| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/Inflater.html) | [**Tree**](http://docs.google.com/package-tree.html) | [**Deprecated**](http://docs.google.com/deprecated-list.html) | [**Index**](http://docs.google.com/index-files/index-1.html) | [**Help**](http://docs.google.com/help-doc.html) | | --- | --- | --- | --- | --- | --- | --- | --- | | | ***Java™ Platform***  ***Standard Ed. 6*** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [**PREV CLASS**](http://docs.google.com/java/util/zip/GZIPOutputStream.html)   [**NEXT CLASS**](http://docs.google.com/java/util/zip/InflaterInputStream.html) | [**FRAMES**](http://docs.google.com/index.html?java/util/zip/Inflater.html)    [**NO FRAMES**](http://docs.google.com/Inflater.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | [CONSTR](#3znysh7) | [METHOD](#2et92p0) | DETAIL: FIELD | [CONSTR](#3dy6vkm) | [METHOD](#2s8eyo1) |

## **java.util.zip**

Class Inflater

[java.lang.Object](http://docs.google.com/java/lang/Object.html)  
 **java.util.zip.Inflater**

public class **Inflater**extends [Object](http://docs.google.com/java/lang/Object.html)

This class provides support for general purpose decompression using the popular ZLIB compression library. The ZLIB compression library was initially developed as part of the PNG graphics standard and is not protected by patents. It is fully described in the specifications at the [java.util.zip package description](http://docs.google.com/package-summary.html#package_description).

The following code fragment demonstrates a trivial compression and decompression of a string using Deflater and Inflater.

try {  
 // Encode a String into bytes  
 String inputString = "blahblahblah??";  
 byte[] input = inputString.getBytes("UTF-8");  
  
 // Compress the bytes  
 byte[] output = new byte[100];  
 Deflater compresser = new Deflater();  
 compresser.setInput(input);  
 compresser.finish();  
 int compressedDataLength = compresser.deflate(output);  
  
 // Decompress the bytes  
 Inflater decompresser = new Inflater();  
 decompresser.setInput(output, 0, compressedDataLength);  
 byte[] result = new byte[100];  
 int resultLength = decompresser.inflate(result);  
 decompresser.end();  
  
 // Decode the bytes into a String  
 String outputString = new String(result, 0, resultLength, "UTF-8");  
 } catch(java.io.UnsupportedEncodingException ex) {  
 // handle  
 } catch (java.util.zip.DataFormatException ex) {  
 // handle  
 }

**See Also:**[Deflater](http://docs.google.com/java/util/zip/Deflater.html)

| **Constructor Summary** | |
| --- | --- |
| [**Inflater**](http://docs.google.com/java/util/zip/Inflater.html#Inflater())()            Creates a new decompressor. |
| [**Inflater**](http://docs.google.com/java/util/zip/Inflater.html#Inflater(boolean))(boolean nowrap)            Creates a new decompressor. |

| **Method Summary** | |
| --- | --- |
| void | [**end**](http://docs.google.com/java/util/zip/Inflater.html#end())()            Closes the decompressor and discards any unprocessed input. |
| protected  void | [**finalize**](http://docs.google.com/java/util/zip/Inflater.html#finalize())()            Closes the decompressor when garbage is collected. |
| boolean | [**finished**](http://docs.google.com/java/util/zip/Inflater.html#finished())()            Returns true if the end of the compressed data stream has been reached. |
| int | [**getAdler**](http://docs.google.com/java/util/zip/Inflater.html#getAdler())()            Returns the ADLER-32 value of the uncompressed data. |
| long | [**getBytesRead**](http://docs.google.com/java/util/zip/Inflater.html#getBytesRead())()            Returns the total number of compressed bytes input so far. |
| long | [**getBytesWritten**](http://docs.google.com/java/util/zip/Inflater.html#getBytesWritten())()            Returns the total number of uncompressed bytes output so far. |
| int | [**getRemaining**](http://docs.google.com/java/util/zip/Inflater.html#getRemaining())()            Returns the total number of bytes remaining in the input buffer. |
| int | [**getTotalIn**](http://docs.google.com/java/util/zip/Inflater.html#getTotalIn())()            Returns the total number of compressed bytes input so far. |
| int | [**getTotalOut**](http://docs.google.com/java/util/zip/Inflater.html#getTotalOut())()            Returns the total number of uncompressed bytes output so far. |
| int | [**inflate**](http://docs.google.com/java/util/zip/Inflater.html#inflate(byte%5B%5D))(byte[] b)            Uncompresses bytes into specified buffer. |
| int | [**inflate**](http://docs.google.com/java/util/zip/Inflater.html#inflate(byte%5B%5D,%20int,%20int))(byte[] b, int off, int len)            Uncompresses bytes into specified buffer. |
| boolean | [**needsDictionary**](http://docs.google.com/java/util/zip/Inflater.html#needsDictionary())()            Returns true if a preset dictionary is needed for decompression. |
| boolean | [**needsInput**](http://docs.google.com/java/util/zip/Inflater.html#needsInput())()            Returns true if no data remains in the input buffer. |
| void | [**reset**](http://docs.google.com/java/util/zip/Inflater.html#reset())()            Resets inflater so that a new set of input data can be processed. |
| void | [**setDictionary**](http://docs.google.com/java/util/zip/Inflater.html#setDictionary(byte%5B%5D))(byte[] b)            Sets the preset dictionary to the given array of bytes. |
| void | [**setDictionary**](http://docs.google.com/java/util/zip/Inflater.html#setDictionary(byte%5B%5D,%20int,%20int))(byte[] b, int off, int len)            Sets the preset dictionary to the given array of bytes. |
| void | [**setInput**](http://docs.google.com/java/util/zip/Inflater.html#setInput(byte%5B%5D))(byte[] b)            Sets input data for decompression. |
| void | [**setInput**](http://docs.google.com/java/util/zip/Inflater.html#setInput(byte%5B%5D,%20int,%20int))(byte[] b, int off, int len)            Sets input data for decompression. |

| **Methods inherited from class java.lang.**[**Object**](http://docs.google.com/java/lang/Object.html) |
| --- |
| [clone](http://docs.google.com/java/lang/Object.html#clone()), [equals](http://docs.google.com/java/lang/Object.html#equals(java.lang.Object)), [getClass](http://docs.google.com/java/lang/Object.html#getClass()), [hashCode](http://docs.google.com/java/lang/Object.html#hashCode()), [notify](http://docs.google.com/java/lang/Object.html#notify()), [notifyAll](http://docs.google.com/java/lang/Object.html#notifyAll()), [toString](http://docs.google.com/java/lang/Object.html#toString()), [wait](http://docs.google.com/java/lang/Object.html#wait()), [wait](http://docs.google.com/java/lang/Object.html#wait(long)), [wait](http://docs.google.com/java/lang/Object.html#wait(long,%20int)) |

| **Constructor Detail** |
| --- |

### Inflater

public **Inflater**(boolean nowrap)

Creates a new decompressor. If the parameter 'nowrap' is true then the ZLIB header and checksum fields will not be used. This provides compatibility with the compression format used by both GZIP and PKZIP.

Note: When using the 'nowrap' option it is also necessary to provide an extra "dummy" byte as input. This is required by the ZLIB native library in order to support certain optimizations.

**Parameters:**nowrap - if true then support GZIP compatible compression

### Inflater

public **Inflater**()

Creates a new decompressor.

| **Method Detail** |
| --- |

### setInput

public void **setInput**(byte[] b,  
 int off,  
 int len)

Sets input data for decompression. Should be called whenever needsInput() returns true indicating that more input data is required.

**Parameters:**b - the input data bytesoff - the start offset of the input datalen - the length of the input data**See Also:**[needsInput()](http://docs.google.com/java/util/zip/Inflater.html#needsInput())

### setInput

public void **setInput**(byte[] b)

Sets input data for decompression. Should be called whenever needsInput() returns true indicating that more input data is required.

**Parameters:**b - the input data bytes**See Also:**[needsInput()](http://docs.google.com/java/util/zip/Inflater.html#needsInput())

### setDictionary

public void **setDictionary**(byte[] b,  
 int off,  
 int len)

Sets the preset dictionary to the given array of bytes. Should be called when inflate() returns 0 and needsDictionary() returns true indicating that a preset dictionary is required. The method getAdler() can be used to get the Adler-32 value of the dictionary needed.

**Parameters:**b - the dictionary data bytesoff - the start offset of the datalen - the length of the data**See Also:**[needsDictionary()](http://docs.google.com/java/util/zip/Inflater.html#needsDictionary()), [getAdler()](http://docs.google.com/java/util/zip/Inflater.html#getAdler())

### setDictionary

public void **setDictionary**(byte[] b)

Sets the preset dictionary to the given array of bytes. Should be called when inflate() returns 0 and needsDictionary() returns true indicating that a preset dictionary is required. The method getAdler() can be used to get the Adler-32 value of the dictionary needed.

**Parameters:**b - the dictionary data bytes**See Also:**[needsDictionary()](http://docs.google.com/java/util/zip/Inflater.html#needsDictionary()), [getAdler()](http://docs.google.com/java/util/zip/Inflater.html#getAdler())

### getRemaining

public int **getRemaining**()

Returns the total number of bytes remaining in the input buffer. This can be used to find out what bytes still remain in the input buffer after decompression has finished.

**Returns:**the total number of bytes remaining in the input buffer

### needsInput

public boolean **needsInput**()

Returns true if no data remains in the input buffer. This can be used to determine if #setInput should be called in order to provide more input.

**Returns:**true if no data remains in the input buffer

### needsDictionary

public boolean **needsDictionary**()

Returns true if a preset dictionary is needed for decompression.

**Returns:**true if a preset dictionary is needed for decompression**See Also:**[setDictionary(byte[], int, int)](http://docs.google.com/java/util/zip/Inflater.html#setDictionary(byte%5B%5D,%20int,%20int))

### finished

public boolean **finished**()

Returns true if the end of the compressed data stream has been reached.

**Returns:**true if the end of the compressed data stream has been reached

### inflate

public int **inflate**(byte[] b,  
 int off,  
 int len)  
 throws [DataFormatException](http://docs.google.com/java/util/zip/DataFormatException.html)

Uncompresses bytes into specified buffer. Returns actual number of bytes uncompressed. A return value of 0 indicates that needsInput() or needsDictionary() should be called in order to determine if more input data or a preset dictionary is required. In the latter case, getAdler() can be used to get the Adler-32 value of the dictionary required.

**Parameters:**b - the buffer for the uncompressed dataoff - the start offset of the datalen - the maximum number of uncompressed bytes **Returns:**the actual number of uncompressed bytes **Throws:** [DataFormatException](http://docs.google.com/java/util/zip/DataFormatException.html) - if the compressed data format is invalid**See Also:**[needsInput()](http://docs.google.com/java/util/zip/Inflater.html#needsInput()), [needsDictionary()](http://docs.google.com/java/util/zip/Inflater.html#needsDictionary())

### inflate

public int **inflate**(byte[] b)  
 throws [DataFormatException](http://docs.google.com/java/util/zip/DataFormatException.html)

Uncompresses bytes into specified buffer. Returns actual number of bytes uncompressed. A return value of 0 indicates that needsInput() or needsDictionary() should be called in order to determine if more input data or a preset dictionary is required. In the latter case, getAdler() can be used to get the Adler-32 value of the dictionary required.

**Parameters:**b - the buffer for the uncompressed data **Returns:**the actual number of uncompressed bytes **Throws:** [DataFormatException](http://docs.google.com/java/util/zip/DataFormatException.html) - if the compressed data format is invalid**See Also:**[needsInput()](http://docs.google.com/java/util/zip/Inflater.html#needsInput()), [needsDictionary()](http://docs.google.com/java/util/zip/Inflater.html#needsDictionary())

### getAdler

public int **getAdler**()

Returns the ADLER-32 value of the uncompressed data.

**Returns:**the ADLER-32 value of the uncompressed data

### getTotalIn

public int **getTotalIn**()

Returns the total number of compressed bytes input so far.

Since the number of bytes may be greater than Integer.MAX\_VALUE, the [getBytesRead()](http://docs.google.com/java/util/zip/Inflater.html#getBytesRead()) method is now the preferred means of obtaining this information.

**Returns:**the total number of compressed bytes input so far

### getBytesRead

public long **getBytesRead**()

Returns the total number of compressed bytes input so far.

**Returns:**the total (non-negative) number of compressed bytes input so far**Since:** 1.5

### getTotalOut

public int **getTotalOut**()

Returns the total number of uncompressed bytes output so far.

Since the number of bytes may be greater than Integer.MAX\_VALUE, the [getBytesWritten()](http://docs.google.com/java/util/zip/Inflater.html#getBytesWritten()) method is now the preferred means of obtaining this information.

**Returns:**the total number of uncompressed bytes output so far

### getBytesWritten

public long **getBytesWritten**()

Returns the total number of uncompressed bytes output so far.

**Returns:**the total (non-negative) number of uncompressed bytes output so far**Since:** 1.5

### reset

public void **reset**()

Resets inflater so that a new set of input data can be processed.

### end

public void **end**()

Closes the decompressor and discards any unprocessed input. This method should be called when the decompressor is no longer being used, but will also be called automatically by the finalize() method. Once this method is called, the behavior of the Inflater object is undefined.

### finalize

protected void **finalize**()

Closes the decompressor when garbage is collected.

**Overrides:**[finalize](http://docs.google.com/java/lang/Object.html#finalize()) in class [Object](http://docs.google.com/java/lang/Object.html)

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/Inflater.html) | [**Tree**](http://docs.google.com/package-tree.html) | [**Deprecated**](http://docs.google.com/deprecated-list.html) | [**Index**](http://docs.google.com/index-files/index-1.html) | [**Help**](http://docs.google.com/help-doc.html) | | --- | --- | --- | --- | --- | --- | --- | --- | | | ***Java™ Platform***  ***Standard Ed. 6*** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [**PREV CLASS**](http://docs.google.com/java/util/zip/GZIPOutputStream.html)   [**NEXT CLASS**](http://docs.google.com/java/util/zip/InflaterInputStream.html) | [**FRAMES**](http://docs.google.com/index.html?java/util/zip/Inflater.html)    [**NO FRAMES**](http://docs.google.com/Inflater.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | [CONSTR](#3znysh7) | [METHOD](#2et92p0) | DETAIL: FIELD | [CONSTR](#3dy6vkm) | [METHOD](#2s8eyo1) |

[Submit a bug or feature](http://bugs.sun.com/services/bugreport/index.jsp)

For further API reference and developer documentation, see [Java SE Developer Documentation](http://docs.google.com/webnotes/devdocs-vs-specs.html). That documentation contains more detailed, developer-targeted descriptions, with conceptual overviews, definitions of terms, workarounds, and working code examples.

Copyright 2006 Sun Microsystems, Inc. All rights reserved. Use is subject to [license terms](http://docs.google.com/legal/license.html). Also see the [documentation redistribution policy](http://java.sun.com/docs/redist.html).