndStopAdvise(
    IAsyncResult asyncResult
);
```

Parameters

`asyncResult`

The `IAsyncResult` object returned by a call to `BeginPoke`.

Exceptions

<table>
<thead>
<tr>
<th>Exception Type</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ArgumentException</td>
<td>This is thrown when <code>asyncResult</code> is an invalid <code>IAsyncResult</code>.</td>
</tr>
<tr>
<td>ArgumentNullException</td>
<td>This is thrown when <code>asyncResult</code> is a null reference.</td>
</tr>
<tr>
<td>DdeException</td>
<td>This is thrown when the server does not terminate the advise loop.</td>
</tr>
</tbody>
</table>

See Also

[DdeClient Class](#) | [NDde.Client Namespace](#)

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeClient.Execute Method

This sends a command to the server application.

```csharp
public virtual void Execute(string command, int timeout);
```

Parameters

- `command`  
The command to be sent to the server application.

- `timeout`  
The amount of time in milliseconds to wait for a response.

Remarks

This operation will timeout if the conversation is paused.

Exceptions

<table>
<thead>
<tr>
<th>Exception Type</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ArgumentException</td>
<td>This is thrown when command exceeds 255 characters or timeout is negative.</td>
</tr>
<tr>
<td>ArgumentNullException</td>
<td>This is thrown when command is a null reference.</td>
</tr>
<tr>
<td>InvalidOperationException</td>
<td>This is thrown when the client is not connected.</td>
</tr>
<tr>
<td>DdeException</td>
<td>This is thrown when the server does not process the command.</td>
</tr>
</tbody>
</table>

See Also

DdeClient Class | NDde.Client Namespace

Send comments on this topic.
Dynamic Data Exchange Library for .NET
DdeClient.Pause Method

This pauses the current conversation.

```csharp
public virtual void Pause();
```

Remarks

Synchronous operations will timeout if the conversation is paused. Asynchronous operations can begin, but will not complete until the conversation has resumed.

Exceptions

<table>
<thead>
<tr>
<th>Exception Type</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>InvalidOperationException</td>
<td>This is thrown when the conversation is already paused.</td>
</tr>
<tr>
<td>DdeException</td>
<td>This is thrown when the conversation could not be paused or when the client is not connected.</td>
</tr>
</tbody>
</table>

See Also

DdeClient Class | NDde.Client Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeClient.Poke Method

Overload List

This sends data to the server application.

public virtual void Poke(String, Byte[], Int32, Int32);

This sends data to the server application.

public virtual void Poke(String, String, Int32);

See Also

DdeClient Class | NDde.Client Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeClient.Poke Method (String, Byte[], Int32, Int32)

This sends data to the server application.

```csharp
public virtual void Poke(
    String item,
    Byte[] data,
    Int32 format,
    Int32 timeout
);
```

Parameters

* `item`
  An item name supported by the current conversation.

* `data`
  The data to send.

* `format`
  The format of the data.

* `timeout`
  The amount of time in milliseconds to wait for a response.

Remarks

This operation will timeout if the conversation is paused.

Exceptions

<table>
<thead>
<tr>
<th>Exception Type</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ArgumentException</td>
<td>This is thrown when item exceeds 255 characters or timeout is negative.</td>
</tr>
<tr>
<td>ArgumentNullException</td>
<td>This is thrown when item or data is a null reference.</td>
</tr>
<tr>
<td>InvalidOperationException</td>
<td>This is thrown when the client is not connected.</td>
</tr>
<tr>
<td>DdeException</td>
<td>This is thrown when the server</td>
</tr>
</tbody>
</table>
does not process the data.

See Also

DdeClient Class | NDde.Client Namespace | DdeClient.Poke
Overload List

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeClient.Poke Method (String, String, Int32)

This sends data to the server application.

```csharp
public virtual void Poke(
    String item,
    String data,
    Int32 timeout
);
```

**Parameters**

*item*
An item name supported by the current conversation.

*data*
The data to send.

*timeout*
The amount of time in milliseconds to wait for a response.

**Remarks**
This operation will timeout if the conversation is paused.

**Exceptions**

<table>
<thead>
<tr>
<th>Exception Type</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ArgumentException</td>
<td>This is thrown when item exceeds 255 characters or timeout is negative.</td>
</tr>
<tr>
<td>ArgumentNullException</td>
<td>This is thrown when item or data is a null reference.</td>
</tr>
<tr>
<td>InvalidOperationException</td>
<td>This is thrown when the client is not connected.</td>
</tr>
<tr>
<td>DdeException</td>
<td>This is thrown when the server does not process the data.</td>
</tr>
</tbody>
</table>

**See Also**
Dynamic Data Exchange Library for .NET
DdeClient.Request Method

Overload List

This requests data using the specified item name.

```csharp
public virtual string Request(String, Int32);
```

This requests data using the specified item name.

```csharp
public virtual byte[] Request(String, Int32, Int32);
```

See Also

[DdeClient Class] | [NDde.Client Namespace]

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
**DdeClient.Request Method (String, Int32)**

This requests data using the specified item name.

```csharp
public virtual string Request(
    String item,
    Int32 timeout
);
```

### Parameters

- **item**
  - An item name supported by the current conversation.

- **timeout**
  - The amount of time in milliseconds to wait for a response.

### Return Value

The data returned by the server application in CF_TEXT format.

### Remarks

This operation will timeout if the conversation is paused.

### Exceptions

<table>
<thead>
<tr>
<th>Exception Type</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ArgumentException</td>
<td>This is thrown when item exceeds 255 characters or timeout is negative.</td>
</tr>
<tr>
<td>ArgumentNullException</td>
<td>This is thrown when item is a null reference.</td>
</tr>
<tr>
<td>InvalidOperationException</td>
<td>This is thrown when the client is not connected.</td>
</tr>
<tr>
<td>DdeException</td>
<td>This is thrown when the server does not process the request.</td>
</tr>
</tbody>
</table>

### See Also
Dynamic Data Exchange Library for .NET
DdeClient.Request Method (String, Int32, Int32)

This requests data using the specified item name.

```csharp
public virtual byte[] Request(
    String item,
    Int32 format,
    Int32 timeout
);
```

Parameters

* item
  An item name supported by the current conversation.

* format
  The format of the data to return.

* timeout
  The amount of time in milliseconds to wait for a response.

Return Value

The data returned by the server application.

Remarks

This operation will timeout if the conversation is paused.

Exceptions

<table>
<thead>
<tr>
<th>Exception Type</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ArgumentException</td>
<td>This is thrown when item exceeds 255 characters or timeout is negative.</td>
</tr>
<tr>
<td>ArgumentNullException</td>
<td>This is thrown when item is a null reference.</td>
</tr>
<tr>
<td>InvalidOperationException</td>
<td>This is thrown when the client is not connected.</td>
</tr>
<tr>
<td>DdeException</td>
<td>This is thrown when the server</td>
</tr>
</tbody>
</table>
does not process the request.

See Also

DdeClient Class | NDde.Client Namespace | DdeClient.Request Overload List

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeClient.Resume Method

This resumes the current conversation.

```csharp
public virtual void Resume();
```

Exceptions

<table>
<thead>
<tr>
<th>Exception Type</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>InvalidOperationException</td>
<td>This is thrown when the conversation was not previously paused or when the client is not connected.</td>
</tr>
<tr>
<td>DdeException</td>
<td>This is thrown when the conversation could not be resumed.</td>
</tr>
</tbody>
</table>

See Also

[DdeClient Class](#) | [NDde.Client Namespace](#)

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
Overload List

This initiates an advise loop on the specified item name.

```csharp
public virtual void StartAdvise(String, Int32, Boolean, Boolean, Int32, Object);
```

This initiates an advise loop on the specified item name.

```csharp
public virtual void StartAdvise(String, Int32, Boolean, Int32);
```

See Also

[DdeClient Class](#) | [NDde.Client Namespace](#)

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
This initiates an advise loop on the specified item name.

```csharp
public virtual void StartAdvise(
    String item,
    Int32 format,
    Boolean hot,
    Boolean acknowledge,
    Int32 timeout,
    Object adviseState
);
```

**Parameters**

- **item**
  An item name supported by the current conversation.

- **format**
  The format of the data to return.

- **hot**
  A bool indicating whether data should be included with the notification.

- **acknowledge**
  A bool indicating whether the client should acknowledge each advisory before the server will send send another.

- **timeout**
  The amount of time in milliseconds to wait for a response.

- **adviseState**
  An application defined data object to associate with this advise loop.

**Remarks**

This operation will timeout if the conversation is paused.

**Events**
<table>
<thead>
<tr>
<th>Event Type</th>
<th>Reason</th>
</tr>
</thead>
<tbody>
<tr>
<td>Advise</td>
<td></td>
</tr>
</tbody>
</table>

### Exceptions

<table>
<thead>
<tr>
<th>Exception Type</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ArgumentException</strong></td>
<td>This is thrown when item exceeds 255 characters or timeout is negative.</td>
</tr>
<tr>
<td><strong>ArgumentNullException</strong></td>
<td>This is thrown when item is a null reference.</td>
</tr>
<tr>
<td><strong>InvalidOperationException</strong></td>
<td>This is thrown when the item is already being advised or when the client is not connected.</td>
</tr>
<tr>
<td><strong>DdeException</strong></td>
<td>This is thrown when the server does not initiate the advise loop.</td>
</tr>
</tbody>
</table>

**See Also**

[DdeClient Class](#) | [NDde.Client Namespace](#) | [DdeClient.StartAdvise Overload List](#)

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
This initiates an advise loop on the specified item name.

```csharp
public virtual void StartAdvise(
    String item,
    Int32 format,
    Boolean hot,
    Int32 timeout
);
```

**Parameters**

- **item**
  - An item name supported by the current conversation.

- **format**
  - The format of the data to return.

- **hot**
  - A bool indicating whether data should be included with the notification.

- **timeout**
  - The amount of time in milliseconds to wait for a response.

**Remarks**

This operation will timeout if the conversation is paused.

**Events**

<table>
<thead>
<tr>
<th>Event Type</th>
<th>Reason</th>
</tr>
</thead>
<tbody>
<tr>
<td>Advise</td>
<td></td>
</tr>
</tbody>
</table>

**Exceptions**

<table>
<thead>
<tr>
<th>Exception Type</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ArgumentException</td>
<td>This is thrown when item exceeds 255 characters or</td>
</tr>
<tr>
<td>Exception</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>IllegalArgumentException</td>
<td>This is thrown when item is a null reference.</td>
</tr>
<tr>
<td>InvalidOperationException</td>
<td>This is thrown when the item is already being advised or when the client is not connected.</td>
</tr>
<tr>
<td>DdeException</td>
<td>This is thrown when the server does not initiate the advise loop.</td>
</tr>
</tbody>
</table>

**See Also**

[DdeClient Class](#) | [NDde.Client Namespace](#) | [DdeClient.StartAdvise Overload List](#)

---

**Send comments on this topic.**

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeClient.StopAdvise Method

This terminates the advise loop for the specified item name.

```csharp
public virtual void StopAdvise(
    String item,
    Int32 timeout
);
```

Parameters

- `item`
  An item name that has an active advise loop.

- `timeout`
  The amount of time in milliseconds to wait for a response.

Remarks

This operation will timeout if the conversation is paused.

Exceptions

<table>
<thead>
<tr>
<th>Exception Type</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ArgumentException</td>
<td>This is thown when item exceeds 255 characters or timeout is negative.</td>
</tr>
<tr>
<td>ArgumentNullException</td>
<td>This is thrown when item is a null reference.</td>
</tr>
<tr>
<td>InvalidOperationException</td>
<td>This is thrown when the item is not being advised or when the client is not connected.</td>
</tr>
<tr>
<td>DdeException</td>
<td>This is thrown when the server does not terminate the advise loop.</td>
</tr>
</tbody>
</table>

See Also

DdeClient Class | NDde.Client Namespace
Dynamic Data Exchange Library for .NET
**DdeClient.TryConnect Method**

This establishes a conversation with a server that supports the specified service name and topic name pair.

```
public virtual int TryConnect();
```

**Return Value**

Zero if the operation succeed or non-zero if the operation failed.

**See Also**

[DdeClient Class][1] | [NDde.Client Namespace][2]

---

[1]: #/DdeClient Class
[2]: #/NDde.Client Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeClient.TryExecute Method

This sends a command to the server application.

```csharp
public virtual int TryExecute(
    String command,
    Int32 timeout
);
```

Parameters

- **command**
  - The command to be sent to the server application.

- **timeout**
  - The amount of time in milliseconds to wait for a response.

Return Value

Zero if the operation succeed or non-zero if the operation failed.

Remarks

This operation will timeout if the conversation is paused.

See Also

DdeClient Class | NDde.Client Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeClient.TryPoke Method

This sends data to the server application.

```csharp
public virtual int TryPoke(
    String item,
    Byte[] data,
    Int32 format,
    Int32 timeout
);
```

Parameters

- `item`  
  An item name supported by the current conversation.

- `data`  
  The data to send.

- `format`  
  The format of the data.

- `timeout`  
  The amount of time in milliseconds to wait for a response.

Return Value

Zero if the operation succeed or non-zero if the operation failed.

Remarks

This operation will timeout if the conversation is paused.

See Also

DdeClient Class | NDde.Client Namespace

---

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
**DdeClient.TryRequest Method**

This requests data using the specified item name.

```csharp
public virtual int TryRequest(
    String item,
    Int32 format,
    Int32 timeout,
    out Byte[]& data
);
```

**Parameters**

- **item**
  An item name supported by the current conversation.

- **format**
  The format of the data to return.

- **timeout**
  The amount of time in milliseconds to wait for a response.

- **data**
  The data returned by the server application.

**Return Value**

Zero if the operation succeeded or non-zero if the operation failed.

**Remarks**

This operation will timeout if the conversation is paused.

**See Also**

[DdeClient Class] | [NDde.Client Namespace]

---

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
**DdeClient Events**

The events of the **DdeClient** class are listed below. For a complete list of **DdeClient** class members, see the **DdeClient Members** topic.

### Public Instance Events

<table>
<thead>
<tr>
<th>Event</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Advise</td>
<td>This is raised when the data has changed for an item name that has an advise loop.</td>
</tr>
<tr>
<td>Disconnected</td>
<td>This is raised when the client has been disconnected.</td>
</tr>
</tbody>
</table>

See Also

[DdeClient Class](#) | [NDde.Client Namespace](#)

---

**Send comments on this topic.**

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
**DdeClient.EventHandler<TEventArgs> Event**

This is raised when the data has changed for an item name that has an advise loop.

```csharp
public event EventHandler<TEventArgs> Advise;
```

See Also

[DdeClient Class] | [NDde.Client Namespace]

---

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeClient.EventHandler<TEventArgs> Event

This is raised when the client has been disconnected.

public event EventHandler<TEventArgs> Disconnected;

See Also

DdeClient Class | NDde.Client Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
This contains information about the **Disconnected** event.

For a list of all members of this type, see [DdeDisconnectedEventArgs Members](#).

```csharp
public class DdeDisconnectedEventArgs : DdeEventArgs
```

### Thread Safety

Public static *(Shared in Visual Basic)* members of this type are safe for multithreaded operations. Instance members are **not** guaranteed to be thread-safe.

### Requirements

**Namespace**: [NDde.Client](#)

**Assembly**: NDde (in NDde.dll)

### See Also

[DdeDisconnectedEventArgs Members](#) | [NDde.Client Namespace](#)

---

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeDisconnectedEventArgs Members

**DdeDisconnectedEventArgs overview**

**Public Instance Properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>IsDisposed</strong></td>
<td>This gets a bool indicating whether the client disconnected because <code>Dispose</code> was explicitly called.</td>
</tr>
<tr>
<td><strong>IsServerInitiated</strong></td>
<td>This gets a bool indicating whether the client disconnected because of the server.</td>
</tr>
</tbody>
</table>

**Public Instance Methods**

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong> (inherited from Object)</td>
<td>Determines whether the specified <code>Object</code> is equal to the current <code>Object</code>.</td>
</tr>
<tr>
<td><strong>GetHashCode</strong> (inherited from Object)</td>
<td>Serves as a hash function for a particular type. <code>GetHashCode</code> is suitable for use in hashing algorithms and data structures like a hash table.</td>
</tr>
<tr>
<td><strong>GetType</strong> (inherited from Object)</td>
<td>Gets the <code>Type</code> of the current instance.</td>
</tr>
<tr>
<td><strong>ToString</strong> (inherited from DdeEventArgs)</td>
<td>This returns a string containing the current values of all properties.</td>
</tr>
</tbody>
</table>

**See Also**

[DdeDisconnectedEventArgs Class] | [NDde.Client Namespace]

---

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeDisconnectedEventArgs Properties

The properties of the **DdeDisconnectedEventArgs** class are listed below. For a complete list of **DdeDisconnectedEventArgs** class members, see the **DdeDisconnectedEventArgs Members** topic.

Public Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>IsDisposed</strong></td>
<td>This gets a bool indicating whether the client disconnected because <strong>Dispose</strong> was explicitly called.</td>
</tr>
<tr>
<td><strong>IsServerInitiated</strong></td>
<td>This gets a bool indicating whether the client disconnected because of the server.</td>
</tr>
</tbody>
</table>

See Also

[DdeDisconnectedEventArgs Class] | [NDde.Client Namespace]

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
This gets a bool indicating whether the client disconnected because Dispose was explicitly called.

```csharp
public Boolean IsDisposed {get;}
```

Remarks
The value will be true if Dispose was explicitly called on DdeClient. The DdeClient sending this event has been disposed and can no longer be accessed. Any exception thrown in the currently executing method will be ignored.

See Also
DdeDisconnectedEventArgs Class | NDde.Client Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
**DdeDisconnectedEventArgs.IsServerInitiated Property**

This gets a bool indicating whether the client disconnected because of the server.

```csharp
public Boolean IsServerInitiated {get;}
```

**See Also**

[DdeDisconnectedEventArgs Class] | [NDde.Client Namespace]

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
**NDde.Server Namespace**

This namespace contains classes for creating DDE server applications.

Namespace hierarchy

## Classes

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>DdeConversation</code></td>
<td>This represents a DDE conversation established on a <code>DdeServer</code>.</td>
</tr>
<tr>
<td><code>DdeServer</code></td>
<td>This represents the server side of DDE conversations.</td>
</tr>
</tbody>
</table>

## Structures

<table>
<thead>
<tr>
<th>Structure</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>DdeServer.ExecuteResult</code></td>
<td>This is the return value of the <code>OnExecute</code> method.</td>
</tr>
<tr>
<td><code>DdeServer.PokeResult</code></td>
<td>This is the return value of the <code>OnPoke</code> method.</td>
</tr>
<tr>
<td><code>DdeServer.RequestResult</code></td>
<td>This is the return value of the <code>OnRequest</code> method.</td>
</tr>
</tbody>
</table>

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Dynamic Data Exchange Library for .NET
This represents a DDE conversation established on a **DdeServer**. For a list of all members of this type, see **DdeConversation Members**.

**System.Object**  **DdeConversation**

```csharp
public class DdeConversation
```

**Thread Safety**

This type is safe for multithreaded operations.

**Requirements**

**Namespace:** [NDde.Server](https://redlinksoft.com/NDde)

**Assembly:** NDde (in NDde.dll)

**See Also**

[DdeConversation Members](https://redlinksoft.com/NDde#DdeConversationMembers) | [NDde.Server Namespace](https://redlinksoft.com/NDde#NDdeServerNamespace)

---

**Send comments on this topic.**

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
## DdeConversation Members

### DdeConversation overview

### Public Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Handle</strong></td>
<td>This gets the DDEML handle associated with this conversation.</td>
</tr>
<tr>
<td><strong>IsPaused</strong></td>
<td>This gets a bool indicating whether this conversation is paused.</td>
</tr>
<tr>
<td><strong>Service</strong></td>
<td>This gets the service name associated with this conversation.</td>
</tr>
<tr>
<td><strong>Tag</strong></td>
<td>This gets an application defined data object associated with this conversation.</td>
</tr>
<tr>
<td><strong>Topic</strong></td>
<td>This gets the topic name associated with this conversation.</td>
</tr>
</tbody>
</table>

### Public Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong> (inherited from Object)</td>
<td>Determines whether the specified <strong>Object</strong> is equal to the current <strong>Object</strong>.</td>
</tr>
<tr>
<td><strong>GetHashCode</strong> (inherited from Object)</td>
<td>Serves as a hash function for a particular type. <strong>GetHashCode</strong> is suitable for use in hashing algorithms and data structures like a hash table.</td>
</tr>
<tr>
<td><strong>GetType</strong> (inherited from Object)</td>
<td>Gets the <strong>Type</strong> of the current instance.</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>This returns a string containing the current values of all</td>
</tr>
</tbody>
</table>
See Also

DdeConversation Class | NDde.Server Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeConversation Properties

The properties of the DdeConversation class are listed below. For a complete list of DdeConversation class members, see the DdeConversation Members topic.

Public Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Handle</strong></td>
<td>This gets the DDEML handle associated with this conversation.</td>
</tr>
<tr>
<td><strong>IsPaused</strong></td>
<td>This gets a bool indicating whether this conversation is paused.</td>
</tr>
<tr>
<td><strong>Service</strong></td>
<td>This gets the service name associated with this conversation.</td>
</tr>
<tr>
<td><strong>Tag</strong></td>
<td>This gets an application defined data object associated with this conversation.</td>
</tr>
<tr>
<td><strong>Topic</strong></td>
<td>This gets the topic name associated with this conversation.</td>
</tr>
</tbody>
</table>

See Also

DdeConversation Class | NDde.Server Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
DdeConversation.Handle Property

This gets the DDEML handle associated with this conversation.

```csharp
public IntPtr Handle {get;}
```

Remarks

This can be used in any DDEML function requiring a conversation handle.

**CAUTION** Incorrect usage of the DDEML can cause this object to function incorrectly and can lead to resource leaks.

See Also

[DdeConversation Class] | [NDde.Server Namespace]

---

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeConversation.IsPaused Property

This gets a bool indicating whether this conversation is paused.

```
public Boolean IsPaused {get;}
```

See Also

DdeConversation Class | NDde.Server Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeConversation.Service Property

This gets the service name associated with this conversation.

```csharp
public String Service {get;}
```

See Also

- DdeConversation Class | NDde.Server Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
DdeConversation.Tag Property

This gets an application defined data object associated with this conversation.

```csharp
public Object Tag {get; set;}
```

Remarks

Use this property to carry state information with the conversation.

See Also

DdeConversation Class | NDde.Server Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
DdeConversation.Topic Property

This gets the topic name associated with this conversation.

```csharp
public String Topic {get;}
```

See Also

DdeConversation Class | NDde.Server Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeConversation Methods

The methods of the **DdeConversation** class are listed below. For a complete list of **DdeConversation** class members, see the **DdeConversation Members** topic.

Public Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>✷ <strong>Equals</strong> (inherited from <strong>Object</strong>)</td>
<td>Determines whether the specified <strong>Object</strong> is equal to the current <strong>Object</strong>.</td>
</tr>
<tr>
<td>✷ <strong>GetHashCode</strong> (inherited from <strong>Object</strong>)</td>
<td>Serves as a hash function for a particular type. <strong>GetHashCode</strong> is suitable for use in hashing algorithms and data structures like a hash table.</td>
</tr>
<tr>
<td>✷ <strong>GetType</strong> (inherited from <strong>Object</strong>)</td>
<td>Gets the <strong>Type</strong> of the current instance.</td>
</tr>
<tr>
<td>✷ <strong>ToString</strong></td>
<td>This returns a string containing the current values of all properties.</td>
</tr>
</tbody>
</table>

See Also

- [DdeConversation Class](#)  | [NDde.Server Namespace](#)

---

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeConversation.ToString Method

This returns a string containing the current values of all properties.

```
public override string ToString();
```

Return Value

A string containing the current values of all properties.

See Also

DdeConversation Class | NDde.Server Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeServer Class

This represents the server side of DDE conversations.
For a list of all members of this type, see DdeServer Members.

System.Object  DdeServer

public class DdeServer : IDisposable

Thread Safety

This type is safe for multithreaded operations.

Remarks

DDE conversations are established by specifying a service name and topic name pair. The service name is usually the name of the application acting as a DDE server. A DDE server can respond to multiple service names, but most servers usually only respond to one. The topic name is a logical context for data and is defined by the server application. A server can and usually does support many topic names.

After this object has registered its service name by calling the Register method clients can connect to it by specifying the service name the server registered and a topic name that it supports.

Event methods are invoked on the thread hosting the DdeContext. All operations must be marshaled onto the thread hosting the DdeContext associated with this object. Method calls will block until that thread becomes available. An exception will be generated if the thread does not become available in a timely manner.

Notes to Inheritors: The event methods must be overridden in a subclass as needed.

Example

The following example demostrates how to use a DdeServer.

[C#]

using System;
using System.Collections;
using System.Timers;
using NDde.Server;

public class Server
{
    public static void Main()
    {
        try
        {
            // Create a server that will register the service name 'myapp'.
            using (DdeServer server = new MyServer("myapp"))
            {
                // Register the service name.
                server.Register();

                // Wait for the user to press ENTER before proceeding.
                Console.WriteLine("Press ENTER to quit...");
                Console.ReadLine();
            }
        }
        catch (Exception e)
        {
            Console.WriteLine(e);
            Console.WriteLine("Press ENTER to quit...");
            Console.ReadLine();
        }
    }

    private sealed class MyServer : DdeServer
    {
        private System.Timers.Timer _Timer = new System.Timers.Timer();

        public MyServer(string service)
        {
            // Create a timer that will advise clients of new data.
            _Timer.Elapsed += this.OnTimerElapsed;
        }
    }
}
_Timer.Interval = 1000;
_Timer.SynchronizingObject = _Timer.Start();
}

private void OnTimerElapsed(object sender, ElapsedEventArgs args)
{
    // Advise all topic name and item name pairs.
    Advise("*", "*");
}

protected override bool OnBeforeConnect(string topic)
{
    Console.WriteLine("OnBeforeConnect:", topic);
    return true;
}

protected override void OnAfterConnect(DdeConversation conversation)
{
    Console.WriteLine("OnAfterConnect:", conversation);
}

protected override void OnDisconnect(DdeConversation conversation)
{
    Console.WriteLine("OnDisconnect:", conversation);
}

protected override bool OnStartAdvise(DdeConversation conversation, string item, int format)
{ Console.WriteLine("OnStartAdvise:");

    // Initiate the advisory loop only if the format is CF_TEXT.
    return format == 1;
}

protected override void OnStopAdvise(DdeConversation conversation, string item)
{
    Console.WriteLine("OnStopAdvise:");

}

protected override ExecuteResult OnExecute(DdeConversation conversation, string command)
{
    Console.WriteLine("OnExecute:");

    // Tell the client that the command was processed.
    return ExecuteResult.Processed;
}

protected override PokeResult OnPoke(DdeConversation conversation, string item, byte[] data, int format)
{
    Console.WriteLine("OnPoke:");

}
protected override RequestResult OnRequest(DdeConversation conversation, string item, int format)
{
    Console.WriteLine("OnRequest:");
    "+ " Service='" + conversation.Service + "'
    + " Topic='" + conversation.Topic + "'
    + " Handle='" + conversation.Handle.ToString() + "'
    + " Item='" + item + "'
    + " Data='" + data.Length.ToString() + "'
    + " Format='" + format.ToString());

    // Tell the client that the data was processed.
    return PokeResult.Processed;
}

protected override byte[] OnAdvise(string topic, string item, int format)
{
    Console.WriteLine("OnAdvise:");
    "+ " Service='" + this.Service + "'
    + " Topic='" + topic + "'
    + " Item='" + item + "'
    + " Format='" + format.ToString();

    // Send data to the client only if the format is CF_TEXT.
    if (format == 1)
    {
        return new RequestResult(System.Text.Encoding.ASCII.GetBytes("Time='" + DateTime.Now.ToString() + "\0"));
    }

    return RequestResult.NotProcessed;
}

protected override byte[] OnAdvise(string topic, string item, int format)
{
    Console.WriteLine("OnAdvise:");
    "+ " Service='" + this.Service + "'
    + " Topic='" + topic + "'
    + " Item='" + item + "'
    + " Format='" + format.ToString();

    // Send data to the client only if the format is CF_TEXT.
    if (format == 1)
    {
        return new RequestResult(System.Text.Encoding.ASCII.GetBytes("Time='" + DateTime.Now.ToString() + "\0"));
    }

    return RequestResult.NotProcessed;
}
```csharp
{   return System.Text.Encoding.ASCII.GetBytes("Time=" + DateTime.Now.ToString() + "\0");
}
return null;

} // class
} // class

[Visual Basic]
Imports NDde.Server

Module Program
    Sub Main()
        Try
            ' Create a server that will
            Using server As DdeServer =

                ' Register the service
                server.Register()

                ' Wait for the user to press
                Console.WriteLine("Press ENTER to quit...")
                Console.ReadLine()

            End Using

        Catch e As Exception
            Console.WriteLine(e)
            Console.WriteLine("Press ENTER to quit...")
            Console.ReadLine()
        End Try
    End Sub
End Module
```
Private Class MyServer
    Inherits DdeServer

    Private WithEvents theTimer As System.Timers.Timer

    Public Sub New(ByVal service As String)
        MyBase.New(service)
        ' Create a timer that will be used to advise clients of new data.
        theTimer.Interval = 1000
        theTimer.SynchronizingObject = Me.Context
        theTimer.Start()
    End Sub

    Private Sub theTimer_Elapsed(ByVal sender As Object, ByVal e As System.Timers.ElapsedEventArgs)
        Handles theTimer.Elapsed
        Me.Advise("*", "**")
    End Sub

    Protected Overrides Function OnBeforeConnect(ByVal topic As String) As Boolean
        Return True
    End Function

    Protected Overrides Sub OnAfterConnect(ByVal conversation As DdeConversation)
    End Sub
Protected Overrides Sub OnDiscon
    Console.WriteLine("OnDisconn" + " Service='" + conversation + " Topic='" + conversation + " Handle='" + conversation
End Sub

Protected Overrides Function OnStartAdvise(ByVal conversation As DdeConversation, ByVal item As String, ByVal format As Integer) As Boolean
    Console.WriteLine("OnStartAdvise": Service='" + conversation + " Topic='" + conversation + " Handle='" + conversation + " Item='" + item + ""
    Initiate the advisory loop only if the format is CF_TEXT.
    Return format = 1
End Function

Protected Overrides Sub OnStopAdvise(ByVal conversation As DdeConversation, ByVal item As String)
    Console.WriteLine("OnStopAdvise": Service='" + conversation + " Topic='" + conversation + " Handle='" + conversation + " Item='" + item + "")
End Sub

Protected Overrides Function OnExecute(ByVal conversation As DdeConversation, ByVal command As String) As ExecuteResult
    Console.WriteLine("OnExecute": Service='" + conversation + " Topic='" + conversation + " Handle='" + conversation + " Command='" + command + 
    ' Tell the client that the command was processed.
    Return ExecuteResult.Processed
End Function
Protected Overrides Function OnPoke(ByVal conversation As DdeConversation, ByVal item As String, ByVal data As Byte(), ByVal format As Integer) As PokeResult

    Console.WriteLine("OnPoke:").PadRight(16) + " Service='" + conversation.Service + "'"
    + " Topic='" + conversation.Topic + "'"
    + " Handle='" + conversation.Handle.ToString() + "'"
    + " Item='" + item + "'"
    + " Data='" + data.Length.ToString() + "'"
    + " Format='" + format.ToString() + "]"

    ' Tell the client that the data was processed.
    Return PokeResult.Processed
End Function

Protected Overrides Function OnRequest(ByVal conversation As DdeConversation, ByVal item As String, ByVal format As Integer) As RequestResult

    Console.WriteLine("OnRequest:").PadRight(16) + " Service='" + conversation.Service + "'"
    + " Topic='" + conversation.Topic + "'"
    + " Handle='" + conversation.Handle.ToString() + "'"
    + " Item='" + item + "'"
    + " Format='" + format.ToString() + "]"

    ' Return data to the client only if the format is CF_TEXT.
    If format = 1 Then
        Return New RequestResult(System.Text.Encoding.ASCII.GetBytes("Time=" + DateTime.Now.ToString() + Convert.ToChar(0)))
    End If
    Return RequestResult.NotProcessed
End Function

Protected Overrides Function OnAdvise(ByVal topic As String, ByVal item As String, ByVal format As Integer) As Byte()

    Console.WriteLine("OnAdvise:").PadRight(16) + " Service='" + Me.Service + "'"
    + " Topic='" + topic + "'"
    + " Item='" + item + "'"
    + " Format='" + format.ToString() + "]"

    ' Send data to the client on
If format = 1 Then
    Return System.Text.Encoding.ASCII.GetBytes("Time=\n    + DateTime.Now.ToString() + Convert.ToChar(0))
End If
Return Nothing
End Function
End Class
End Module

Requirements

Namespace: NDde.Server
Assembly: NDde (in NDde.dll)

See Also

DdeServer Members | NDde.Server Namespace
Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
# DdeServer Members

## DdeServer overview

### Public Instance Constructors

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DdeServer</td>
<td>Overloaded.</td>
</tr>
</tbody>
</table>

### Public Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Context</td>
<td>This gets the context associated with his instance.</td>
</tr>
<tr>
<td>IsRegistered</td>
<td>This gets a bool indicating whether the service name is registered.</td>
</tr>
<tr>
<td>Service</td>
<td>This gets the service name associated with this server.</td>
</tr>
</tbody>
</table>

### Public Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Advise</td>
<td>This notifies all clients that data has changed for the specified topic name and item name pair.</td>
</tr>
<tr>
<td>Disconnect</td>
<td>Overloaded.</td>
</tr>
<tr>
<td>Dispose</td>
<td>Overloaded. This unregisters service name and releases all resources held by this instance.</td>
</tr>
<tr>
<td>Equals</td>
<td>Determines whether the specified Object is equal to the current Object.</td>
</tr>
<tr>
<td>GetHashCode</td>
<td>Serves as a hash function for a particular type. GetHashCode is suitable for use in hashing algorithms and data structures like a hash table.</td>
</tr>
<tr>
<td>GetType</td>
<td>Gets the Type of the current instance.</td>
</tr>
</tbody>
</table>

(inherited from Object)
<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Pause</strong></td>
<td>Overloaded.</td>
</tr>
<tr>
<td><strong>Register</strong></td>
<td>This registers the service name.</td>
</tr>
<tr>
<td><strong>Resume</strong></td>
<td>Overloaded.</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>(inherited from Object) Returns a <code>String</code> that represents the current <code>Object</code>.</td>
</tr>
<tr>
<td><strong>Unregister</strong></td>
<td>This unregisters the service name.</td>
</tr>
</tbody>
</table>

**Protected Instance Methods**

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Dispose</strong></td>
<td>Overloaded. This contains the implementation to release all resources held by this instance.</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>(inherited from Object) Allows an <code>Object</code> to attempt to free resources and perform other cleanup operations before the <code>Object</code> is reclaimed by garbage collection.</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>(inherited from Object) Creates a shallow copy of the current <code>Object</code>.</td>
</tr>
<tr>
<td><strong>OnAdvise</strong></td>
<td>This is invoked when the server is performing a hot advise.</td>
</tr>
<tr>
<td><strong>OnAfterConnect</strong></td>
<td>This is invoked when a client has successfully established a conversation.</td>
</tr>
<tr>
<td><strong>OnBeforeConnect</strong></td>
<td>This is invoked when a client attempts to establish a conversation.</td>
</tr>
<tr>
<td><strong>OnDisconnect</strong></td>
<td>This is invoked when a client terminates a conversation.</td>
</tr>
<tr>
<td><strong>OnExecute</strong></td>
<td>This is invoked when a client sends a command.</td>
</tr>
<tr>
<td><strong>OnPoke</strong></td>
<td>This is invoked when a client...</td>
</tr>
<tr>
<td>Event</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>--------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>OnRequest</strong></td>
<td>Sends data. This is invoked when a client attempts to request data.</td>
</tr>
<tr>
<td><strong>OnStartAdvise</strong></td>
<td>This is invoked when a client attempts to initiate an advise loop.</td>
</tr>
<tr>
<td><strong>OnStopAdvise</strong></td>
<td>This is invoked when a client terminates an advise loop.</td>
</tr>
</tbody>
</table>

**See Also**

[DdeServer Class] | [NDde.Server Namespace]

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeServer Constructor

Overload List

This initializes a new instance of the DdeServer class that can register the specified service name.

public DdeServer(String);

This initializes a new instance of the DdeServer class that can register the specified service name and uses the specified context.

public DdeServer(String,DdeContext);

This initializes a new instance of the DdeServer class that can register the specified service name and using the specified synchronizing object.

public DdeServer(String,ISynchronizeInvoke);

See Also

DdeServer Class  |  NDde.Server Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeServer Constructor (String)

This initializes a new instance of the DdeServer class that can register the specified service name.

```java
public DdeServer(String service);
```

Parameters

service

The service name that this instance can register.

Exceptions

<table>
<thead>
<tr>
<th>Exception Type</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ArgumentException</td>
<td>This is thrown when service exceeds 255 characters.</td>
</tr>
<tr>
<td>ArgumentNullException</td>
<td>This is thrown when service is a null reference.</td>
</tr>
</tbody>
</table>

See Also

DdeServer Class | NDde.Server Namespace | DdeServer Constructor Overload List

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeServer Constructor (String, ISynchronizeInvoke)

This initializes a new instance of the `DdeServer` class that can register the specified service name and using the specified synchronizing object.

```java
public DdeServer(
    String service,
    ISynchronizeInvoke synchronizingObject
);
```

Parameters

- **service**
  - The service name that this instance can register.

- **synchronizingObject**
  - The synchronizing object to use for this instance.

Exceptions

<table>
<thead>
<tr>
<th>Exception Type</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ArgumentException</td>
<td>This is thrown when service exceeds 255 characters.</td>
</tr>
<tr>
<td>ArgumentNullException</td>
<td>This is thrown when service is a null reference.</td>
</tr>
</tbody>
</table>

See Also

- `DdeServer Class`  |  `NDde.Server Namespace`  |  `DdeServer Constructor Overload List`  

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeServer Constructor (String, DdeContext)

This initializes a new instance of the `DdeServer` class that can register the specified service name and uses the specified context.

```java
public DdeServer(
    String service,
    DdeContext context
);
```

Parameters

- **service**
  The service name that this instance can register.

- **context**
  The context to use for execution.

Exceptions

<table>
<thead>
<tr>
<th>Exception Type</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ArgumentException</td>
<td>This is thrown when service exceeds 255 characters..</td>
</tr>
<tr>
<td>ArgumentNullException</td>
<td>This is thrown when service is a null reference.</td>
</tr>
</tbody>
</table>

See Also

- [DdeServer Class](#) | [NDde.Server Namespace](#) | [DdeServer Constructor Overload List](#)

---

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeServer Properties

The properties of the **DdeServer** class are listed below. For a complete list of **DdeServer** class members, see the **DdeServer Members** topic.

Public Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Context</strong></td>
<td>This gets the context associated with this instance.</td>
</tr>
<tr>
<td><strong>IsRegistered</strong></td>
<td>This gets a bool indicating whether the service name is registered.</td>
</tr>
<tr>
<td><strong>Service</strong></td>
<td>This gets the service name associated with this server.</td>
</tr>
</tbody>
</table>

See Also

[DdeServer Class](#) | [NDde.Server Namespace](#)

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeServer.Context Property

This gets the context associated with this instance.

```csharp
public DdeContext Context {get; set;}
```

See Also

DdeServer Class | NDde.Server Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeServer.IsRegistered Property

This gets a bool indicating whether the service name is registered.

```csharp
public virtual Boolean IsRegistered {get;}
```

See Also

[DdeServer Class] | [NDde.Server Namespace]

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeServer.Service Property

This gets the service name associated with this server.

```csharp
public virtual String Service {get; set;}
```

See Also

[DdeServer Class] | [NDde.Server Namespace]

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
## DdeServer Methods

The methods of the **DdeServer** class are listed below. For a complete list of **DdeServer** class members, see the **DdeServer Members** topic.

### Public Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Advise</strong></td>
<td>This notifies all clients that data has changed for the specified topic name and item name pair.</td>
</tr>
<tr>
<td><strong>Disconnect</strong></td>
<td>Overloaded.</td>
</tr>
<tr>
<td><strong>Dispose</strong></td>
<td>Overloaded. This unregisters service name and releases all resources held by this instance.</td>
</tr>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <strong>Object</strong> is equal to the current <strong>Object</strong>.</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. <strong>GetHashCode</strong> is suitable for use in hashing algorithms and data structures like a hash table.</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <strong>Type</strong> of the current instance.</td>
</tr>
<tr>
<td><strong>Pause</strong></td>
<td>Overloaded.</td>
</tr>
<tr>
<td><strong>Register</strong></td>
<td>This registers the service name.</td>
</tr>
<tr>
<td><strong>Resume</strong></td>
<td>Overloaded.</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <strong>String</strong> that represents the current <strong>Object</strong>.</td>
</tr>
<tr>
<td><strong>Unregister</strong></td>
<td>This unregisters the service name.</td>
</tr>
</tbody>
</table>

### Protected Instance Methods
<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Dispose</strong></td>
<td>Overloaded. This contains the implementation to release all resources held by this instance.</td>
</tr>
<tr>
<td><strong>Finalize</strong> (inherited from <strong>Object</strong>)</td>
<td>Allows an <strong>Object</strong> to attempt to free resources and perform other cleanup operations before the <strong>Object</strong> is reclaimed by garbage collection.</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong> (inherited from <strong>Object</strong>)</td>
<td>Creates a shallow copy of the current <strong>Object</strong>.</td>
</tr>
<tr>
<td><strong>OnAdvise</strong></td>
<td>This is invoked when the server is performing a hot advise.</td>
</tr>
<tr>
<td><strong>OnAfterConnect</strong></td>
<td>This is invoked when a client has successfully established a conversation.</td>
</tr>
<tr>
<td><strong>OnBeforeConnect</strong></td>
<td>This is invoked when a client attempts to establish a conversation.</td>
</tr>
<tr>
<td><strong>OnDisconnect</strong></td>
<td>This is invoked when a client terminates a conversation.</td>
</tr>
<tr>
<td><strong>OnExecute</strong></td>
<td>This is invoked when a client sends a command.</td>
</tr>
<tr>
<td><strong>OnPoke</strong></td>
<td>This is invoked when a client sends data.</td>
</tr>
<tr>
<td><strong>OnRequest</strong></td>
<td>This is invoked when a client attempts to request data.</td>
</tr>
<tr>
<td><strong>OnStartAdvise</strong></td>
<td>This is invoked when a client attempts to initiate an advise loop.</td>
</tr>
<tr>
<td><strong>OnStopAdvise</strong></td>
<td>This is invoked when a client terminates an advise loop.</td>
</tr>
</tbody>
</table>

**See Also**
Dynamic Data Exchange Library for .NET
DdeServer.Advise Method

This notifies all clients that data has changed for the specified topic name and item name pair.

```csharp
public virtual void Advise(
    String topic,
    String item
);
```

Parameters

- **topic**
  A topic name supported by this server.

- **item**
  An item name supported by this server.

Remarks

Use an asterix to indicate that the topic name, item name, or both should be wild.

Exceptions

<table>
<thead>
<tr>
<th>Exception Type</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ArgumentException</td>
<td>This is thrown when topic or item exceeds 255 characters..</td>
</tr>
<tr>
<td>ArgumentNullException</td>
<td>This is thrown when topic or item is a null reference.</td>
</tr>
<tr>
<td>InvalidOperationException</td>
<td>This is thrown when the server is not registered.</td>
</tr>
<tr>
<td>DdeException</td>
<td>This is thrown when the notification could not be posted.</td>
</tr>
</tbody>
</table>

See Also

DdeServer Class | NDde.Server Namespace
Dynamic Data Exchange Library for .NET
DdeServer.Disconnect Method

Overload List

This terminates all conversations.

```csharp
public virtual void Disconnect();
```

This terminates the specified conversation.

```csharp
public virtual void Disconnect(DdeConversation);
```

See Also

[DdeServer Class] | [NDde.Server Namespace]

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeServer.Disconnect Method ()

This terminates all conversations.

```
public virtual void Disconnect();
```

Exceptions

<table>
<thead>
<tr>
<th>Exception Type</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>InvalidOperationException</td>
<td>This is thrown when the server is not registered.</td>
</tr>
<tr>
<td>DdeException</td>
<td>This is thrown when the conversations could not be terminated.</td>
</tr>
</tbody>
</table>

See Also

DdeServer Class | NDde.Server Namespace | DdeServer.Disconnect Overload List

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
This terminates the specified conversation.

```csharp
public virtual void Disconnect(
    DdeConversation conversation);
```

Parameters

*conversation*

The conversation to terminate.

Exceptions

<table>
<thead>
<tr>
<th>Exception Type</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ArgumentNullException</td>
<td>This is thrown when conversation is a null reference.</td>
</tr>
<tr>
<td>InvalidOperationException</td>
<td>This is thrown when the server is not registered.</td>
</tr>
<tr>
<td>DdeException</td>
<td>This is thrown when the conversation could not be terminated.</td>
</tr>
</tbody>
</table>

See Also

[DdeServer Class] | NDde.Server Namespace | DdeServer.Disconnect Overload List

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
**Dynamic Data Exchange Library for .NET**
DdeServer.Dispose Method

This unregisters service name and releases all resources held by this instance.

Overload List

This unregisters service name and releases all resources held by this instance.

   public void Dispose();

This contains the implementation to release all resources held by this instance.

   protected virtual void Dispose(Boolean);

See Also

   DdeServer Class | NDde.Server Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
This unregisters service name and releases all resources held by this instance.

```csharp
public void Dispose();
```

Implements

```csharp
IDisposable.
```

See Also

[DdeServer Class] | [NDde.Server Namespace] | [DdeServer.Dispose Overload List]

[Send comments on this topic.]

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeServer.Dispose Method (Boolean)

This contains the implementation to release all resources held by this instance.

```csharp
protected virtual void Dispose(
    Boolean disposing
);
```

Parameters

- **disposing**
  
  True if called by Dispose, false otherwise.

See Also

DdeServer Class | NDde.Server Namespace | DdeServer.Dispose Overload List

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeServer.OnAdvise Method

This is invoked when the server is performing a hot advise.

```csharp
protected virtual byte[] OnAdvise(
            String topic,
            String item,
            Int32 format
        );
```

Parameters

- `topic`
  - The topic name associated with this event.

- `item`
  - The item name associated with this event.

- `format`
  - The format of the data.

Return Value

- The data that will be sent to the clients.

Remarks

- The default implementation sends nothing to the clients.

See Also

- DdeServer Class | NDde.Server Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeServer.OnAfterConnect Method

This is invoked when a client has successfully established a conversation.

```csharp
protected virtual void OnAfterConnect(
    DdeConversation conversation);
```

Parameters

*conversation*

The conversation associated with this event.

See Also

DdeServer Class | NDde.Server Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
# DdeServer.OnBeforeConnect Method

This is invoked when a client attempts to establish a conversation.

```csharp
protected virtual bool OnBeforeConnect(String topic);
```

## Parameters

- **topic**
  - The topic name associated with this event.

## Return Value

- True to allow the connection, false otherwise.

## Remarks

The default implementation accepts all connections.

## See Also

- [DdeServer Class](#) | [NDde.Server Namespace](#)

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
This is invoked when a client terminates a conversation.

```csharp
protected virtual void OnDisconnect(
    DdeConversation conversation);
```

**Parameters**

*conversation*

The conversation associated with this event.

**See Also**

[DdeServer Class](#) | [NDde.Server Namespace](#)

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeServer.OnExecute Method

This is invoked when a client sends a command.

```csharp
protected virtual ExecuteResult OnExecute(DdeConversation conversation, String command);
```

Parameters

- `conversation`  
  The conversation associated with this event.

- `command`  
  The command to be executed.

Return Value

An `ExecuteResult` indicating the result.

Remarks

The default implementation returns `ExecuteResult.NotProcessed` to the client.

See Also

DdeServer Class | NDde.Server Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
**DdeServer.OnPoke Method**

This is invoked when a client sends data.

```csharp
protected virtual PokeResult OnPoke(
    DdeConversation conversation,
    String item,
    Byte[] data,
    Int32 format
);
```

**Parameters**

- **conversation**
  The conversation associated with this event.

- **item**
  The item name associated with this event.

- **data**
  The data associated with this event.

- **format**
  The format of the data.

**Return Value**

A `PokeResult` indicating the result.

**Remarks**

The default implementation returns `PokeResult.NotProcessed` to the client.

**See Also**

[DdeServer Class] | [NDde.Server Namespace]

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeServer.OnRequest Method

This is invoked when a client attempts to request data.

```csharp
protected virtual RequestResult OnRequest(
    DdeConversation conversation,
    String item,
    Int32 format
);
```

Parameters

- `conversation` The conversation associated with this event.
- `item` The item name associated with this event.
- `format` The format of the data.

Return Value

A `RequestResult` indicating the result.

Remarks

The default implementation returns `RequestResult.NotProcessed` to the client.

See Also

- [DdeServer Class](#) | [NDde.Server Namespace](#)

---

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
DdeServer.OnStartAdvise Method

This is invoked when a client attempts to initiate an advise loop.

```csharp
protected virtual bool OnStartAdvise(
    DdeConversation conversation,
    String item,
    Int32 format
);
```

Parameters

- **conversation**
  - The conversation associated with this event.

- **item**
  - The item name associated with this event.

- **format**
  - The format of the data.

Return Value

- True to allow the advise loop, false otherwise.

Remarks

- The default implementation accepts all advise loops.

See Also

- [DdeServer Class](#) | [NDde.Server Namespace](#)
Dynamic Data Exchange Library for .NET
DdeServer.OnStopAdvise Method

This is invoked when a client terminates an advise loop.

```csharp
protected virtual void OnStopAdvise(
    DdeConversation conversation,
    String item
);
```

Parameters

*conversation*
  The conversation associated with this event.

*item*
  The item name associated with this event.

See Also

DdeServer Class | NDde.Server Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeServer.Pause Method

Overload List

This pauses all conversations.

```csharp
public virtual void Pause();
```

This pauses the specified conversation.

```csharp
public virtual void Pause(DdeConversation);
```

Remarks

Pausing a conversation causes this server to queue events until the conversation resumes.

See Also

[DdeServer Class] | NDde.Server Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
This pauses all conversations.

```csharp
public virtual void Pause();
```

Remarks
Pausing a conversation causes this object to queue events until the conversation resumes.

Exceptions

<table>
<thead>
<tr>
<th>Exception Type</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>InvalidOperationException</td>
<td>This is thrown when the server is not registered.</td>
</tr>
<tr>
<td>DdeException</td>
<td>This is thrown when the conversations could not be paused.</td>
</tr>
</tbody>
</table>

See Also

[DdeServer Class](#) | [NDde.Server Namespace](#) | [DdeServer.Pause](#) | Overload List

---

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeServer.Pause Method (DdeConversation)

This pauses the specified conversation.

```csharp
public virtual void Pause(DdeConversation conversation);
```

Parameters

- `conversation` The conversation to pause.

Exceptions

<table>
<thead>
<tr>
<th>Exception Type</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ArgumentException</td>
<td>This is thrown when conversation is a null reference.</td>
</tr>
<tr>
<td>InvalidOperationException</td>
<td>This is thrown when the conversation is already paused or when the server is not registered.</td>
</tr>
<tr>
<td>DdeException</td>
<td>This is thrown when the conversation could not be paused.</td>
</tr>
</tbody>
</table>

See Also

- DdeServer Class
- NDde.Server Namespace
- DdeServer.Pause
- Overload List

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeServer.Register Method

This registers the service name.

```csharp
public virtual void Register();
```

Exceptions

<table>
<thead>
<tr>
<th>Exception Type</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>InvalidOperationException</td>
<td>This is thrown when the server is already registered.</td>
</tr>
<tr>
<td>DdeException</td>
<td>This is thrown when the service name could not be registered.</td>
</tr>
</tbody>
</table>

See Also

DdeServer Class | NDde.Server Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
**DdeServer.Resume Method**

**Overload List**

This resumes all conversations.

```csharp
public virtual void Resume();
```

This resumes the specified conversation.

```csharp
public virtual void Resume(DdeConversation);
```

See Also

[DdeServer Class] | [NDde.Server Namespace]

---

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeServer.Resume Method ()

This resumes all conversations.

```csharp
public virtual void Resume();
```

Exceptions

<table>
<thead>
<tr>
<th>Exception Type</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>InvalidOperationException</td>
<td>This is thrown when the server is not registered.</td>
</tr>
<tr>
<td>DdeException</td>
<td>This is thrown when the conversations could not be resumed.</td>
</tr>
</tbody>
</table>

See Also

- [DdeServer Class](#)
- [NDde.Server Namespace](#)
- [DdeServer.Resume Overload List](#)

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeServer.Resume Method (DdeConversation)

This resumes the specified conversation.

```csharp
public virtual void Resume(
    DdeConversation conversation
);
```

Parameters

*conversation*

The conversation to resume.

Exceptions

<table>
<thead>
<tr>
<th>Exception Type</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ArgumentException</td>
<td>This is thrown when conversation is a null reference.</td>
</tr>
<tr>
<td>InvalidOperationException</td>
<td>This is thrown when the conversation is not paused or when the server is not registered.</td>
</tr>
<tr>
<td>DdeException</td>
<td>This is thrown when the conversation could not be resumed.</td>
</tr>
</tbody>
</table>

See Also

DdeServer Class | NDde.Server Namespace | DdeServer.Resume Overload List

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeServer.Unregister Method

This unregisters the service name.

```csharp
public virtual void Unregister();
```

Exceptions

<table>
<thead>
<tr>
<th>Exception Type</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>InvalidOperationException</code></td>
<td>This is thrown when the server is not registered.</td>
</tr>
</tbody>
</table>

See Also

[DdeServer Class] | [NDde.Server Namespace]

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
This is the return value of the `OnExecute` method.

For a list of all members of this type, see `DdeServer.ExecuteResult Members`.

**System.Object**  **Value Type**  **DdeServer.ExecuteResult**

### Thread Safety

Public static (`Shared` in Visual Basic) members of this type are safe for multithreaded operations. Instance members are **not** guaranteed to be thread-safe.

### Requirements

**Namespace**: `NDde.Server`  
**Assembly**: NDde (in NDde.dll)

### See Also

[DdeServer.ExecuteResult Members] | [NDde.Server Namespace]

[Send comments on this topic.]  
Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)  
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
# DdeServer.ExecuteResult Members

## DdeServer.ExecuteResult overview

### Public Static Fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>NotProcessed</strong></td>
<td>Return this value if the command was not executed successfully.</td>
</tr>
<tr>
<td><strong>PauseConversation</strong></td>
<td>Return this value to pause the conversation and execute the command asynchronously. After the conversation has been resumed the OnExecute method will run again.</td>
</tr>
<tr>
<td><strong>Processed</strong></td>
<td>Return this value if the command was executed successfully.</td>
</tr>
<tr>
<td><strong>TooBusy</strong></td>
<td>Return this value if the server is too busy.</td>
</tr>
</tbody>
</table>

### Public Static Operators

<table>
<thead>
<tr>
<th>Operator</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equality Operator</strong></td>
<td>This determines whether two ExecuteResult objects are equal.</td>
</tr>
<tr>
<td><strong>Inequality Operator</strong></td>
<td>This determines whether two ExecuteResult objects are not equal.</td>
</tr>
</tbody>
</table>

### Public Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong></td>
<td>This determines whether two object instances are equal.</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>This returns a hash code for the object.</td>
</tr>
<tr>
<td><strong>GetType</strong> (inherited from <strong>Object</strong>)</td>
<td>Gets the Type of the current object.</td>
</tr>
<tr>
<td>Object)ToString (inherited from ValueType)</td>
<td>Returns the fully qualified type name of this instance.</td>
</tr>
</tbody>
</table>

**See Also**

*DdeServer.ExecuteResult Structure* | *NDde.Server Namespace*

*Send comments on this topic.*

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeServer.ExecuteResult Fields

The fields of the **DdeServer.ExecuteResult** structure are listed below. For a complete list of **DdeServer.ExecuteResult** structure members, see the [DdeServer.ExecuteResult Members](#) topic.

### Public Static Fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>NotProcessed</code></td>
<td>Return this value if the command was not executed successfully.</td>
</tr>
<tr>
<td><code>PauseConversation</code></td>
<td>Return this value to pause the conversation and execute the command asynchronously. After the conversation has been resumed the <strong>OnExecute</strong> method will run again.</td>
</tr>
<tr>
<td><code>Processed</code></td>
<td>Return this value if the command was executed successfully.</td>
</tr>
<tr>
<td><code>TooBusy</code></td>
<td>Return this value if the server is too busy.</td>
</tr>
</tbody>
</table>

### See Also

[DdeServer.ExecuteResult Structure](#) | [NDde.Server Namespace](#)

---

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeServer.ExecuteResult.NotProcessed Field

Return this value if the command was not executed successfully.

```csharp
public static readonly NotProcessed NotProcessed;
```

See Also

[DdeServer.ExecuteResult Class] [NDde.Server Namespace]

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeServer.ExecuteResult.PauseConversation Field

Return this value to pause the conversation and execute the command asynchronously. After the conversation has been resumed the OnExecute method will run again.

```csharp
public static readonly PauseConversation PauseConversation;
```

See Also

DdeServer.ExecuteResult Class  |  NDde.Server Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeServer.ExecuteResult.Processed Field

Return this value if the command was executed successfully.

```csharp
public static readonly Processed Processed;
```

See Also

DdeServer.ExecuteResult Class | NDde.Server Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeServer.ExecuteResult.TooBusy Field

Return this value if the server is too busy.

```csharp
public static readonly TooBusy TooBusy;
```

See Also

DdeServer.ExecuteResult Class | NDde.Server Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
The methods of the `DdeServer.ExecuteResult` structure are listed below. For a complete list of `DdeServer.ExecuteResult` structure members, see the `DdeServer.ExecuteResult Members` topic.

**Public Instance Methods**

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>Equals</code></td>
<td>This determines whether two object instances are equal.</td>
</tr>
<tr>
<td><code>GetHashCode</code></td>
<td>This returns a hash code for the object.</td>
</tr>
<tr>
<td><code>GetType</code> (inherited from <code>Object</code>)</td>
<td>Gets the <code>Type</code> of the current instance.</td>
</tr>
<tr>
<td><code>ToString</code> (inherited from <code>ValueType</code>)</td>
<td>Returns the fully qualified type name of this instance.</td>
</tr>
</tbody>
</table>

See Also

- [DdeServer.ExecuteResult Structure](#) | [NDde.Server Namespace](#)

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeServer.ExecuteResult.Equals Method

This determines whether two object instances are equal.

public override bool Equals(
    Object o
);

Parameters

  o
  The object to compare with the current object.

Return Value

  True if the specified object is equal to the current object, false otherwise.

See Also

  DdeServer.ExecuteResult Structure | NDde.Server Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeServer.ExecuteResult.GetHashCode Method

This returns a hash code for the object.

```csharp
public override int GetHashCode();
```

Return Value

A hash code for the object.

See Also

DdeServer.ExecuteResult Structure | NDde.Server Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeServer.ExecuteResult Operators

The operators of the DdeServer.ExecuteResult structure are listed below. For a complete list of DdeServer.ExecuteResult structure members, see the DdeServer.ExecuteResult Members topic.

Public Static Operators

<table>
<thead>
<tr>
<th>Operator</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>==</code> s Equality Operator</td>
<td>This determines whether two ExecuteResult objects are equal.</td>
</tr>
<tr>
<td><code>!=</code> s Inequality Operator</td>
<td>This determines whether two ExecuteResult objects are not equal.</td>
</tr>
</tbody>
</table>

See Also

DdeServer.ExecuteResult Structure | DdeServer.ExecuteResult Members | NDde.Server Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
DdeServer.ExecuteResult Equality Operator

This determines whether two `ExecuteResult` objects are equal.

```
public static bool operator ==(
    DdeServer.ExecuteResult lhs,
    DdeServer.ExecuteResult rhs
);
```

Parameters

- `lhs`  
  The left hand side object.

- `rhs`  

Return Value

True if the two objects are equal, false otherwise.

See Also

- [DdeServer.ExecuteResult Structure](#) | [NDde.Server Namespace](#)

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeServer.ExecuteResult Inequality Operator

This determines whether two ExecuteResult objects are not equal.

```csharp
public static bool operator !=(
    DdeServer.ExecuteResult lhs,
    DdeServer.ExecuteResult rhs
);
```

Parameters

- **lhs**
  - The left hand side object.

- **rhs**

Return Value

True if the two objects are not equal, false otherwise.

See Also

DdeServer.ExecuteResult Structure | NDde.Server Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
This is the return value of the `OnPoke` method.

For a list of all members of this type, see `DdeServer.PokeResult Members`.

**System.Object**  **ValueType**  **DdeServer.PokeResult**

### Thread Safety

Public static (`Shared` in Visual Basic) members of this type are safe for multithreaded operations. Instance members are **not** guaranteed to be thread-safe.

### Requirements

**Namespace:** `NDde.Server`  
**Assembly:** NDde (in NDde.dll)

### See Also

- `DdeServer.PokeResult Members`  |  `NDde.Server Namespace`

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)  
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
# DdeServer.PokeResult Members

## DdeServer.PokeResult overview

### Public Static Fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>✨ NotProcessed</td>
<td>Return this value if the poke was not successful.</td>
</tr>
<tr>
<td>✨ PauseConversation</td>
<td>Return this value to pause the conversation and execute the poke asynchronously. After the conversation has been resumed the OnPoke method will run again.</td>
</tr>
<tr>
<td>✨ Processed</td>
<td>Return this value if the poke was successful.</td>
</tr>
<tr>
<td>✨ TooBusy</td>
<td>Return this value if the server is too busy.</td>
</tr>
</tbody>
</table>

### Public Static Operators

<table>
<thead>
<tr>
<th>Operator</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>✨ Equality Operator</td>
<td>This determines whether two PokeResult objects are equal.</td>
</tr>
<tr>
<td>✨ Inequality Operator</td>
<td>This determines whether two ExecuteResult objects are not equal.</td>
</tr>
</tbody>
</table>

### Public Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>✨ Equals</td>
<td>This determines whether two object instances are equal.</td>
</tr>
<tr>
<td>✨ GetHashCode</td>
<td>This returns a hash code for the object.</td>
</tr>
<tr>
<td>✨ GetType</td>
<td>Gets the Type of the current instance.</td>
</tr>
<tr>
<td>✨ ToString</td>
<td>Returns the fully qualified type</td>
</tr>
</tbody>
</table>
ValueType) name of this instance.

See Also

DdeServer.PokeResult Structure | NDde.Server Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
The fields of the `DdeServer.PokeResult` structure are listed below. For a complete list of `DdeServer.PokeResult` structure members, see the `DdeServer.PokeResult Members` topic.

**Public Static Fields**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>NotProcessed</code></td>
<td>Return this value if the poke was not successful.</td>
</tr>
<tr>
<td><code>PauseConversation</code></td>
<td>Return this value to pause the conversation and execute the poke asynchronously. After the conversation has been resumed, the <code>OnPoke</code> method will run again.</td>
</tr>
<tr>
<td><code>Processed</code></td>
<td>Return this value if the poke was successful.</td>
</tr>
<tr>
<td><code>TooBusy</code></td>
<td>Return this value if the server is too busy.</td>
</tr>
</tbody>
</table>

**See Also**

[DdeServer.PokeResult Structure](#) | [NDde.Server Namespace](#)

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
**DdeServer.PokeResult.NotProcessed Field**

Return this value if the poke was not successful.

```java
public static readonly NotProcessed NotProcessed;
```

**See Also**

[DdeServer.PokeResult Class](#) | [NDde.Server Namespace](#)

---

**Send comments on this topic.**

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Return this value to pause the conversation and execute the poke asynchronously. After the conversation has been resumed the OnPoke method will run again.

```csharp
public static readonly PauseConversation;
```

See Also

[DdeServer.PokeResult Class](#) | [NDde.Server Namespace](#)

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
DdeServer.PokeResult.Processed Field

Return this value if the poke was successful.

```csharp
public static readonly Processed Processed;
```

See Also

DdeServer.PokeResult Class | NDde.Server Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeServer.PokeResult.TooBusy Field

Return this value if the server is too busy.

public static readonly TooBusy TooBusy;

See Also

DdeServer.PokeResult Class | NDde.Server Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeServer.PokeResult Methods

The methods of the DdeServer.PokeResult structure are listed below. For a complete list of DdeServer.PokeResult structure members, see the DdeServer.PokeResult Members topic.

Public Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Equals</td>
<td>This determines whether two object instances are equal.</td>
</tr>
<tr>
<td>GetHashCode</td>
<td>This returns a hash code for the object.</td>
</tr>
<tr>
<td>GetType (inherited from Object)</td>
<td>Gets the Type of the current instance.</td>
</tr>
<tr>
<td>ToString (inherited from ValueType)</td>
<td>Returns the fully qualified type name of this instance.</td>
</tr>
</tbody>
</table>

See Also

DdeServer.PokeResult Structure | NDde.Server Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeServer.PokeResult.Equals Method

This determines whether two object instances are equal.

```csharp
public override bool Equals(
    Object o
);
```

Parameters

- `o`: The object to compare with the current object.

Return Value

True if the specified object is equal to the current object, false otherwise.

See Also

DdeServer.PokeResult Structure | NDde.Server Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]

This returns a hash code for the object.

```
public override int GetHashCode();
```

Return Value

A hash code for the object.

See Also

DdeServer.PokeResult Structure | NDde.Server Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
DdeServer.PokeResult Operators

The operators of the `DdeServer.PokeResult` structure are listed below. For a complete list of `DdeServer.PokeResult` structure members, see the `DdeServer.PokeResult Members` topic.

Public Static Operators

<table>
<thead>
<tr>
<th>Operator</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>==</code></td>
<td><strong>Equality Operator</strong> This determines whether two <code>PokeResult</code> objects are equal.</td>
</tr>
<tr>
<td><code>!=</code></td>
<td><strong>Inequality Operator</strong> This determines whether two <code>ExecuteResult</code> objects are not equal.</td>
</tr>
</tbody>
</table>

See Also

[DdeServer.PokeResult Structure] | [DdeServer.PokeResult Members] | [NDde.Server Namespace]

---

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeServer.PokeResult Equality Operator

This determines whether two PokeResult objects are equal.

```csharp
public static bool operator ==(DdeServer.PokeResult lhs, DdeServer.PokeResult rhs);
```

**Parameters**

- `lhs`  
  The left hand side object.

- `rhs`  

**Return Value**

True if the two objects are equal, false otherwise.

**See Also**

- DdeServer.PokeResult Structure  |  NDde.Server Namespace

---

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
This determines whether two `ExecuteResult` objects are not equal.

```csharp
public static bool operator !=(
    DdeServer.PokeResult lhs,
    DdeServer.PokeResult rhs
);
```

**Parameters**

- `lhs`
  - The left hand side object.
- `rhs`

**Return Value**

True if the two objects are not equal, false otherwise.

**See Also**

[DdeServer.PokeResult Structure](#) | [NDde.Server Namespace](#)

[Send comments on this topic.](#)

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
This is the return value of the `OnRequest` method.

For a list of all members of this type, see `DdeServer.RequestResult Members`.

```
    System.Object   ValueType
    DdeServer.RequestResult
```

### Thread Safety

Public static (`Shared` in Visual Basic) members of this type are safe for multithreaded operations. Instance members are not guaranteed to be thread-safe.

### Requirements

- **Namespace:** `NDde.Server`
- **Assembly:** NDde (in NDde.dll)

### See Also

- `DdeServer.RequestResult Members`
- `NDde.Server Namespace`

---

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeServer.RequestResult Members

**DdeServer.RequestResult overview**

### Public Static Fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>$ NotProcessed</code></td>
<td>Return this value if the request was not successful.</td>
</tr>
<tr>
<td><code>$ PauseConversation</code></td>
<td>Return this value to pause the conversation and execute the request asynchronously. After the conversation has been resumed the <code>OnRequest</code> method will run again.</td>
</tr>
</tbody>
</table>

### Public Static Operators

<table>
<thead>
<tr>
<th>Operator</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>$ Equality Operator</code></td>
<td>This determines whether two <code>RequestResult</code> objects are equal.</td>
</tr>
<tr>
<td><code>$ Inequality Operator</code></td>
<td>This determines whether two <code>ExecuteResult</code> objects are not equal.</td>
</tr>
</tbody>
</table>

### Public Instance Constructors

<table>
<thead>
<tr>
<th>Constructor</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>$ DdeServer.RequestResult Constructor</code></td>
<td>This initializes the <code>RequestResult</code> struct with the data to return to the client.</td>
</tr>
</tbody>
</table>

### Public Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>$ Data</code></td>
<td>The data to send to the client application.</td>
</tr>
</tbody>
</table>

### Public Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>$ Equals</code></td>
<td>This determines whether two object instances are equal.</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>GetHashCode</td>
<td>This returns a hash code for the object.</td>
</tr>
<tr>
<td>GetType (inherited from Object)</td>
<td>Gets the Type of the current instance.</td>
</tr>
<tr>
<td>ToString (inherited from ValueType)</td>
<td>Returns the fully qualified type name of this instance.</td>
</tr>
</tbody>
</table>

See Also

DdeServer.RequestResult Structure | NDde.Server Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
This initializes the `RequestResult` struct with the data to return to the client.

```csharp
public DdeServer.RequestResult(
    Byte[] data
);
```

**Parameters**

*data*

The data to return to the client.

**See Also**

[DdeServer.RequestResult Structure] | [NDde.Server Namespace]

[Send comments on this topic.]

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
The fields of the `DdeServer.RequestResult` structure are listed below. For a complete list of `DdeServer.RequestResult` structure members, see the [DdeServer.RequestResult Members](#) topic.

### Public Static Fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>NotProcessed</code></td>
<td>Return this value if the request was not successful.</td>
</tr>
<tr>
<td><code>PauseConversation</code></td>
<td>Return this value to pause the conversation and execute the request asynchronously. After the conversation has been resumed the <code>OnRequest</code> method will run again.</td>
</tr>
</tbody>
</table>

**See Also**

[DdeServer.RequestResult Structure](#) | [NDde.Server Namespace](#)

---

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeServer.RequestResult.NotProcessed Field

Return this value if the request was not successful.

```csharp
public static readonly NotProcessed NotProcessed;
```

See Also

DdeServer.RequestResult Class | NDde.Server Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeServer.RequestResult.PauseConversation Field

Return this value to pause the conversation and execute the request asynchronously. After the conversation has been resumed the `OnRequest` method will run again.

```csharp
public static readonly PauseConversation PauseConversation;
```

See Also

- [DdeServer.RequestResult Class](#) | [NDde.Server Namespace](#)

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
The properties of the `DdeServer.RequestResult` structure are listed below. For a complete list of `DdeServer.RequestResult` structure members, see the `DdeServer.RequestResult Members` topic.

### Public Instance Properties

| **Data** | The data to send to the client application. |

**See Also**

[DdeServer.RequestResult Structure](#) | [NDde.Server Namespace](#)

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeServer.RequestResult.Data Property

The data to send to the client application.

```csharp
public Byte[] Data {get; set;}
```

See Also

- DdeServer.RequestResult Structure | NDde.Server Namespace
- Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeServer.RequestResult Methods

The methods of the `DdeServer.RequestResult` structure are listed below. For a complete list of `DdeServer.RequestResult` structure members, see the `DdeServer.RequestResult Members` topic.

Public Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>Equals</code></td>
<td>This determines whether two object instances are equal.</td>
</tr>
<tr>
<td><code>GetHashCode</code></td>
<td>This returns a hash code for the object.</td>
</tr>
<tr>
<td><code>GetType</code> (inherited from <code>Object</code>)</td>
<td>Gets the <code>Type</code> of the current instance.</td>
</tr>
<tr>
<td><code>ToString</code> (inherited from <code>ValueType</code>)</td>
<td>Returns the fully qualified type name of this instance.</td>
</tr>
</tbody>
</table>

See Also

`DdeServer.RequestResult Structure` | `NDde.Server Namespace`

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeServer.RequestResult.Equals Method

This determines whether two object instances are equal.

```csharp
public override bool Equals(
    Object o
);
```

Parameters

- `o`  
  The object to compare with the current object.

Return Value

True if the specified object is equal to the current object, false otherwise.

See Also

DdeServer.RequestResult Structure | NDde.Server Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
This returns a hash code for the object.

```csharp
public override int GetHashCode();
```

Return Value

A hash code for the object.

See Also

[DdeServer.RequestResult Structure] | [NDde.Server Namespace]

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
The operators of the **DdeServer.RequestResult** structure are listed below. For a complete list of **DdeServer.RequestResult** structure members, see the [DdeServer.RequestResult Members](#) topic.

### Public Static Operators

<table>
<thead>
<tr>
<th>Operator</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>==</code></td>
<td><strong>Equality Operator</strong></td>
<td>This determines whether two <strong>RequestResult</strong> objects are equal.</td>
</tr>
<tr>
<td><code>!=</code></td>
<td><strong>Inequality Operator</strong></td>
<td>This determines whether two <strong>ExecuteResult</strong> objects are not equal.</td>
</tr>
</tbody>
</table>

### See Also

[DdeServer.RequestResult Structure](#) | [DdeServer.RequestResult Members](#) | [NDde.Server Namespace](#)

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeServer.RequestResult Equality Operator

This determines whether two RequestResult objects are equal.

```csharp
public static bool operator ==(DdeServer.RequestResult lhs, DdeServer.RequestResult rhs);
```

Parameters

- `lhs`  
  The left hand side object.

- `rhs`  

Return Value

True if the two objects are equal, false otherwise.

See Also

- DdeServer.RequestResult Structure | NDde.Server Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)

Generated from assembly NDde [2.1.563.0]
Dynamic Data Exchange Library for .NET
DdeServer.RequestResult Inequality Operator

This determines whether two `ExecuteResult` objects are not equal.

```csharp
public static bool operator !=(
    DdeServer.RequestResult lhs,
    DdeServer.RequestResult rhs
);
```

Parameters

- `lhs`
  - The left hand side object.

- `rhs`

Return Value

True if the two objects are not equal, false otherwise.

See Also

[DdeServer.RequestResult Structure] | [NDde.Server Namespace]

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Generated from assembly NDde [2.1.563.0]
Hierarchy

System.Object
  System.EventArgs
    NDde.DdeEventArgs
  System.Exception ---- System.Runtime.Serialization.ISerializable
    NDde.DdeException

See Also

NDde Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Dynamic Data Exchange Library for .NET
NDde.Advanced.DdeContext ----
System.ComponentModel.ISynchronizeInvoke,
System.IDisposable

NDde.Advanced.DdeMessageLoop ----
System.ComponentModel.ISynchronizeInvoke,
System.IDisposable

NDde.Advanced.DdeTransaction
NDde.Advanced.IDdeTransactionFilter

System.EventArgs

NDde.DdeEventArgs

NDde.Advanced.DdeRegistrationEventArgs

See Also

NDde.Advanced Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Dynamic Data Exchange Library for .NET
Hierarchy

System.Object

NDde.Advanced.Monitor.DdeMonitor ---- System.IDisposable

System.EventArgs

NDde.DdeEventArgs

NDde.Advanced.Monitor.DdeActivityEventArgs


NDde.Advanced.Monitor.DdeLinkActivityEventArgs


System.ValueType

System.Enum ---- System.IComparable, System.IConvertible, System.IFormattable

NDde.Advanced.Monitor.DdeMessageActivityKind

NDde.Advanced.Monitor.DdeMonitorFlags

See Also

NDde.Advanced.Monitor Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Dynamic Data Exchange Library for .NET
See Also

NDde.Client Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)
Dynamic Data Exchange Library for .NET
Hierarchy

System.Object

NDde.Server.DdeConversation

NDde.Server.DdeServer ---- System.IDisposable

System.ValueType

NDde.Server.DdeServer.ExecuteResult

NDde.Server.DdeServer.PokeResult

NDde.Server.DdeServer.RequestResult

See Also

NDde.Server Namespace

Send comments on this topic.

Copyright (c) 2005-2006 by Brian Gideon (briangideon@yahoo.com)