DotZLib .Net wrapper for ZLib1.dll
## DotZLib Namespace

### Classes

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AdlerChecksum</td>
<td>Implements a checksum generator that computes the Adler checksum on data</td>
</tr>
<tr>
<td>ChecksumGeneratorBase</td>
<td>Implements the common functionality needed for all ChecksumGenerators</td>
</tr>
<tr>
<td>CodecBase</td>
<td>Implements the common functionality needed for all Codecs</td>
</tr>
<tr>
<td>CRC32Checksum</td>
<td>Implements a CRC32 checksum generator</td>
</tr>
<tr>
<td>Deflater</td>
<td>Implements a data compressor, using the deflate algorithm in the ZLib dll</td>
</tr>
<tr>
<td>GZipStream</td>
<td>Implements a compressed Stream, in GZip (.gz) format.</td>
</tr>
<tr>
<td>Inflater</td>
<td>Implements a data decompressor, using the inflate algorithm in the ZLib dll</td>
</tr>
<tr>
<td>Info</td>
<td>Encapsulates general information about the ZLib library</td>
</tr>
<tr>
<td>ZLibException</td>
<td>The exception that is thrown when an error occurs on the zlib dll</td>
</tr>
</tbody>
</table>

### Interfaces

<table>
<thead>
<tr>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
</table>
**ChecksumGenerator**
Declares methods and properties that enable a running checksum to be calculated.

**Codec**
Declares methods and events for implementing compressors/decompressors.

---

### Delegates

<table>
<thead>
<tr>
<th>Delegate</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>DataAvailableHandler</strong></td>
<td>Represents the method that will be called from a codec when new data are available.</td>
</tr>
</tbody>
</table>

### Enumerations

<table>
<thead>
<tr>
<th>Enumeration</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CompressLevel</strong></td>
<td>Defines constants for the available compression levels in zlib</td>
</tr>
</tbody>
</table>
DotZLib .Net wrapper for ZLib1.dll
AdlerChecksum Class

Implements a checksum generator that computes the Adler checksum on data

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

System.Object  ChecksumGeneratorBase
  AdlerChecksum

public sealed class AdlerChecksum : ChecksumGeneratorBase

Requirements

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

See Also

AdlerChecksum Members | DotZLib Namespace

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
## AdlerChecksum Members

### AdlerChecksum overview

#### Public Instance Constructors

<table>
<thead>
<tr>
<th>Constructor</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AdlerChecksum</td>
<td>Overloaded. Initializes a new instance of the AdlerChecksum class.</td>
</tr>
</tbody>
</table>

#### Public Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value (inherited from ChecksumGeneratorBase)</td>
<td>Gets the current checksum value</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, 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>Reset (inherited from ChecksumGeneratorBase)</td>
<td>Resets the current checksum to zero</td>
</tr>
<tr>
<td>ToString (inherited from Object)</td>
<td>Returns a String that represents the current Object.</td>
</tr>
<tr>
<td>Update (inherited from ChecksumGeneratorBase)</td>
<td>Overloaded. Updates the current checksum with the data from a string, using a specific encoding</td>
</tr>
<tr>
<td>Update</td>
<td>Overloaded. Updates the current checksum with part of an array of bytes</td>
</tr>
</tbody>
</table>
Protected Instance Fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>_current</td>
<td>(inherited from ChecksumGeneratorBase) The value of the current checksum</td>
</tr>
</tbody>
</table>

Protected Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Finalize</td>
<td>(inherited from Object) 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</td>
<td>(inherited from Object) Creates a shallow copy of the current Object.</td>
</tr>
</tbody>
</table>

See Also

AdlerChecksum Class | DotZLib Namespace

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
AdlerChecksum Constructor

Initializes a new instance of the Adler checksum generator

Overload List

Initializes a new instance of the Adler checksum generator

public AdlerChecksum();

Initializes a new instance of the Adler checksum generator with a specified value

public AdlerChecksum(uint);

See Also

AdlerChecksum Class | DotZLib Namespace

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
AdlerChecksum Constructor ()

Initializes a new instance of the Adler checksum generator

```java
public AdlerChecksum();
```

See Also

AdlerChecksum Class | DotZLib Namespace | AdlerChecksum Constructor Overload List

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
AdlerChecksum Constructor (UInt32)

Initializes a new instance of the Adler checksum generator with a specified value

```csharp
public AdlerChecksum(
    uint initialValue
);
```

Parameters

initialValue
The value to set the current checksum to

See Also

AdlerChecksum Class | DotZLib Namespace | AdlerChecksum Constructor Overload List

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
AdlerChecksum Methods

The methods of the AdlerChecksum class are listed below. For a complete list of AdlerChecksum class members, see the AdlerChecksum 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 Object is equal to the current Object.</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table.</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the Type of the current instance.</td>
</tr>
<tr>
<td><strong>Reset</strong></td>
<td>Resets the current checksum to zero.</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a String that represents the current Object.</td>
</tr>
<tr>
<td><strong>Update</strong></td>
<td>Overloaded. Updates the current checksum with the data from a string, using a specific encoding.</td>
</tr>
<tr>
<td><strong>Update</strong></td>
<td>Overloaded. Updates the current checksum with part of an array of bytes.</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 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 <strong>Object</strong>)</td>
<td>Creates a shallow copy of the current <strong>Object</strong>.</td>
</tr>
</tbody>
</table>

**See Also**

- [AdlerChecksum Class](#) | [DotZLib Namespace](#)

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
AdlerChecksum.Update Method

Updates the current checksum with part of an array of bytes

Overload List

Inherited from ChecksumGeneratorBase.

public void Update(byte[]);
Updates the current checksum with part of an array of bytes

public override void Update(byte[], int, int);

Inherited from ChecksumGeneratorBase.

public void Update(string);

Inherited from ChecksumGeneratorBase.

public void Update(string, Encoding);

See Also

AdlerChecksum Class | DotZLib Namespace

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
AdlerChecksum.Update Method (Byte[], Int32, Int32)

Updates the current checksum with part of an array of bytes

```csharp
public override void Update(
    byte[] data,
    int offset,
    int count
);
```

Parameters

- `data`
  The data to update the checksum with

- `offset`
  Where in `data` to start updating

- `count`
  The number of bytes from `data` to use

Implements

- ChecksumGenerator.Update

Exceptions

<table>
<thead>
<tr>
<th>Exception Type</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ArgumentException</td>
<td>The sum of offset and count is larger than the length of <code>data</code></td>
</tr>
<tr>
<td>NullReferenceException</td>
<td><code>data</code> is a null reference</td>
</tr>
<tr>
<td>ArgumentOutOfRangeException</td>
<td>Offset or count is negative.</td>
</tr>
</tbody>
</table>

See Also

- AdlerChecksum Class
- DotZLib Namespace
- AdlerChecksum.Update Overload List

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
**ChecksumGenerator Interface**

Declares methods and properties that enables a running checksum to be calculated.

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

```csharp
public interface ChecksumGenerator
```

**Requirements**

**Namespace:** [DotZLib](#)

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

**See Also**

[ChecksumGenerator Members](#) | [DotZLib Namespace](#)

---

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
**ChecksumGenerator Members**

**ChecksumGenerator overview**

Public Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value</td>
<td>Gets the current value of the checksum</td>
</tr>
</tbody>
</table>

Public Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reset</td>
<td>Clears the current checksum to 0</td>
</tr>
<tr>
<td>Update</td>
<td>Overloaded. Updates the current checksum with the data from a string, using a specific encoding</td>
</tr>
</tbody>
</table>

See Also

- [ChecksumGenerator Interface](#) | [DotZLib Namespace](#)

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
ChecksumGenerator Properties

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

Public Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Value</strong></td>
<td>Gets the current value of the checksum</td>
</tr>
</tbody>
</table>

See Also

ChecksumGenerator Interface | DotZLib Namespace

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
ChecksumGenerator/value Property

Gets the current value of the checksum

```csharp
uint Value {get;}
```

See Also

[ChecksumGenerator Interface] | [DotZLib Namespace]

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
The methods of the ChecksumGenerator interface are listed below. For a complete list of ChecksumGenerator interface members, see the ChecksumGenerator Members topic.

## Public Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Reset</strong></td>
<td>Clears the current checksum to 0</td>
</tr>
<tr>
<td><strong>Update</strong></td>
<td>Overloaded. Updates the current checksum with the data from a string, using a specific encoding</td>
</tr>
</tbody>
</table>

See Also

[ChecksumGenerator Interface] | [DotZLib Namespace]

---

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
**ChecksumGenerator.Reset Method**

Clears the current checksum to 0

```csharp
void Reset();
```

**See Also**

[ChecksumGenerator Interface] | [DotZLib Namespace]

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
ChecksumGenerator.Update Method

Updates the current checksum with an array of bytes

Overload List

Updates the current checksum with an array of bytes

    void Update(byte[]);

Updates the current checksum with part of an array of bytes

    void Update(byte[], int, int);

Updates the current checksum with the data from a string

    void Update(string);

Updates the current checksum with the data from a string, using a specific encoding

    void Update(string, Encoding);

See Also

    ChecksumGenerator Interface | DotZLib Namespace

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
ChecksumGenerator.Update Method (Byte[])  

Updates the current checksum with an array of bytes

```csharp
void Update(byte[] data);
```

Parameters

`data`  
The data to update the checksum with

See Also

ChecksumGenerator Interface  |  DotZLib Namespace  |  ChecksumGenerator.Update Overload List

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
Updates the current checksum with part of an array of bytes

```csharp
void Update(
    byte[] data,
    int offset,
    int count
);
```

**Parameters**

- `data`  
The data to update the checksum with
- `offset`  
  Where in `data` to start updating
- `count`  
The number of bytes from `data` to use

**Exceptions**

<table>
<thead>
<tr>
<th>Exception Type</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ArgumentException</code></td>
<td>The sum of offset and count is larger than the length of <code>data</code></td>
</tr>
<tr>
<td><code>ArgumentNullException</code></td>
<td><code>data</code> is a null reference</td>
</tr>
<tr>
<td><code>ArgumentOutOfRangeException</code></td>
<td>Offset or count is negative.</td>
</tr>
</tbody>
</table>

**See Also**

[ChecksumGenerator Interface](#) | [DotZLib Namespace](#) | [ChecksumGenerator.Update Overload List](#)
DotZLib .Net wrapper for ZLib1.dll
ChecksumGenerator.Update Method (String)

Updates the current checksum with the data from a string

```csharp
void Update(
    string data
);
```

Parameters

`data`

The string to update the checksum with

Remarks

The characters in the string are converted by the UTF-8 encoding

See Also

ChecksumGenerator Interface | DotZLib Namespace | ChecksumGenerator.Update Overload List

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
ChecksumGenerator.Update Method (String, Encoding)

Updates the current checksum with the data from a string, using a specific encoding

```csharp
void Update(
    string data,
    Encoding encoding
);
```

Parameters

- `data`:
  The string to update the checksum with

- `encoding`:
  The encoding to use

See Also

- ChecksumGenerator Interface
- DotZLib Namespace
- ChecksumGenerator.Update Overload List

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
**ChecksumGeneratorBase Class**

Implements the common functionality needed for all ChecksumGenerators.

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

**System.Object** ChecksumGeneratorBase

```csharp
public abstract class ChecksumGeneratorBase : ChecksumGenerator
```

**Example**

**Requirements**

- **Namespace:** [DotLib](#)
- **Assembly:** DotLib (in DotLib.dll)

**See Also**

- [ChecksumGeneratorBase Members](#) | [DotLib Namespace](#)

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
## ChecksumGeneratorBase Members

### ChecksumGeneratorBase overview

### Public Instance Constructors

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ChecksumGeneratorBase</code></td>
<td>Overloaded. Initializes a new instance of the ChecksumGeneratorBase class.</td>
</tr>
</tbody>
</table>

### Public Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>Value</code></td>
<td>Gets the current checksum value</td>
</tr>
</tbody>
</table>

### Public Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>Equals</code> (inherited from <code>Object</code>)</td>
<td>Determines whether the specified <code>Object</code> is equal to the current <code>Object</code>.</td>
</tr>
<tr>
<td><code>GetHashCode</code> (inherited from <code>Object</code>)</td>
<td>Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table.</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>Reset</code></td>
<td>Resets the current checksum to zero</td>
</tr>
<tr>
<td><code>ToString</code> (inherited from <code>Object</code>)</td>
<td>Returns a <code>String</code> that represents the current <code>Object</code>.</td>
</tr>
<tr>
<td><code>Update</code></td>
<td>Overloaded. Updates the current checksum with the data from a string, using a specific encoding</td>
</tr>
</tbody>
</table>

### Protected Instance Fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>_current</code></td>
<td>The value of the current</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 <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>
</tbody>
</table>

See Also

[ChecksumGeneratorBase Class](#) | [DotZLib Namespace](#)

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
**ChecksumGeneratorBase Constructor**

Initializes a new instance of the checksum generator base - the current checksum is set to zero

**Overload List**

Initializes a new instance of the checksum generator base - the current checksum is set to zero

```csharp
public ChecksumGeneratorBase();
```

Initializes a new instance of the checksum generator base with a specified value

```csharp
public ChecksumGeneratorBase(uint);
```

**See Also**

[ChecksumGeneratorBase Class](#) | [DotZLib Namespace](#)

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
ChecksumGeneratorBase Constructor ()

Initializes a new instance of the checksum generator base - the current checksum is set to zero

public ChecksumGeneratorBase();

See Also
ChecksumGeneratorBase Class | DotZLib Namespace | ChecksumGeneratorBase Constructor Overload List

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
CheckSUMGeneratorBase Constructor (UInt32)

Initializes a new instance of the checksum generator basewith a specified value

```csharp
public ChecksumGeneratorBase(
    uint initialValue);
```

**Parameters**

*initialValue*

The value to set the current checksum to

**See Also**

CheckSUMGeneratorBase Class | DotZLib Namespace | CheckSUMGeneratorBase Constructor Overload List

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
The fields of the **ChecksumGeneratorBase** class are listed below. For a complete list of **ChecksumGeneratorBase** class members, see the **ChecksumGeneratorBase Members** topic.

### Protected Instance Fields

| _current | The value of the current checksum |

---

See Also

[ChecksumGeneratorBase Class](#) | [DotZLib Namespace](#)

---

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
ChecksumGeneratorBase._current Field

The value of the current checksum

```
protected uint _current;
```

See Also

[ChecksumGeneratorBase Class] | [DotZLib Namespace]

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
ChecksumGeneratorBase Properties

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

Public Instance Properties

| Value | Gets the current checksum value |

See Also

- `ChecksumGeneratorBase Class` | `DotZLib Namespace`

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
ChecksumGeneratorBase.Value Property

Gets the current checksum value

```csharp
public uint Value {get;}
```

Implements

- [ChecksumGenerator.Value](#)

See Also

- [ChecksumGeneratorBase Class](#) | [DotZLib Namespace](#)

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
The methods of the **ChecksumGeneratorBase** class are listed below. For a complete list of **ChecksumGeneratorBase** class members, see the [ChecksumGeneratorBase 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, 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>Reset</strong></td>
<td>Resets the current checksum to zero</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>Update</strong></td>
<td>Overloaded. Updates the current checksum with the data from a string, using a specific encoding</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 <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>
</tbody>
</table>

See Also
ChecksumGeneratorBase Class | DotZLib Namespace

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
ChecksumGeneratorBase.Reset Method

Resets the current checksum to zero

```java
public void Reset();
```

Implements

ChecksumGenerator.Reset

See Also

ChecksumGeneratorBase Class | DotZLib Namespace

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
ChecksumGeneratorBase.Update Method

Updates the current checksum with an array of bytes.

Overload List

Updates the current checksum with an array of bytes.

public void Update(byte[]);

Updates the current checksum with part of an array of bytes

public abstract void Update(byte[], int, int);

Updates the current checksum with the data from a string

public void Update(string);

Updates the current checksum with the data from a string, using a specific encoding

public void Update(string, Encoding);

See Also

ChecksumGeneratorBase Class | DotZLib Namespace

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
ChecksumGeneratorBase.Update Method (Byte[])

Updates the current checksum with an array of bytes.

```csharp
public void Update(byte[] data);
```

Parameters

`data`

The data to update the checksum with

Implements

ChecksumGenerator.Update

See Also

ChecksumGeneratorBase Class | DotZLib Namespace | ChecksumGeneratorBase.Update Overload List

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
Updates the current checksum with part of an array of bytes

```csharp
public abstract void Update(
    byte[] data,
    int offset,
    int count
);
```

**Parameters**

*data*
- The data to update the checksum with

*offset*
- Where in `data` to start updating

*count*
- The number of bytes from `data` to use

**Implements**

ChecksumGenerator.Update

**Remarks**

All the other `Update` methods are implemented in terms of this one. This is therefore the only method a derived class has to implement

**Exceptions**

<table>
<thead>
<tr>
<th>Exception Type</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ArgumentException</td>
<td>The sum of offset and count is larger than the length of <code>data</code></td>
</tr>
<tr>
<td>NullReferenceException</td>
<td><code>data</code> is a null reference</td>
</tr>
<tr>
<td>ArgumentOutOfRangeException</td>
<td>Offset or count is negative.</td>
</tr>
</tbody>
</table>

**See Also**

ChecksumGeneratorBase.Update Method (Byte[], Int32, Int32)
ChecksumGeneratorBase Class | DotZLib Namespace | ChecksumGeneratorBase.Update Overload List

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
Update the current checksum with the data from a string

```csharp
public void Update(string data);
```

Parameters

- `data` - The string to update the checksum with

Implements

- `ChecksumGenerator.Update`

Remarks

- The characters in the string are converted by the UTF-8 encoding

See Also

- `ChecksumGeneratorBase Class` | `DotZLib Namespace` | `ChecksumGeneratorBase.Update Overload List`

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
ChecksumGeneratorBase.Update Method (String, Encoding)

Updates the current checksum with the data from a string, using a specific encoding

```csharp
public void Update(
    string data,
    Encoding encoding
);
```

**Parameters**

- `data`
  - The string to update the checksum with

- `encoding`
  - The encoding to use

**Implements**

ChecksumGenerator.Update

**See Also**

ChecksumGeneratorBase Class | DotZLib Namespace | ChecksumGeneratorBase.Update Overload List

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
Codec Interface

Declares methods and events for implementing compressors/decompressors

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

```java
public interface Codec
```

Requirements

Namespace: DotZLib

Assembly: DotZLib (in DotZLib.dll)

See Also

Codec Members | DotZLib Namespace

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
Codec Members

Codec overview

Public Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Checksum</strong></td>
<td>Gets the checksum of the data that has been added so far</td>
</tr>
</tbody>
</table>

Public Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Add</strong></td>
<td>Overloaded. Adds more data to the codec to be processed.</td>
</tr>
<tr>
<td><strong>Finish</strong></td>
<td>Finishes up any pending data that needs to be processed and handled.</td>
</tr>
</tbody>
</table>

Public Instance Events

<table>
<thead>
<tr>
<th>Event</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>DataAvailable</strong></td>
<td>Occurs when more processed data are available.</td>
</tr>
</tbody>
</table>

See Also

- Codec Interface
- DotZLib Namespace

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
Codec Properties

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

Public Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Checksum</td>
<td>Gets the checksum of the data that has been added so far</td>
</tr>
</tbody>
</table>

See Also

Codec Interface | DotZLib Namespace

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
Codec.Checksum Property

Gets the checksum of the data that has been added so far

```csharp
uint Checksum {get;}
```

See Also

- [Codec Interface](#)
- [DotZLib Namespace](#)

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
The methods of the **Codec** interface are listed below. For a complete list of **Codec** interface members, see the **Codec Members** topic.

**Public Instance Methods**

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Add</strong></td>
<td>Overloaded. Adds more data to the codec to be processed.</td>
</tr>
<tr>
<td><strong>Finish</strong></td>
<td>Finishes up any pending data that needs to be processed and handled.</td>
</tr>
</tbody>
</table>

**See Also**

[Codec Interface](#) | [DotZLib Namespace](#)

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
Codec.Add Method

Adds more data to the codec to be processed.

Overload List

Adds more data to the codec to be processed.

void Add(byte[]);

Adds more data to the codec to be processed.

void Add(byte[], int, int);

See Also

Codec Interface | DotZLib Namespace

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
Codec.Add Method (Byte[])

Adds more data to the codec to be processed.

```csharp
void Add(byte[] data);
```

**Parameters**

*data*  
Byte array containing the data to be added to the codec

**Remarks**

Adding data may, or may not, raise the **DataAvailable** event

**See Also**

Codec Interface | DotZLib Namespace | Codec.Add Overload List

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
Codec.Add Method (Byte[], Int32, Int32)

Adds more data to the codec to be processed.

```csharp
void Add(
    byte[] data,
    int offset,
    int count
);
```

Parameters

- **data**
  - Byte array containing the data to be added to the codec

- **offset**
  - The index of the first byte to add from `data`

- **count**
  - The number of bytes to add

Remarks

Adding data may, or may not, raise the `DataAvailable` event

See Also

- Codec Interface | DotZLib Namespace | Codec.Add Overload List

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
Codec_FINISH Method

Finishes up any pending data that needs to be processed and handled.

```void Finish();```

See Also

- Codec Interface | DotZLib Namespace

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
Codec Events

The events of the **Codec** interface are listed below. For a complete list of **Codec** interface members, see the [Codec Members](#) topic.

**Public Instance Events**

<table>
<thead>
<tr>
<th>Event</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DataAvailable</td>
<td>Occurs when more processed data are available.</td>
</tr>
</tbody>
</table>

**See Also**

[Codec Interface](#) | [DotZLib Namespace](#)

---

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
Codec.DataAvailable Event

Occurs when more processed data are available.

```csharp
    event DataAvailableHandler DataAvailable;
```

See Also

- Codec Interface
- DotZLib Namespace

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
CodecBase Class

Implements the common functionality needed for all Codec
For a list of all members of this type, see CodecBase Members.
System.Object CodecBase

```csharp
public abstract class CodecBase : Codec, IDisposable
```

Requirements

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

See Also

CodecBase Members | DotZLib Namespace

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
# CodecBase Members

## CodecBase overview

### Protected Static Fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>kBufferSize</td>
<td>The size of the internal buffers</td>
</tr>
</tbody>
</table>

### Public Instance Constructors

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

### Public Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Checksum</td>
<td>Gets the checksum of the data that has been added so far</td>
</tr>
</tbody>
</table>

### Public Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add</td>
<td>Overloaded. Adds more data to the codec to be processed.</td>
</tr>
<tr>
<td>Dispose</td>
<td>Releases any unmanaged resources and calls the CleanUp method of the derived class</td>
</tr>
<tr>
<td>Equals</td>
<td>Determines whether the specified Object is equal to the current Object.</td>
</tr>
<tr>
<td>Finish</td>
<td>Finishes up any pending data that needs to be processed and handled.</td>
</tr>
<tr>
<td>GetHashCode</td>
<td>Serves as a hash function for a particular type, 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>
- **ToString** (inherited from Object): Returns a String that represents the current Object.

**Public Instance Events**

- **DataAvailable**: Occurs when more processed data are available.

**Protected Instance Fields**

- **_isDisposed**: True if the object instance has been disposed, false otherwise.

**Protected Instance Methods**

- **CleanUp**: Performs any codec specific cleanup.
- **copyInput**: Copies a number of bytes to the internal codec buffer - ready for processing.
- **Finalize**: Destroys this instance.
- **MemberwiseClone** (inherited from Object): Creates a shallow copy of the current Object.
- **OnDataAvailable**: Fires the DataAvailable event.
- **resetOutput**: Resets the internal output buffers to a known state - ready for processing.
- **setChecksum**: Updates the running checksum property.

**See Also**

CodecBase Class | DotZLib Namespace

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
**CodecBase Constructor**

Initializes a new instance of the `CodecBase` class.

```java
public CodecBase();
```

**See Also**

[CodecBase Class](#) | [DotZLib Namespace](#)

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
CodecBase Fields

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

### Protected Static Fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>kBufferSize</td>
<td>The size of the internal buffers</td>
</tr>
</tbody>
</table>

### Protected Instance Fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>_isDisposed</td>
<td>True if the object instance has been disposed, false otherwise</td>
</tr>
</tbody>
</table>

**See Also**

- CodecBase Class | DotZLib Namespace

---

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
**CodecBase._isDisposed Field**

True if the object instance has been disposed, false otherwise

```cpp
protected bool _isDisposed;
```

**See Also**

[CodecBase Class](#) | [DotZLib Namespace](#)

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
CodecBase.kBufferSize Field

The size of the internal buffers

```cpp
protected const int kBufferSize;
```

See Also

CodecBase Class | DotZLib Namespace

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
The properties of the **CodecBase** class are listed below. For a complete list of **CodecBase** class members, see the **CodecBase Members** topic.

### Public Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Checksum</strong></td>
<td>Gets the checksum of the data that has been added so far</td>
</tr>
</tbody>
</table>

**See Also**

[CodecBase Class](#) | [DotZLib Namespace](#)
DotZLib .Net wrapper for ZLib1.dll
**CodecBase.Checksum Property**

Gets the checksum of the data that has been added so far

```csharp
public uint Checksum {get;}
```

**Implements**

[Codec.Checksum](#)

**See Also**

[CodecBase Class](#) | [DotZLib Namespace](#)

---

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
CodecBase Methods

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

**Public Instance Methods**

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Add</strong></td>
<td>Overloaded. Adds more data to the codec to be processed.</td>
</tr>
<tr>
<td><strong>Dispose</strong></td>
<td>Releases any unmanaged resources and calls the <strong>CleanUp</strong> method of the derived class</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>Finish</strong></td>
<td>Finishes up any pending data that needs to be processed and handled.</td>
</tr>
<tr>
<td><strong>GetHashCode</strong> (inherited from <strong>Object</strong>)</td>
<td>Serves as a hash function for a particular type, 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>Object</strong>)</td>
<td>Returns a <strong>String</strong> that represents the current <strong>Object</strong>.</td>
</tr>
</tbody>
</table>

**Protected Instance Methods**

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CleanUp</strong></td>
<td>Performs any codec specific cleanup</td>
</tr>
<tr>
<td><strong>copyInput</strong></td>
<td>Copies a number of bytes to the internal codec buffer - ready for processing</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Destroys this instance</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>(inherited from Object) Creates a shallow copy of the current Object.</td>
</tr>
<tr>
<td><strong>OnDataAvailable</strong></td>
<td>Fires the DataAvailable event</td>
</tr>
<tr>
<td><strong>resetOutput</strong></td>
<td>Resets the internal output buffers to a known state - ready for processing</td>
</tr>
<tr>
<td><strong>setChecksum</strong></td>
<td>Updates the running checksum property</td>
</tr>
</tbody>
</table>

**See Also**

[CodecBase Class] | [DotZLib Namespace]

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
CodecBase.Add Method

Adds more data to the codec to be processed.

Overload List

Adds more data to the codec to be processed.

public void Add(byte[]);

Adds more data to the codec to be processed.

public abstract void Add(byte[], int, int);

See Also

CodecBase Class | DotZLib Namespace

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
CodecBase.Add Method (Byte[])  

Adds more data to the codec to be processed.

```csharp
public void Add(
    byte[] data
);
```

Parameters

- `data`  
  Byte array containing the data to be added to the codec

Implements

- Codec.Add

Remarks

Adding data may, or may not, raise the DataAvailable event

See Also

- CodecBase Class  |  DotZLib Namespace  |  CodecBase.Add Overload List

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
CodecBase.Add Method (Byte[], Int32, Int32)

Adds more data to the codec to be processed.

```csharp
public abstract void Add(
    byte[] data,
    int offset,
    int count
);
```

Parameters

- **data**
  Byte array containing the data to be added to the codec

- **offset**
  The index of the first byte to add from `data`

- **count**
  The number of bytes to add

Implements

- Codec.Add

Remarks

Adding data may, or may not, raise the `DataAvailable` event. This must be implemented by a derived class

See Also

- CodecBase Class
- DotZLib Namespace
- CodecBase.Add Overload List

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
CodecBase.CleanUp Method

Performs any codec specific cleanup

```csharp
protected abstract void CleanUp();
```

Remarks

This must be implemented by a derived class

See Also

[CodecBase Class](#) | [DotZLib Namespace](#)

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
Copies a number of bytes to the internal codec buffer - ready for processing

```java
protected void copyInput(
    byte[] data,
    int startIndex,
    int count
);
```

**Parameters**

- `data`  
  The byte array that contains the data to copy

- `startIndex`  
  The index of the first byte to copy

- `count`  
  The number of bytes to copy from `data`

**See Also**

- [CodecBase Class](#) | [DotZLib Namespace](#)
CodecBase.Dispose Method

Releases any unmanaged resources and calls the CleanUp method of the derived class

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

Implements

IDisposable.Dispose

See Also

CodecBase Class | DotZLib Namespace

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
CodecBase.Finalize Method

Destroys this instance

```csharp
protected override void Finalize();
```

See Also

CodecBase Class | DotZLib Namespace

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
CodecBase.Finish Method

Finishes up any pending data that needs to be processed and handled.

```java
public abstract void Finish();
```

Implements

Codec.Finish

Remarks

This must be implemented by a derived class

See Also

CodecBase Class | DotZLib Namespace

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
CodecBase.OnDataAvailable Method

Fires the DataAvailable event

protected void OnDataAvailable();

See Also

CodecBase Class | DotZLib Namespace

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
Resets the internal output buffers to a known state - ready for processing

```java
protected void resetOutput();
```

See Also

- [CodecBase Class](#)
- [DotZLib Namespace](#)
DotZLib .Net wrapper for ZLib1.dll
**CodecBase.setChecksum Method**

Updates the running checksum property

```
protected void setChecksum(
    uint newSum
);
```

**Parameters**

`newSum`

The new checksum value

**See Also**

[CodecBase Class](#) | [DotZLib Namespace](#)

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
CodecBase Events

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

### Public Instance Events

<table>
<thead>
<tr>
<th>Event</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DataAvailable</td>
<td>Occurs when more processed data are available.</td>
</tr>
</tbody>
</table>

**See Also**

[CodecBase Class](#) | [DotZLib Namespace](#)

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
CodecBase>DataAvailable Event

Occurs when more processed data are available.

**public event** DataAvailableHandler DataAvailable;

**Implements**

Codec.DataAvailable

**See Also**

CodecBase Class | DotZLib Namespace

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
CompressLevel Enumeration

Defines constants for the available compression levels in zlib

```csharp
public enum CompressLevel
```

Members

<table>
<thead>
<tr>
<th>Member Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>The default compression level with a reasonable compromise between compression and speed.</td>
</tr>
<tr>
<td>None</td>
<td>No compression at all. The data are passed straight through.</td>
</tr>
<tr>
<td>Best</td>
<td>The maximum compression rate available.</td>
</tr>
<tr>
<td>Fastest</td>
<td>The fastest available compression level.</td>
</tr>
</tbody>
</table>

Requirements

Namespace: DotZLib

Assembly: DotZLib (in DotZLib.dll)

See Also

DotZLib Namespace

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
CRC32Checksum Class

Implements a CRC32 checksum generator

For a list of all members of this type, see CRC32Checksum Members.

System.Object  ChecksumGeneratorBase

CRC32Checksum

public sealed class CRC32Checksum : ChecksumGeneratorBase

Requirements

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

See Also

CRC32Checksum Members  |  DotZLib Namespace

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
**CRC32Checksum Members**

**CRC32Checksum overview**

**Public Instance Constructors**

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRC32Checksum</td>
<td>Overloaded. Initializes a new instance of the CRC32Checksum class.</td>
</tr>
</tbody>
</table>

**Public Instance Properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value</td>
<td>Gets the current checksum value</td>
</tr>
</tbody>
</table>

**Public Instance Methods**

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Equals</td>
<td>Determines whether the specified <a href="#">Object</a> is equal to the current <a href="#">Object</a>.</td>
</tr>
<tr>
<td>GetHashCode</td>
<td>Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table.</td>
</tr>
<tr>
<td>GetType</td>
<td>Gets the <a href="#">Type</a> of the current instance.</td>
</tr>
<tr>
<td>Reset</td>
<td>Resets the current checksum to zero</td>
</tr>
<tr>
<td>ToString</td>
<td>Returns a <a href="#">String</a> that represents the current <a href="#">Object</a>.</td>
</tr>
<tr>
<td>Update</td>
<td>Overloaded. Updates the current checksum with the data from a string, using a specific encoding</td>
</tr>
<tr>
<td>Update</td>
<td>Overloaded. Updates the current checksum with part of an array of bytes</td>
</tr>
</tbody>
</table>
Protected Instance Fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>_current</td>
<td>The value of the current checksum</td>
</tr>
</tbody>
</table>

Protected Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Finalize</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</td>
<td>Creates a shallow copy of the current Object.</td>
</tr>
</tbody>
</table>

See Also

CRC32Checksum Class | DotZLib Namespace

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
**CRC32Checksum Constructor**

Initializes a new instance of the CRC32 checksum generator

**Overload List**

Initializes a new instance of the CRC32 checksum generator

    public CRC32Checksum();

Initializes a new instance of the CRC32 checksum generator with a specified value

    public CRC32Checksum(uint);

**See Also**

CRC32Checksum Class | DotZLib Namespace

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
**CRC32Checksum Constructor ()**

Initializes a new instance of the CRC32 checksum generator

```java
public CRC32Checksum();
```

**See Also**

- [CRC32Checksum Class](#)
- [DotZLib Namespace](#)
- [CRC32Checksum Constructor Overload List](#)

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
CRC32Checksum Constructor (UInt32)

Initializes a new instance of the CRC32 checksum generator with a specified value

```csharp
public CRC32Checksum(
    uint initialValue
);
```

Parameters

`initialValue`

The value to set the current checksum to

See Also

[ CRC32Checksum Class ](#) | [ DotZLib Namespace ](#) | [ CRC32Checksum Constructor Overload List ](#)

---

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
The methods of the **CRC32Checksum** class are listed below. For a complete list of **CRC32Checksum** class members, see the **CRC32Checksum Members** topic.

### Public Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>Equals</code> (inherited from Object)</td>
<td>Determines whether the specified <strong>Object</strong> is equal to the current <strong>Object</strong>.</td>
</tr>
<tr>
<td><code>GetHashCode</code> (inherited from Object)</td>
<td>Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table.</td>
</tr>
<tr>
<td><code>GetType</code> (inherited from Object)</td>
<td>Gets the <strong>Type</strong> of the current instance.</td>
</tr>
<tr>
<td><code>Reset</code> (inherited from ChecksumGeneratorBase)</td>
<td>Resets the current checksum to zero</td>
</tr>
<tr>
<td><code>ToString</code> (inherited from Object)</td>
<td>Returns a <strong>String</strong> that represents the current <strong>Object</strong>.</td>
</tr>
<tr>
<td><code>Update</code> (inherited from ChecksumGeneratorBase)</td>
<td>Overloaded. Updates the current checksum with the data from a string, using a specific encoding</td>
</tr>
<tr>
<td><code>Update</code></td>
<td>Overloaded. Updates the current checksum with part of an array of bytes</td>
</tr>
</tbody>
</table>

### Protected Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>Finalize</code> (inherited from Object)</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>
</tbody>
</table>
**MemberwiseClone** (inherited from **Object**) | Creates a shallow copy of the current **Object**.

See Also

[CRC32Checksum Class] | [DotZLib Namespace]

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
**CRC32Checksum.Update Method**

Updates the current checksum with part of an array of bytes

**Overload List**

Inherited from **ChecksumGeneratorBase**.

`public void Update(byte[]);`

Updates the current checksum with part of an array of bytes

`public override void Update(byte[],int,int);`

Inherited from **ChecksumGeneratorBase**.

`public void Update(string);`

Inherited from **ChecksumGeneratorBase**.

`public void Update(string,Encoding);`

**See Also**

[CRC32Checksum Class] [DotZLib Namespace]

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
Updates the current checksum with part of an array of bytes

```csharp
public override void Update(
    byte[] data,
    int offset,
    int count
);
```

**Parameters**

- `data`
  - The data to update the checksum with

- `offset`
  - Where in `data` to start updating

- `count`
  - The number of bytes from `data` to use

**Implements**

- `ChecksumGenerator.Update`

**Exceptions**

<table>
<thead>
<tr>
<th>Exception Type</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ArgumentException</code></td>
<td>The sum of offset and count is larger than the length of <code>data</code></td>
</tr>
<tr>
<td><code>NullReferenceException</code></td>
<td><code>data</code> is a null reference</td>
</tr>
<tr>
<td><code>ArgumentOutOfRangeException</code></td>
<td>Offset or count is negative.</td>
</tr>
</tbody>
</table>

**See Also**

- [CRC32Checksum Class](#) | [DotZLib Namespace](#) |
- [CRC32Checksum.Update Overload List](#)

---

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
DataAvailableHandler Delegate

Represents the method that will be called from a codec when new data are available.

```csharp
public delegate void DataAvailableHandler(
    byte[] data,
    int startIndex,
    int count
);
```

Remarks

On return from this method, the data may be overwritten, so grab it while you can. You cannot assume that startIndex will be zero.

Requirements

Namespace: DotZLib

Assembly: DotZLib (in DotZLib.dll)

See Also

DotZLib Namespace

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
Deflater Class

Implements a data compressor, using the deflate algorithm in the ZLib dll

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

System.Object CodecBase

Deflater

```csharp
public sealed class Deflater : CodecBase
```

Requirements

Namespace: DotZLib

Assembly: DotZLib (in DotZLib.dll)

See Also

Deflater Members | DotZLib Namespace

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
### Deflater Members

#### Deflater overview

**Public Instance Constructors**

<table>
<thead>
<tr>
<th>Constructor</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deflater Constructor</td>
<td>Constructs an new instance of the Deflater</td>
</tr>
</tbody>
</table>

**Public Instance Properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Checksum</td>
<td>Gets the checksum of the data that has been added so far</td>
</tr>
</tbody>
</table>

**Public Instance Methods**

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add</td>
<td>Overloaded. Adds more data to the codec to be processed.</td>
</tr>
<tr>
<td>Add (inherited from CodecBase)</td>
<td>Overloaded. Adds more data to the codec to be processed.</td>
</tr>
<tr>
<td>Dispose (inherited from CodecBase)</td>
<td>Releases any unmanaged resources and calls the CleanUp method of the derived class</td>
</tr>
<tr>
<td>Equals (inherited from Object)</td>
<td>Determines whether the specified Object is equal to the current Object.</td>
</tr>
<tr>
<td>Finish</td>
<td>Finishes up any pending data that needs to be processed and handled.</td>
</tr>
<tr>
<td>GetHashCode (inherited from Object)</td>
<td>Serves as a hash function for a particular type, 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><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>-----------------------------------------</td>
<td>----------------------------------------------------------------</td>
</tr>
</tbody>
</table>

### Public Instance Events

<table>
<thead>
<tr>
<th><strong>DataAvailable</strong> (inherited from <strong>CodecBase</strong>)</th>
<th>Occurs when more processed data are available.</th>
</tr>
</thead>
</table>

### Protected Instance Fields

<table>
<thead>
<tr>
<th><strong>_isDisposed</strong> (inherited from <strong>CodecBase</strong>)</th>
<th>True if the object instance has been disposed, false otherwise</th>
</tr>
</thead>
</table>

### Protected Instance Methods

<table>
<thead>
<tr>
<th><strong>CleanUp</strong></th>
<th>Closes the internal zlib deflate stream</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>copyInput</strong> (inherited from <strong>CodecBase</strong>)</td>
<td>Copies a number of bytes to the internal codec buffer - ready for processing</td>
</tr>
<tr>
<td><strong>Finalize</strong> (inherited from <strong>CodecBase</strong>)</td>
<td>Destroys this instance</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>OnDataAvailable</strong> (inherited from <strong>CodecBase</strong>)</td>
<td>Fires the <strong>DataAvailable</strong> event</td>
</tr>
<tr>
<td><strong>resetOutput</strong> (inherited from <strong>CodecBase</strong>)</td>
<td>Resets the internal output buffers to a known state - ready for processing</td>
</tr>
<tr>
<td><strong>setChecksum</strong> (inherited from <strong>CodecBase</strong>)</td>
<td>Updates the running checksum property</td>
</tr>
</tbody>
</table>

### See Also

- [Deflater Class](#) | [DotZLib Namespace](#)
DotZLib .Net wrapper for ZLib1.dll
Deflater Constructor

Constructs an new instance of the Deflater

```java
public Deflater(
    CompressLevel level
);
```

Parameters

level
The compression level to use for this Deflater

See Also

Deflater Class | DotZLib Namespace

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
**Deflater Methods**

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

### Public Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Add</strong></td>
<td>Overloaded. Adds more data to the codec to be processed.</td>
</tr>
<tr>
<td><strong>Add</strong> (inherited from CodecBase)</td>
<td>Overloaded. Adds more data to the codec to be processed.</td>
</tr>
<tr>
<td><strong>Dispose</strong> (inherited from CodecBase)</td>
<td>Releases any unmanaged resources and calls the <strong>CleanUp</strong> method of the derived class.</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>Finish</strong></td>
<td>Finishes up any pending data that needs to be processed and handled.</td>
</tr>
<tr>
<td><strong>GetHashCode</strong> (inherited from <strong>Object</strong>)</td>
<td>Serves as a hash function for a particular type, 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>Object</strong>)</td>
<td>Returns a <strong>String</strong> that represents the current <strong>Object</strong>.</td>
</tr>
</tbody>
</table>

### Protected Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CleanUp</strong></td>
<td>Closes the internal zlib deflate stream</td>
</tr>
<tr>
<td><strong>copyInput</strong> (inherited from CodecBase)</td>
<td>Copies a number of bytes to the internal codec buffer - ready for</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>---------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>Finalize</strong> (inherited from CodecBase)</td>
<td>Destroys this instance</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong> (inherited from Object)</td>
<td>Creates a shallow copy of the current <strong>Object</strong>.</td>
</tr>
<tr>
<td><strong>OnDataAvailable</strong> (inherited from CodecBase)</td>
<td>Fires the <strong>DataAvailable</strong> event</td>
</tr>
<tr>
<td><strong>resetOutput</strong> (inherited from CodecBase)</td>
<td>Resets the internal output buffers to a known state - ready for processing</td>
</tr>
<tr>
<td><strong>setChecksum</strong> (inherited from CodecBase)</td>
<td>Updates the running checksum property</td>
</tr>
</tbody>
</table>

See Also

[Deflater Class] | [DotZLib Namespace]

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
Deflater.Add Method

Adds more data to the codec to be processed.

Overload List

Inherited from CodecBase.

  public void Add(byte[]);

Adds more data to the codec to be processed.

  public override void Add(byte[], int, int);

See Also

Deflater Class | DotZLib Namespace

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
Deflater.Add Method (Byte[], Int32, Int32)

Adds more data to the codec to be processed.

```
public override void Add(
    byte[] data, int offset, int count);
```

Parameters

* data
  Byte array containing the data to be added to the codec

* offset
  The index of the first byte to add from data

* count
  The number of bytes to add

Implements

Codec.Add

Remarks

Adding data may, or may not, raise the DataAvailable event

See Also

Deflater Class | DotZLib Namespace | Deflater.Add Overload List

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
Deflater.CleanUp Method

Closes the internal zlib deflate stream

```csharp
protected override void CleanUp();
```

See Also

[Deflater Class](#) | [DotZLib Namespace](#)

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
Deflater.Finish Method

Finishes up any pending data that needs to be processed and handled.

```csharp
public override void Finish();
```

Implements

- `Codec.Finish`

See Also

- `Deflater Class` | `DotZLib Namespace`

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
GZipStream Class

Implements a compressed Stream, in GZip (.gz) format.
For a list of all members of this type, see GZipStream Members.

System.Object MarshalByRefObject Stream
GZipStream

public class GZipStream : Stream

Requirements

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

See Also

GZipStream Members | DotZLib Namespace

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
## GZipStream Members

### GZipStream overview

### Public Instance Constructors

<table>
<thead>
<tr>
<th>GZipStream</th>
<th>Overloaded. Initializes a new instance of the GZipStream class.</th>
</tr>
</thead>
</table>

### Public Instance Properties

<table>
<thead>
<tr>
<th>CanRead</th>
<th>Returns true of this stream can be read from, false otherwise</th>
</tr>
</thead>
<tbody>
<tr>
<td>CanSeek</td>
<td>Returns false.</td>
</tr>
<tr>
<td>CanWrite</td>
<td>Returns true if this stream is writeable, false otherwise</td>
</tr>
<tr>
<td>Length</td>
<td>Gets the size of the stream. Not supported.</td>
</tr>
<tr>
<td>Position</td>
<td>Gets/sets the current position in the GZipStream. Not supported.</td>
</tr>
</tbody>
</table>

### Public Instance Methods

<table>
<thead>
<tr>
<th>BeginRead (inherited from Stream)</th>
<th>Begins an asynchronous read operation.</th>
</tr>
</thead>
<tbody>
<tr>
<td>BeginWrite (inherited from Stream)</td>
<td>Begins an asynchronous write operation.</td>
</tr>
<tr>
<td>Close (inherited from Stream)</td>
<td>Closes the current stream and releases any resources (such as sockets and file handles) associated with the current stream.</td>
</tr>
<tr>
<td>CreateObjRef (inherited from MarshalByRefObject)</td>
<td>Creates an object that contains all the relevant information</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>Dispose</td>
<td>Closes the external file handle</td>
</tr>
<tr>
<td>EndRead (inherited from Stream)</td>
<td>Waits for the pending asynchronous read to complete.</td>
</tr>
<tr>
<td>EndWrite (inherited from Stream)</td>
<td>Ends an asynchronous write operation.</td>
</tr>
<tr>
<td>Equals (inherited from Object)</td>
<td>Determines whether the specified Object is equal to the current Object.</td>
</tr>
<tr>
<td>Flush</td>
<td>Flushes the GZipStream.</td>
</tr>
<tr>
<td>GetHashCode (inherited from Object)</td>
<td>Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table.</td>
</tr>
<tr>
<td>GetLifetimeService (inherited from MarshalByRefObject)</td>
<td>Retrieves the current lifetime service object that controls the lifetime policy for this instance.</td>
</tr>
<tr>
<td>GetType (inherited from Object)</td>
<td>Gets the Type of the current instance.</td>
</tr>
<tr>
<td>InitializeComponent (inherited from MarshalByRefObject)</td>
<td>Obtains a lifetime service object to control the lifetime policy for this instance.</td>
</tr>
<tr>
<td>Read</td>
<td>Attempts to read a number of bytes from the stream.</td>
</tr>
<tr>
<td>ReadByte</td>
<td>Attempts to read a single byte from the stream.</td>
</tr>
<tr>
<td>Seek</td>
<td>Not supported.</td>
</tr>
<tr>
<td>SetLength</td>
<td>Not supported.</td>
</tr>
<tr>
<td>ToString (inherited from Object)</td>
<td>Returns a String that represents the current Object.</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>-----------------------------------</td>
</tr>
<tr>
<td>Write</td>
<td>Writes a number of bytes to the stream</td>
</tr>
<tr>
<td>WriteByte</td>
<td>Writes a single byte to the stream</td>
</tr>
</tbody>
</table>

**Protected Instance Methods**

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CreateWaitHandle (inherited from Stream)</td>
<td>Allocates a <a href="#">WaitHandle</a> object.</td>
</tr>
<tr>
<td>Finalize</td>
<td>Destroys this instance</td>
</tr>
<tr>
<td>MemberwiseClone (inherited from Object)</td>
<td>Creates a shallow copy of the current <a href="#">Object</a>.</td>
</tr>
</tbody>
</table>

**Explicit Interface Implementations**

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IDisposable.Dispose (inherited from Stream)</td>
<td></td>
</tr>
</tbody>
</table>

**See Also**

[GZipStream Class](#) | [DotZLib Namespace](#)

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
GZipStream Constructor

Creates a new file as a writeable GZipStream

Overload List

Opens an existing file as a readable GZipStream

public GZipStream(string);

Creates a new file as a writeable GZipStream

public GZipStream(string, CompressLevel);

See Also

GZipStream Class | DotZLib Namespace

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
GZipStream Constructor (String, CompressLevel)

Creates a new file as a writeable GZipStream

```java
public GZipStream(
    string fileName,
    CompressLevel level
);
```

Parameters

- `fileName`
  - The name of the compressed file to create
- `level`
  - The compression level to use when adding data

Exceptions

<table>
<thead>
<tr>
<th>Exception Type</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZLibException</td>
<td>If an error occurred in the internal zlib function</td>
</tr>
</tbody>
</table>

See Also

- [GZipStream Class](#) | [DotZLib Namespace](#) | [GZipStream Constructor Overload List](#)

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
GZipStream Constructor (String)

Opens an existing file as a readable GZipStream

```java
public GZipStream(
    string fileName
);
```

Parameters

*fileName*

The name of the file to open

Exceptions

<table>
<thead>
<tr>
<th>Exception Type</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZLibException</td>
<td>If an error occurred in the internal zlib function</td>
</tr>
</tbody>
</table>

See Also

- GZipStream Class
- DotZLib Namespace
- GZipStream Constructor Overload List

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
The properties of the `GZipStream` class are listed below. For a complete list of `GZipStream` class members, see the `GZipStream Members` topic.

**Public Instance Properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>CanRead</code></td>
<td>Returns true if this stream can be read from, false otherwise</td>
</tr>
<tr>
<td><code>CanSeek</code></td>
<td>Returns false.</td>
</tr>
<tr>
<td><code>CanWrite</code></td>
<td>Returns true if this stream is writeable, false otherwise</td>
</tr>
<tr>
<td><code>Length</code></td>
<td>Gets the size of the stream. Not supported.</td>
</tr>
<tr>
<td><code>Position</code></td>
<td>Gets/sets the current position in the <code>GZipStream</code>. Not supported.</td>
</tr>
</tbody>
</table>

See Also

`GZipStream Class` | `DotZLib Namespace`
DotZLib .Net wrapper for ZLib1.dll
GZipStream.CanRead Property

Returns true of this stream can be read from, false otherwise

```csharp
public override bool CanRead {get;}
```

See Also

GZipStream Class | DotZLib Namespace

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
GZipStream.CanSeek Property

Returns false.

```csharp
public override bool CanSeek {get;}
```

See Also

[GZipStream Class](#) | [DotZLib Namespace](#)

---

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
GZipStream.CanWrite Property

Returns true if this tsream is writeable, false otherwise

```csharp
public override bool CanWrite {get;}
```

See Also

GZipStream Class | DotZLib Namespace

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
GZipStream.Length Property

Gets the size of the stream. Not supported.

```csharp
public override long Length {get;}
```

Remarks

In this implementation this property is not supported

Exceptions

<table>
<thead>
<tr>
<th>Exception Type</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>NotSupportedException</td>
<td>Always thrown</td>
</tr>
</tbody>
</table>

See Also

GZipStream Class | DotZLib Namespace

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
GZipStream.Position Property

Gets/sets the current position in the GZipStream. Not supported.

```csharp
public override long Position {get; set;}
```

Remarks

In this implementation this property is not supported

Exceptions

<table>
<thead>
<tr>
<th>Exception Type</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>NotSupportedException</td>
<td>Always thrown</td>
</tr>
</tbody>
</table>

See Also

GZipStream Class | DotZLib Namespace

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
GZipStream Methods

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

Public Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BeginRead (inherited from Stream)</td>
<td>Begins an asynchronous read operation.</td>
</tr>
<tr>
<td>BeginWrite (inherited from Stream)</td>
<td>Begins an asynchronous write operation.</td>
</tr>
<tr>
<td>Close (inherited from Stream)</td>
<td>Closes the current stream and releases any resources (such as sockets and file handles) associated with the current stream.</td>
</tr>
<tr>
<td>CreateObjRef (inherited from MarshalByRefObject)</td>
<td>Creates an object that contains all the relevant information required to generate a proxy used to communicate with a remote object.</td>
</tr>
<tr>
<td>Dispose</td>
<td>Closes the external file handle</td>
</tr>
<tr>
<td>EndRead (inherited from Stream)</td>
<td>Waits for the pending asynchronous read to complete.</td>
</tr>
<tr>
<td>EndWrite (inherited from Stream)</td>
<td>Ends an asynchronous write operation.</td>
</tr>
<tr>
<td>Equals (inherited from Object)</td>
<td>Determines whether the specified Object is equal to the current Object.</td>
</tr>
<tr>
<td>Flush</td>
<td>Flushes the GZipStream.</td>
</tr>
<tr>
<td>GetHashCode (inherited from Object)</td>
<td>Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table.</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>GetLifetimeService</strong></td>
<td>(inherited from MarshalByRefObject) Retrieves the current lifetime service object that controls the lifetime policy for this instance.</td>
</tr>
<tr>
<td><strong>GetTypeInfo</strong></td>
<td>(inherited from Object) Gets the <strong>Type</strong> of the current instance.</td>
</tr>
<tr>
<td><strong>InitializeLifetimeService</strong></td>
<td>(inherited from MarshalByRefObject) Obtains a lifetime service object to control the lifetime policy for this instance.</td>
</tr>
<tr>
<td><strong>Read</strong></td>
<td>Attempts to read a number of bytes from the stream.</td>
</tr>
<tr>
<td><strong>ReadByte</strong></td>
<td>Attempts to read a single byte from the stream.</td>
</tr>
<tr>
<td><strong>Seek</strong></td>
<td>Not supported.</td>
</tr>
<tr>
<td><strong>SetLength</strong></td>
<td>Not supported.</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>(inherited from Object) Returns a <strong>String</strong> that represents the current <strong>Object</strong>.</td>
</tr>
<tr>
<td><strong>Write</strong></td>
<td>Writes a number of bytes to the stream</td>
</tr>
<tr>
<td><strong>WriteByte</strong></td>
<td>Writes a single byte to the stream</td>
</tr>
</tbody>
</table>

**Protected Instance Methods**

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CreateWaitHandle</strong></td>
<td>(inherited from Stream) Allocates a <strong>WaitHandle</strong> object.</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Destroys this instance</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>(inherited from Object) Creates a shallow copy of the current <strong>Object</strong>.</td>
</tr>
</tbody>
</table>

**Explicit Interface Implementations**

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>IDisposable.Dispose</strong></td>
<td>(inherited from Stream)</td>
</tr>
</tbody>
</table>
See Also

GZipStream Class | DotZLib Namespace

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
GZipStream.Dispose Method

Closes the external file handle

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

Implements

[IDisposable.Dispose]

See Also

[GZipStream Class] [DotZLib Namespace]

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
GZipStream.Finalize Method

Destroys this instance

```
protected override void Finalize();
```

See Also

[GZipStream Class] | [DotZLib Namespace]

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
GZipStream.Flush Method

Flushes the GZipStream.

```csharp
public override void Flush();
```

Remarks

In this implementation, this method does nothing. This is because excessive flushing may degrade the achievable compression rates.

See Also

GZipStream Class | DotZLib Namespace

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
GZipStream.Read Method

Attempts to read a number of bytes from the stream.

```csharp
public override int Read(
    byte[] buffer,
    int offset,
    int count
);
```

Parameters

- **buffer**
  - The destination data buffer

- **offset**
  - The index of the first destination byte in buffer

- **count**
  - The number of bytes requested

Return Value

The number of bytes read

Exceptions

<table>
<thead>
<tr>
<th>Exception Type</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ArgumentNullException</td>
<td>If buffer is null</td>
</tr>
<tr>
<td>ArgumentOutOfRangeException</td>
<td>If count or offset are negative</td>
</tr>
<tr>
<td>ArgumentException</td>
<td>If offset + count is &gt; buffer.Length</td>
</tr>
<tr>
<td>NotSupportedException</td>
<td>If this stream is not readable.</td>
</tr>
<tr>
<td>ObjectDisposedException</td>
<td>If this stream has been disposed.</td>
</tr>
</tbody>
</table>

See Also
DotZLib .Net wrapper for ZLib1.dll
GZipStream.ReadByte Method

Attempts to read a single byte from the stream.

```csharp
public override int ReadByte();
```

Return Value

The byte that was read, or -1 in case of error or End-Of-File

See Also

GZipStream Class | DotZLib Namespace

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
GZipStream.Seek Method

Not supported.

```csharp
public override long Seek(
    long offset,
    SeekOrigin origin
);
```

Parameters

- `offset`
- `origin`

Return Value

Exceptions

<table>
<thead>
<tr>
<th>Exception Type</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>NotSupportedException</td>
<td>Always thrown</td>
</tr>
</tbody>
</table>

See Also

- [GZipStream Class](#) | [DotZLib Namespace](#)

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
GZipStream.SetLength Method

Not supported.

```csharp
public override void SetLength(
    long value
);
```

Parameters

- `value`

Exceptions

<table>
<thead>
<tr>
<th>Exception Type</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>NotSupportedException</td>
<td>Always thrown</td>
</tr>
</tbody>
</table>

See Also

- GZipStream Class
- DotZLib Namespace

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
GZipStream.Write Method

Writes a number of bytes to the stream

```csharp
public override void Write(
    byte[] buffer,
    int offset,
    int count
);
```

Parameters

- `buffer`
- `offset`
- `count`

Exceptions

<table>
<thead>
<tr>
<th>Exception Type</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ArgumentNullException</td>
<td>If <code>buffer</code> is null</td>
</tr>
<tr>
<td>ArgumentOutOfRangeException</td>
<td>If <code>count</code> or <code>offset</code> are negative</td>
</tr>
<tr>
<td>ArgumentException</td>
<td>If <code>offset + count</code> is &gt; <code>buffer.Length</code></td>
</tr>
<tr>
<td>NotSupportedException</td>
<td>If this stream is not writeable.</td>
</tr>
<tr>
<td>ObjectDisposedException</td>
<td>If this stream has been disposed.</td>
</tr>
</tbody>
</table>

See Also

- GZipStream Class | DotZLib Namespace

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
GZipStream.WriteByte Method

Writes a single byte to the stream

```csharp
public override void WriteByte(
    byte value
);
```

Parameters

`value`

The byte to add to the stream.

Exceptions

<table>
<thead>
<tr>
<th>Exception Type</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>NotSupportedException</td>
<td>If this stream is not writeable.</td>
</tr>
<tr>
<td>ObjectDisposedException</td>
<td>If this stream has been disposed.</td>
</tr>
</tbody>
</table>

See Also

GZipStream Class | DotZLib Namespace

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
**Inflater Class**

Implements a data decompressor, using the inflate algorithm in the ZLib dll.

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

*System.Object*  *CodecBase*

**Inflater**

```csharp
public class Inflater : CodecBase
```

**Requirements**

**Namespace:** DotZLib  
**Assembly:** DotZLib (in DotZLib.dll)

**See Also**

Inflater Members | DotZLib Namespace

---

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
# Inflater Members

## Inflater overview

## Public Instance Constructors

<table>
<thead>
<tr>
<th>Constructor</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Inflater Constructor</strong></td>
<td>Constructs a new instance of the <strong>Inflater</strong></td>
</tr>
</tbody>
</table>

## Public Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Checksum</strong> (inherited from <strong>CodecBase</strong>)</td>
<td>Gets the checksum of the data that has been added so far</td>
</tr>
</tbody>
</table>

## Public Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Add</strong></td>
<td>Overloaded. Adds more data to the codec to be processed.</td>
</tr>
<tr>
<td><strong>Add</strong> (inherited from <strong>CodecBase</strong>)</td>
<td>Overloaded. Adds more data to the codec to be processed.</td>
</tr>
<tr>
<td><strong>Dispose</strong> (inherited from <strong>CodecBase</strong>)</td>
<td>Releases any unmanaged resources and calls the <strong>CleanUp</strong> method of the derived class</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>Finish</strong></td>
<td>Finishes up any pending data that needs to be processed and handled.</td>
</tr>
<tr>
<td><strong>GetHashCode</strong> (inherited from <strong>Object</strong>)</td>
<td>Serves as a hash function for a particular type, 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>
</tbody>
</table>
**Public Instance Events**

<table>
<thead>
<tr>
<th>Event</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DataAvailable (inherited from CodecBase)</td>
<td>Occurs when more processed data are available.</td>
</tr>
</tbody>
</table>

**Protected Instance Fields**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>isDisposed (inherited from CodecBase)</td>
<td>True if the object instance has been disposed, false otherwise</td>
</tr>
</tbody>
</table>

**Protected Instance Methods**

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CleanUp</td>
<td>Closes the internal zlib inflate stream</td>
</tr>
<tr>
<td>copyInput (inherited from CodecBase)</td>
<td>Copies a number of bytes to the internal codec buffer - ready for processing</td>
</tr>
<tr>
<td>Finalize (inherited from CodecBase)</td>
<td>Destroys this instance</td>
</tr>
<tr>
<td>MemberwiseClone (inherited from Object)</td>
<td>Creates a shallow copy of the current Object.</td>
</tr>
<tr>
<td>OnDataAvailable (inherited from CodecBase)</td>
<td>Fires the DataAvailable event</td>
</tr>
<tr>
<td>resetOutput (inherited from CodecBase)</td>
<td>Resets the internal output buffers to a known state - ready for processing</td>
</tr>
<tr>
<td>setChecksum (inherited from CodecBase)</td>
<td>Updates the running checksum property</td>
</tr>
</tbody>
</table>

**See Also**

Inflater Class | DotZLib Namespace

---

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
Inflater Constructor

Constructs an new instance of the Inflater

```java
public Inflater();
```

See Also

Inflater Class | DotZLib Namespace

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
# Inflater Methods

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

## Public Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Add</strong></td>
<td>Overloaded. Adds more data to the codec to be processed.</td>
</tr>
<tr>
<td><strong>Add</strong> (inherited from <strong>CodecBase</strong>)</td>
<td>Overloaded. Adds more data to the codec to be processed.</td>
</tr>
<tr>
<td><strong>Dispose</strong> (inherited from <strong>CodecBase</strong>)</td>
<td>Releases any unmanaged resources and calls the <strong>CleanUp</strong> method of the derived class.</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>Finish</strong></td>
<td>Finishes up any pending data that needs to be processed and handled.</td>
</tr>
<tr>
<td><strong>GetHashCode</strong> (inherited from <strong>Object</strong>)</td>
<td>Serves as a hash function for a particular type, 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>Object</strong>)</td>
<td>Returns a <strong>String</strong> that represents the current <strong>Object</strong>.</td>
</tr>
</tbody>
</table>

## Protected Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CleanUp</strong></td>
<td>Closes the internal zlib inflate stream.</td>
</tr>
</tbody>
</table>
| **copyInput** (inherited from **CodecBase**) | Copies a number of bytes to the internal codec buffer - ready for }
<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Finalize</td>
<td>(inherited from CodecBase) Destroys this instance</td>
</tr>
<tr>
<td>MemberwiseClone</td>
<td>(inherited from Object) Creates a shallow copy of the current Object.</td>
</tr>
<tr>
<td>OnDataAvailable</td>
<td>(inherited from CodecBase) Fires the DataAvailable event</td>
</tr>
<tr>
<td>resetOutput</td>
<td>(inherited from CodecBase) Resets the internal output buffers to a known state - ready for processing</td>
</tr>
<tr>
<td>setChecksum</td>
<td>(inherited from CodecBase) Updates the running checksum property</td>
</tr>
</tbody>
</table>

**See Also**

Inflater Class | DotZLib Namespace

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
**Inflater.Add Method**

Adds more data to the codec to be processed.

**Overload List**

Inherited from [CodecBase](#).

```csharp
public void Add(byte[]);
```

Adds more data to the codec to be processed.

```csharp
public override void Add(byte[], int, int);
```

**See Also**

[Inflater Class](#) | [DotZLib Namespace](#)

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
**Inflater.Add Method (Byte[], Int32, Int32)**

Adds more data to the codec to be processed.

```csharp
public override void Add(
    byte[] data,
    int offset,
    int count
);
```

**Parameters**

- **data**
  Byte array containing the data to be added to the codec

- **offset**
  The index of the first byte to add from `data`

- **count**
  The number of bytes to add

**Implements**

- Codec.Add

**Remarks**

Adding data may, or may not, raise the `DataAvailable` event

**See Also**

- Inflater Class  |  DotZLib Namespace  |  Inflater.Add Overload List

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
Inflater.CleanUp Method

Closes the internal zlib inflate stream

```csharp
protected override void CleanUp();
```

See Also

- Inflater Class
- DotZLib Namespace

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
Inflater.Finish Method

Finishes up any pending data that needs to be processed and handled.

```
public override void Finish();
```

Implements

Codec.Finish

See Also

Inflater Class | DotZLib Namespace

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
**Info Class**

Encapsulates general information about the ZLib library

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

```csharp
System.Object Info
```

```csharp
public class Info
```

**Requirements**

**Namespace:** DotZLib

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

**See Also**

[Info Members](link) | [DotZLib Namespace](link)

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
### Info Members

#### Info overview

#### Public Static Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>Version</code></td>
<td>Gets the version of ZLib as a string, e.g. &quot;1.2.1&quot;</td>
</tr>
</tbody>
</table>

#### Public Instance Constructors

<table>
<thead>
<tr>
<th>Constructor</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>Info Constructor</code></td>
<td>Constructs an instance of the <code>Info</code> class.</td>
</tr>
</tbody>
</table>

#### Public Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>HasDebugInfo</code></td>
<td>True if the library is compiled with debug info</td>
</tr>
<tr>
<td><code>SizeOfOffset</code></td>
<td>Gets the size of the <code>z_off_t</code> type that was compiled into Zlib</td>
</tr>
<tr>
<td><code>SizeOfPointer</code></td>
<td>Gets the size of the pointers that were compiled into Zlib</td>
</tr>
<tr>
<td><code>SizeOfUInt</code></td>
<td>Gets the size of the unsigned int that was compiled into Zlib</td>
</tr>
<tr>
<td><code>SizeOfULong</code></td>
<td>Gets the size of the unsigned long that was compiled into Zlib</td>
</tr>
<tr>
<td><code>UsesAssemblyCode</code></td>
<td>True if the library is compiled with assembly optimizations</td>
</tr>
</tbody>
</table>

#### Public Instance Methods

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

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<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 Object)</td>
<td>Returns a <strong>String</strong> that represents the current <strong>Object</strong>.</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 <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 Object)</td>
<td>Creates a shallow copy of the current <strong>Object</strong>.</td>
</tr>
</tbody>
</table>

See Also

[Info Class](#) | [DotZLib Namespace](#)

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
**Info Constructor**

Constructs an instance of the `Info` class.

```java
public Info();
```

**See Also**

[Info Class](#) | [DotZLib Namespace](#)

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
## Info Properties

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

### Public Static Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Version</strong></td>
<td>Gets the version of ZLib as a string, e.g. &quot;1.2.1&quot;</td>
</tr>
</tbody>
</table>

### Public Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>HasDebugInfo</strong></td>
<td>True if the library is compiled with debug info</td>
</tr>
<tr>
<td><strong>SizeOfOffset</strong></td>
<td>Gets the size of the <code>z_off_t</code> type that was compiled into Zlib</td>
</tr>
<tr>
<td><strong>SizeOfPointer</strong></td>
<td>Gets the size of the pointers that were compiled into Zlib</td>
</tr>
<tr>
<td><strong>SizeOfUInt</strong></td>
<td>Gets the size of the unsigned int that was compiled into Zlib</td>
</tr>
<tr>
<td><strong>SizeOfULong</strong></td>
<td>Gets the size of the unsigned long that was compiled into Zlib</td>
</tr>
<tr>
<td><strong>UsesAssemblyCode</strong></td>
<td>True if the library is compiled with assembly optimizations</td>
</tr>
</tbody>
</table>

### See Also

[Info Class](#) | [DotZLib Namespace](#)

---

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
Info.HasDebugInfo Property

True if the library is compiled with debug info

```
public bool HasDebugInfo {get;}
```

See Also

[Info Class](#) | [DotZLib Namespace](#)

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
Info.SizeOfOffset Property

Gets the size of the z_off_t type that was compiled into Zlib

```csharp
public int SizeOfOffset {get;}
```

See Also

[Info Class](#) | [DotZLib Namespace](#)

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
Info.SizeOfPointer Property

Gets the size of the pointers that were compiled into Zlib

public int SizeOfPointer {get;}

See Also

Info Class | DotZLib Namespace

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
Info.SizeOfUInt Property

Gets the size of the unsigned int that was compiled into Zlib

```csharp
public int SizeOfUInt {get;}
```

See Also

Info Class | DotZLib Namespace

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
Info.SizeOfULong Property

Gets the size of the unsigned long that was compiled into Zlib

```csharp
public int SizeOfULong {get;}
```

See Also

Info Class | DotZLib Namespace

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
Info.UsesAssemblyCode Property

True if the library is compiled with assembly optimizations

```csharp
public bool UsesAssemblyCode {get;}
```

See Also

- Info Class
- DotZLib Namespace

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
Info.Version Property

Gets the version of ZLib as a string, e.g. "1.2.1"

```
public static string Version {get;}
```

See Also

[Info Class](#) | [DotZLib Namespace](#)

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
ZLibException Class

The exception that is thrown when an error occurs on the zlib dll
For a list of all members of this type, see ZLibException Members.

See Also

ZLibException Members | DotZLib Namespace

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib.dll
## ZLibException Members

**ZLibException overview**

### Public Instance Constructors

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZLibException</td>
<td>Overloaded. Initializes a new instance of the ZLibException class.</td>
</tr>
</tbody>
</table>

### Public Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HelpLink</td>
<td>Gets or sets a link to the help file associated with this exception.</td>
</tr>
<tr>
<td>InnerException</td>
<td>Gets the Exception instance that caused the current exception.</td>
</tr>
<tr>
<td>Message</td>
<td>Gets a message that describes the current exception.</td>
</tr>
<tr>
<td>Source</td>
<td>Gets or sets the name of the application or the object that causes the error.</td>
</tr>
<tr>
<td>StackTrace</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>TargetSite</td>
<td>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>Equals</td>
<td>Determines whether the specified Object is equal to the current Object.</td>
</tr>
<tr>
<td>GetBaseException</td>
<td>When overridden in a derived class, returns the Exception that</td>
</tr>
</tbody>
</table>
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, suitable for use in hashing algorithms and data structures like a hash table.</td>
</tr>
<tr>
<td><strong>GetObjectData</strong> (inherited from Exception)</td>
<td>When overridden in a derived class, sets the <code>SerializationInfo</code> with information about the exception.</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 Exception)</td>
<td>Creates and returns a string representation of 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 to a specific 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 <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> (inherited from Object)</td>
<td>Creates a shallow copy of the current <code>Object</code>.</td>
</tr>
</tbody>
</table>

### See Also

- ZLibException Class  | DotZLib Namespace  

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
ZLibException Constructor

Initializes a new instance of the ZLibException class with a specified error message and error code

Overload List

Initializes a new instance of the ZLibException class with a specified error code

public ZLibException(int);

Initializes a new instance of the ZLibException class with a specified error message and error code

public ZLibException(int,string);

See Also

ZLibException Class | DotZLib Namespace

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
ZLibException Constructor (Int32, String)

Initializes a new instance of the ZLibException class with a specified error message and error code

```java
public ZLibException(
    int errorCode,
    string msg
);
```

Parameters

- `errorCode`
  The zlib error code that caused the exception

- `msg`
  A message that (hopefully) describes the error

See Also

- ZLibException Class | DotZLib Namespace | ZLibException Constructor Overload List

Copyright 2004 Henrik Ravn
DotZLib .Net wrapper for ZLib1.dll
ZLibException Constructor (Int32)

Initializes a new instance of the ZLibException class with a specified error code

```csharp
public ZLibException(int errorCode);
```

Parameters

`errorCode`

The zlib error code that caused the exception

See Also

ZLibException Class | DotZLib Namespace | ZLibException Constructor Overload List

Copyright 2004 Henrik Ravn