| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/Pack200.Packer.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/jar/Pack200.html)   [**NEXT CLASS**](http://docs.google.com/java/util/jar/Pack200.Unpacker.html) | [**FRAMES**](http://docs.google.com/index.html?java/util/jar/Pack200.Packer.html)    [**NO FRAMES**](http://docs.google.com/Pack200.Packer.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | [FIELD](#3znysh7) | CONSTR | [METHOD](#2et92p0) | DETAIL: [FIELD](#tyjcwt) | CONSTR | [METHOD](#2bn6wsx) |

## **java.util.jar**

Interface Pack200.Packer

**Enclosing class:**[Pack200](http://docs.google.com/java/util/jar/Pack200.html)

public static interface **Pack200.Packer**

The packer engine applies various transformations to the input JAR file, making the pack stream highly compressible by a compressor such as gzip or zip. An instance of the engine can be obtained using [Pack200.newPacker()](http://docs.google.com/java/util/jar/Pack200.html#newPacker()). The high degree of compression is achieved by using a number of techniques described in the JSR 200 specification. Some of the techniques are sorting, re-ordering and co-location of the constant pool.

The pack engine is initialized to an initial state as described by their properties below. The initial state can be manipulated by getting the engine properties (using [properties()](http://docs.google.com/java/util/jar/Pack200.Packer.html#properties())) and storing the modified properties on the map. The resource files will be passed through with no changes at all. The class files will not contain identical bytes, since the unpacker is free to change minor class file features such as constant pool order. However, the class files will be semantically identical, as specified in the Java Virtual Machine Specification <http://java.sun.com/docs/books/vmspec/html/ClassFile.doc.html>.

By default, the packer does not change the order of JAR elements. Also, the modification time and deflation hint of each JAR element is passed unchanged. (Any other ZIP-archive information, such as extra attributes giving Unix file permissions, are lost.)

Note that packing and unpacking a JAR will in general alter the bytewise contents of classfiles in the JAR. This means that packing and unpacking will in general invalidate any digital signatures which rely on bytewise images of JAR elements. In order both to sign and to pack a JAR, you must first pack and unpack the JAR to "normalize" it, then compute signatures on the unpacked JAR elements, and finally repack the signed JAR. Both packing steps should use precisely the same options, and the segment limit may also need to be set to "-1", to prevent accidental variation of segment boundaries as class file sizes change slightly.

(Here's why this works: Any reordering the packer does of any classfile structures is idempotent, so the second packing does not change the orderings produced by the first packing. Also, the unpacker is guaranteed by the JSR 200 specification to produce a specific bytewise image for any given transmission ordering of archive elements.)

In order to maintain backward compatibility, if the input JAR-files are solely comprised of 1.5 (or lesser) classfiles, a 1.5 compatible pack file is produced. Otherwise a 1.6 compatible pack200 file is produced.

**Since:** 1.5

| **Field Summary** | |
| --- | --- |
| static [String](http://docs.google.com/java/lang/String.html) | [**CLASS\_ATTRIBUTE\_PFX**](http://docs.google.com/java/util/jar/Pack200.Packer.html#CLASS_ATTRIBUTE_PFX)            When concatenated with a class attribute name, indicates the format of that attribute, using the layout language specified in the JSR 200 specification. |
| static [String](http://docs.google.com/java/lang/String.html) | [**CODE\_ATTRIBUTE\_PFX**](http://docs.google.com/java/util/jar/Pack200.Packer.html#CODE_ATTRIBUTE_PFX)            When concatenated with a code attribute name, indicates the format of that attribute. |
| static [String](http://docs.google.com/java/lang/String.html) | [**DEFLATE\_HINT**](http://docs.google.com/java/util/jar/Pack200.Packer.html#DEFLATE_HINT)            If this property is set to [TRUE](http://docs.google.com/java/util/jar/Pack200.Packer.html#TRUE) or [FALSE](http://docs.google.com/java/util/jar/Pack200.Packer.html#FALSE), the packer will set the deflation hint accordingly in the output archive, and will not transmit the individual deflation hints of archive elements. |
| static [String](http://docs.google.com/java/lang/String.html) | [**EFFORT**](http://docs.google.com/java/util/jar/Pack200.Packer.html#EFFORT)            If this property is set to a single decimal digit, the packer will use the indicated amount of effort in compressing the archive. |
| static [String](http://docs.google.com/java/lang/String.html) | [**ERROR**](http://docs.google.com/java/util/jar/Pack200.Packer.html#ERROR)            The string "error", a possible value for certain properties. |
| static [String](http://docs.google.com/java/lang/String.html) | [**FALSE**](http://docs.google.com/java/util/jar/Pack200.Packer.html#FALSE)            The string "false", a possible value for certain properties. |
| static [String](http://docs.google.com/java/lang/String.html) | [**FIELD\_ATTRIBUTE\_PFX**](http://docs.google.com/java/util/jar/Pack200.Packer.html#FIELD_ATTRIBUTE_PFX)            When concatenated with a field attribute name, indicates the format of that attribute. |
| static [String](http://docs.google.com/java/lang/String.html) | [**KEEP**](http://docs.google.com/java/util/jar/Pack200.Packer.html#KEEP)            The string "keep", a possible value for certain properties. |
| static [String](http://docs.google.com/java/lang/String.html) | [**KEEP\_FILE\_ORDER**](http://docs.google.com/java/util/jar/Pack200.Packer.html#KEEP_FILE_ORDER)            If this property is set to [TRUE](http://docs.google.com/java/util/jar/Pack200.Packer.html#TRUE), the packer will transmit all elements in their original order within the source archive. |
| static [String](http://docs.google.com/java/lang/String.html) | [**LATEST**](http://docs.google.com/java/util/jar/Pack200.Packer.html#LATEST)            The string "latest", a possible value for certain properties. |
| static [String](http://docs.google.com/java/lang/String.html) | [**METHOD\_ATTRIBUTE\_PFX**](http://docs.google.com/java/util/jar/Pack200.Packer.html#METHOD_ATTRIBUTE_PFX)            When concatenated with a method attribute name, indicates the format of that attribute. |
| static [String](http://docs.google.com/java/lang/String.html) | [**MODIFICATION\_TIME**](http://docs.google.com/java/util/jar/Pack200.Packer.html#MODIFICATION_TIME)            If this property is set to the special string [LATEST](http://docs.google.com/java/util/jar/Pack200.Packer.html#LATEST), the packer will attempt to determine the latest modification time, among all the available entries in the original archive or the latest modification time of all the available entries in each segment. |
| static [String](http://docs.google.com/java/lang/String.html) | [**PASS**](http://docs.google.com/java/util/jar/Pack200.Packer.html#PASS)            The string "pass", a possible value for certain properties. |
| static [String](http://docs.google.com/java/lang/String.html) | [**PASS\_FILE\_PFX**](http://docs.google.com/java/util/jar/Pack200.Packer.html#PASS_FILE_PFX)            Indicates that a file should be passed through bytewise, with no compression. |
| static [String](http://docs.google.com/java/lang/String.html) | [**PROGRESS**](http://docs.google.com/java/util/jar/Pack200.Packer.html#PROGRESS)            The unpacker's progress as a percentage, as periodically updated by the unpacker. |
| static [String](http://docs.google.com/java/lang/String.html) | [**SEGMENT\_LIMIT**](http://docs.google.com/java/util/jar/Pack200.Packer.html#SEGMENT_LIMIT)            This property is a numeral giving the estimated target size N (in bytes) of each archive segment. |
| static [String](http://docs.google.com/java/lang/String.html) | [**STRIP**](http://docs.google.com/java/util/jar/Pack200.Packer.html#STRIP)            The string "strip", a possible value for certain properties. |
| static [String](http://docs.google.com/java/lang/String.html) | [**TRUE**](http://docs.google.com/java/util/jar/Pack200.Packer.html#TRUE)            The string "true", a possible value for certain properties. |
| static [String](http://docs.google.com/java/lang/String.html) | [**UNKNOWN\_ATTRIBUTE**](http://docs.google.com/java/util/jar/Pack200.Packer.html#UNKNOWN_ATTRIBUTE)            Indicates the action to take when a class-file containing an unknown attribute is encountered. |

| **Method Summary** | |
| --- | --- |
| void | [**addPropertyChangeListener**](http://docs.google.com/java/util/jar/Pack200.Packer.html#addPropertyChangeListener(java.beans.PropertyChangeListener))([PropertyChangeListener](http://docs.google.com/java/beans/PropertyChangeListener.html) listener)            Registers a listener for PropertyChange events on the properties map. |
| void | [**pack**](http://docs.google.com/java/util/jar/Pack200.Packer.html#pack(java.util.jar.JarFile,%20java.io.OutputStream))([JarFile](http://docs.google.com/java/util/jar/JarFile.html) in, [OutputStream](http://docs.google.com/java/io/OutputStream.html) out)            Takes a JarFile and converts it into a Pack200 archive. |
| void | [**pack**](http://docs.google.com/java/util/jar/Pack200.Packer.html#pack(java.util.jar.JarInputStream,%20java.io.OutputStream))([JarInputStream](http://docs.google.com/java/util/jar/JarInputStream.html) in, [OutputStream](http://docs.google.com/java/io/OutputStream.html) out)            Takes a JarInputStream and converts it into a Pack200 archive. |
| [SortedMap](http://docs.google.com/java/util/SortedMap.html)<[String](http://docs.google.com/java/lang/String.html),[String](http://docs.google.com/java/lang/String.html)> | [**properties**](http://docs.google.com/java/util/jar/Pack200.Packer.html#properties())()            Get the set of this engine's properties. |
| void | [**removePropertyChangeListener**](http://docs.google.com/java/util/jar/Pack200.Packer.html#removePropertyChangeListener(java.beans.PropertyChangeListener))([PropertyChangeListener](http://docs.google.com/java/beans/PropertyChangeListener.html) listener)            Remove a listener for PropertyChange events, added by the [addPropertyChangeListener(java.beans.PropertyChangeListener)](http://docs.google.com/java/util/jar/Pack200.Packer.html#addPropertyChangeListener(java.beans.PropertyChangeListener)). |

| **Field Detail** |
| --- |

### SEGMENT\_LIMIT

static final [String](http://docs.google.com/java/lang/String.html) **SEGMENT\_LIMIT**

This property is a numeral giving the estimated target size N (in bytes) of each archive segment. If a single input file requires more than N bytes, it will be given its own archive segment.

As a special case, a value of -1 will produce a single large segment with all input files, while a value of 0 will produce one segment for each class. Larger archive segments result in less fragmentation and better compression, but processing them requires more memory.

The size of each segment is estimated by counting the size of each input file to be transmitted in the segment, along with the size of its name and other transmitted properties.

The default is 1000000 (a million bytes). This allows input JAR files of moderate size to be transmitted in one segment. It also puts a limit on memory requirements for packers and unpackers.

A 10Mb JAR packed without this limit will typically pack about 10% smaller, but the packer may require a larger Java heap (about ten times the segment limit).

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.util.jar.Pack200.Packer.SEGMENT_LIMIT)

### KEEP\_FILE\_ORDER

static final [String](http://docs.google.com/java/lang/String.html) **KEEP\_FILE\_ORDER**

If this property is set to [TRUE](http://docs.google.com/java/util/jar/Pack200.Packer.html#TRUE), the packer will transmit all elements in their original order within the source archive.

If it is set to [FALSE](http://docs.google.com/java/util/jar/Pack200.Packer.html#FALSE), the packer may reorder elements, and also remove JAR directory entries, which carry no useful information for Java applications. (Typically this enables better compression.)

The default is [TRUE](http://docs.google.com/java/util/jar/Pack200.Packer.html#TRUE), which preserves the input information, but may cause the transmitted archive to be larger than necessary.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.util.jar.Pack200.Packer.KEEP_FILE_ORDER)

### EFFORT

static final [String](http://docs.google.com/java/lang/String.html) **EFFORT**

If this property is set to a single decimal digit, the packer will use the indicated amount of effort in compressing the archive. Level 1 may produce somewhat larger size and faster compression speed, while level 9 will take much longer but may produce better compression.

The special value 0 instructs the packer to copy through the original JAR file directly, with no compression. The JSR 200 standard requires any unpacker to understand this special case as a pass-through of the entire archive.

The default is 5, investing a modest amount of time to produce reasonable compression.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.util.jar.Pack200.Packer.EFFORT)

### DEFLATE\_HINT

static final [String](http://docs.google.com/java/lang/String.html) **DEFLATE\_HINT**

If this property is set to [TRUE](http://docs.google.com/java/util/jar/Pack200.Packer.html#TRUE) or [FALSE](http://docs.google.com/java/util/jar/Pack200.Packer.html#FALSE), the packer will set the deflation hint accordingly in the output archive, and will not transmit the individual deflation hints of archive elements.

If this property is set to the special string [KEEP](http://docs.google.com/java/util/jar/Pack200.Packer.html#KEEP), the packer will attempt to determine an independent deflation hint for each available element of the input archive, and transmit this hint separately.

The default is [KEEP](http://docs.google.com/java/util/jar/Pack200.Packer.html#KEEP), which preserves the input information, but may cause the transmitted archive to be larger than necessary.

It is up to the unpacker implementation to take action upon the hint to suitably compress the elements of the resulting unpacked jar.

The deflation hint of a ZIP or JAR element indicates whether the element was deflated or stored directly.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.util.jar.Pack200.Packer.DEFLATE_HINT)

### MODIFICATION\_TIME

static final [String](http://docs.google.com/java/lang/String.html) **MODIFICATION\_TIME**

If this property is set to the special string [LATEST](http://docs.google.com/java/util/jar/Pack200.Packer.html#LATEST), the packer will attempt to determine the latest modification time, among all the available entries in the original archive or the latest modification time of all the available entries in each segment. This single value will be transmitted as part of the segment and applied to all the entries in each segment, [SEGMENT\_LIMIT](http://docs.google.com/java/util/jar/Pack200.Packer.html#SEGMENT_LIMIT).

This can marginally decrease the transmitted size of the archive, at the expense of setting all installed files to a single date.

If this property is set to the special string [KEEP](http://docs.google.com/java/util/jar/Pack200.Packer.html#KEEP), the packer transmits a separate modification time for each input element.

The default is [KEEP](http://docs.google.com/java/util/jar/Pack200.Packer.html#KEEP), which preserves the input information, but may cause the transmitted archive to be larger than necessary.

It is up to the unpacker implementation to take action to suitably set the modification time of each element of its output file.

**See Also:**[SEGMENT\_LIMIT](http://docs.google.com/java/util/jar/Pack200.Packer.html#SEGMENT_LIMIT), [Constant Field Values](http://docs.google.com/constant-values.html#java.util.jar.Pack200.Packer.MODIFICATION_TIME)

### PASS\_FILE\_PFX

static final [String](http://docs.google.com/java/lang/String.html) **PASS\_FILE\_PFX**

Indicates that a file should be passed through bytewise, with no compression. Multiple files may be specified by specifying additional properties with distinct strings appended, to make a family of properties with the common prefix.

There is no pathname transformation, except that the system file separator is replaced by the JAR file separator '/'.

The resulting file names must match exactly as strings with their occurrences in the JAR file.

If a property value is a directory name, all files under that directory will be passed also.

Examples:

Map p = packer.properties();  
 p.put(PASS\_FILE\_PFX+0, "mutants/Rogue.class");  
 p.put(PASS\_FILE\_PFX+1, "mutants/Wolverine.class");  
 p.put(PASS\_FILE\_PFX+2, "mutants/Storm.class");  
 # Pass all files in an entire directory hierarchy:  
 p.put(PASS\_FILE\_PFX+3, "police/");

.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.util.jar.Pack200.Packer.PASS_FILE_PFX)

### UNKNOWN\_ATTRIBUTE

static final [String](http://docs.google.com/java/lang/String.html) **UNKNOWN\_ATTRIBUTE**

Indicates the action to take when a class-file containing an unknown attribute is encountered. Possible values are the strings [ERROR](http://docs.google.com/java/util/jar/Pack200.Packer.html#ERROR), [STRIP](http://docs.google.com/java/util/jar/Pack200.Packer.html#STRIP), and [PASS](http://docs.google.com/java/util/jar/Pack200.Packer.html#PASS).

The string [ERROR](http://docs.google.com/java/util/jar/Pack200.Packer.html#ERROR) means that the pack operation as a whole will fail, with an exception of type IOException. The string [STRIP](http://docs.google.com/java/util/jar/Pack200.Packer.html#STRIP) means that the attribute will be dropped. The string [PASS](http://docs.google.com/java/util/jar/Pack200.Packer.html#PASS) means that the whole class-file will be passed through (as if it were a resource file) without compression, with a suitable warning. This is the default value for this property.

Examples:

Map p = pack200.getProperties();  
 p.put(UNKNOWN\_ATTRIBUTE, ERROR);  
 p.put(UNKNOWN\_ATTRIBUTE, STRIP);  
 p.put(UNKNOWN\_ATTRIBUTE, PASS);

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.util.jar.Pack200.Packer.UNKNOWN_ATTRIBUTE)

### CLASS\_ATTRIBUTE\_PFX

static final [String](http://docs.google.com/java/lang/String.html) **CLASS\_ATTRIBUTE\_PFX**

When concatenated with a class attribute name, indicates the format of that attribute, using the layout language specified in the JSR 200 specification.

For example, the effect of this option is built in: pack.class.attribute.SourceFile=RUH.

The special strings [ERROR](http://docs.google.com/java/util/jar/Pack200.Packer.html#ERROR), [STRIP](http://docs.google.com/java/util/jar/Pack200.Packer.html#STRIP), and [PASS](http://docs.google.com/java/util/jar/Pack200.Packer.html#PASS) are also allowed, with the same meaning as [UNKNOWN\_ATTRIBUTE](http://docs.google.com/java/util/jar/Pack200.Packer.html#UNKNOWN_ATTRIBUTE). This provides a way for users to request that specific attributes be refused, stripped, or passed bitwise (with no class compression).

Code like this might be used to support attributes for JCOV:

Map p = packer.properties();  
 p.put(CODE\_ATTRIBUTE\_PFX+"CoverageTable", "NH[PHHII]");  
 p.put(CODE\_ATTRIBUTE\_PFX+"CharacterRangeTable", "NH[PHPOHIIH]");  
 p.put(CLASS\_ATTRIBUTE\_PFX+"SourceID", "RUH");  
 p.put(CLASS\_ATTRIBUTE\_PFX+"CompilationID", "RUH");

Code like this might be used to strip debugging attributes:

Map p = packer.properties();  
 p.put(CODE\_ATTRIBUTE\_PFX+"LineNumberTable", STRIP);  
 p.put(CODE\_ATTRIBUTE\_PFX+"LocalVariableTable", STRIP);  
 p.put(CLASS\_ATTRIBUTE\_PFX+"SourceFile", STRIP);

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.util.jar.Pack200.Packer.CLASS_ATTRIBUTE_PFX)

### FIELD\_ATTRIBUTE\_PFX

static final [String](http://docs.google.com/java/lang/String.html) **FIELD\_ATTRIBUTE\_PFX**

When concatenated with a field attribute name, indicates the format of that attribute. For example, the effect of this option is built in: pack.field.attribute.Deprecated=. The special strings [ERROR](http://docs.google.com/java/util/jar/Pack200.Packer.html#ERROR), [STRIP](http://docs.google.com/java/util/jar/Pack200.Packer.html#STRIP), and [PASS](http://docs.google.com/java/util/jar/Pack200.Packer.html#PASS) are also allowed.

**See Also:**[CLASS\_ATTRIBUTE\_PFX](http://docs.google.com/java/util/jar/Pack200.Packer.html#CLASS_ATTRIBUTE_PFX), [Constant Field Values](http://docs.google.com/constant-values.html#java.util.jar.Pack200.Packer.FIELD_ATTRIBUTE_PFX)

### METHOD\_ATTRIBUTE\_PFX

static final [String](http://docs.google.com/java/lang/String.html) **METHOD\_ATTRIBUTE\_PFX**

When concatenated with a method attribute name, indicates the format of that attribute. For example, the effect of this option is built in: pack.method.attribute.Exceptions=NH[RCH]. The special strings [ERROR](http://docs.google.com/java/util/jar/Pack200.Packer.html#ERROR), [STRIP](http://docs.google.com/java/util/jar/Pack200.Packer.html#STRIP), and [PASS](http://docs.google.com/java/util/jar/Pack200.Packer.html#PASS) are also allowed.

**See Also:**[CLASS\_ATTRIBUTE\_PFX](http://docs.google.com/java/util/jar/Pack200.Packer.html#CLASS_ATTRIBUTE_PFX), [Constant Field Values](http://docs.google.com/constant-values.html#java.util.jar.Pack200.Packer.METHOD_ATTRIBUTE_PFX)

### CODE\_ATTRIBUTE\_PFX

static final [String](http://docs.google.com/java/lang/String.html) **CODE\_ATTRIBUTE\_PFX**

When concatenated with a code attribute name, indicates the format of that attribute. For example, the effect of this option is built in: pack.code.attribute.LocalVariableTable=NH[PHOHRUHRSHH]. The special strings [ERROR](http://docs.google.com/java/util/jar/Pack200.Packer.html#ERROR), [STRIP](http://docs.google.com/java/util/jar/Pack200.Packer.html#STRIP), and [PASS](http://docs.google.com/java/util/jar/Pack200.Packer.html#PASS) are also allowed.

**See Also:**[CLASS\_ATTRIBUTE\_PFX](http://docs.google.com/java/util/jar/Pack200.Packer.html#CLASS_ATTRIBUTE_PFX), [Constant Field Values](http://docs.google.com/constant-values.html#java.util.jar.Pack200.Packer.CODE_ATTRIBUTE_PFX)

### PROGRESS

static final [String](http://docs.google.com/java/lang/String.html) **PROGRESS**

The unpacker's progress as a percentage, as periodically updated by the unpacker. Values of 0 - 100 are normal, and -1 indicates a stall. Observe this property with a [PropertyChangeListener](http://docs.google.com/java/beans/PropertyChangeListener.html).

At a minimum, the unpacker must set progress to 0 at the beginning of a packing operation, and to 100 at the end.

**See Also:**[addPropertyChangeListener(java.beans.PropertyChangeListener)](http://docs.google.com/java/util/jar/Pack200.Packer.html#addPropertyChangeListener(java.beans.PropertyChangeListener)), [Constant Field Values](http://docs.google.com/constant-values.html#java.util.jar.Pack200.Packer.PROGRESS)

### KEEP

static final [String](http://docs.google.com/java/lang/String.html) **KEEP**

The string "keep", a possible value for certain properties.

**See Also:**[DEFLATE\_HINT](http://docs.google.com/java/util/jar/Pack200.Packer.html#DEFLATE_HINT), [MODIFICATION\_TIME](http://docs.google.com/java/util/jar/Pack200.Packer.html#MODIFICATION_TIME), [Constant Field Values](http://docs.google.com/constant-values.html#java.util.jar.Pack200.Packer.KEEP)

### PASS

static final [String](http://docs.google.com/java/lang/String.html) **PASS**

The string "pass", a possible value for certain properties.

**See Also:**[UNKNOWN\_ATTRIBUTE](http://docs.google.com/java/util/jar/Pack200.Packer.html#UNKNOWN_ATTRIBUTE), [CLASS\_ATTRIBUTE\_PFX](http://docs.google.com/java/util/jar/Pack200.Packer.html#CLASS_ATTRIBUTE_PFX), [FIELD\_ATTRIBUTE\_PFX](http://docs.google.com/java/util/jar/Pack200.Packer.html#FIELD_ATTRIBUTE_PFX), [METHOD\_ATTRIBUTE\_PFX](http://docs.google.com/java/util/jar/Pack200.Packer.html#METHOD_ATTRIBUTE_PFX), [CODE\_ATTRIBUTE\_PFX](http://docs.google.com/java/util/jar/Pack200.Packer.html#CODE_ATTRIBUTE_PFX), [Constant Field Values](http://docs.google.com/constant-values.html#java.util.jar.Pack200.Packer.PASS)

### STRIP

static final [String](http://docs.google.com/java/lang/String.html) **STRIP**

The string "strip", a possible value for certain properties.

**See Also:**[UNKNOWN\_ATTRIBUTE](http://docs.google.com/java/util/jar/Pack200.Packer.html#UNKNOWN_ATTRIBUTE), [CLASS\_ATTRIBUTE\_PFX](http://docs.google.com/java/util/jar/Pack200.Packer.html#CLASS_ATTRIBUTE_PFX), [FIELD\_ATTRIBUTE\_PFX](http://docs.google.com/java/util/jar/Pack200.Packer.html#FIELD_ATTRIBUTE_PFX), [METHOD\_ATTRIBUTE\_PFX](http://docs.google.com/java/util/jar/Pack200.Packer.html#METHOD_ATTRIBUTE_PFX), [CODE\_ATTRIBUTE\_PFX](http://docs.google.com/java/util/jar/Pack200.Packer.html#CODE_ATTRIBUTE_PFX), [Constant Field Values](http://docs.google.com/constant-values.html#java.util.jar.Pack200.Packer.STRIP)

### ERROR

static final [String](http://docs.google.com/java/lang/String.html) **ERROR**

The string "error", a possible value for certain properties.

**See Also:**[UNKNOWN\_ATTRIBUTE](http://docs.google.com/java/util/jar/Pack200.Packer.html#UNKNOWN_ATTRIBUTE), [CLASS\_ATTRIBUTE\_PFX](http://docs.google.com/java/util/jar/Pack200.Packer.html#CLASS_ATTRIBUTE_PFX), [FIELD\_ATTRIBUTE\_PFX](http://docs.google.com/java/util/jar/Pack200.Packer.html#FIELD_ATTRIBUTE_PFX), [METHOD\_ATTRIBUTE\_PFX](http://docs.google.com/java/util/jar/Pack200.Packer.html#METHOD_ATTRIBUTE_PFX), [CODE\_ATTRIBUTE\_PFX](http://docs.google.com/java/util/jar/Pack200.Packer.html#CODE_ATTRIBUTE_PFX), [Constant Field Values](http://docs.google.com/constant-values.html#java.util.jar.Pack200.Packer.ERROR)

### TRUE

static final [String](http://docs.google.com/java/lang/String.html) **TRUE**

The string "true", a possible value for certain properties.

**See Also:**[KEEP\_FILE\_ORDER](http://docs.google.com/java/util/jar/Pack200.Packer.html#KEEP_FILE_ORDER), [DEFLATE\_HINT](http://docs.google.com/java/util/jar/Pack200.Packer.html#DEFLATE_HINT), [Constant Field Values](http://docs.google.com/constant-values.html#java.util.jar.Pack200.Packer.TRUE)

### FALSE

static final [String](http://docs.google.com/java/lang/String.html) **FALSE**

The string "false", a possible value for certain properties.

**See Also:**[KEEP\_FILE\_ORDER](http://docs.google.com/java/util/jar/Pack200.Packer.html#KEEP_FILE_ORDER), [DEFLATE\_HINT](http://docs.google.com/java/util/jar/Pack200.Packer.html#DEFLATE_HINT), [Constant Field Values](http://docs.google.com/constant-values.html#java.util.jar.Pack200.Packer.FALSE)

### LATEST

static final [String](http://docs.google.com/java/lang/String.html) **LATEST**

The string "latest", a possible value for certain properties.

**See Also:**[MODIFICATION\_TIME](http://docs.google.com/java/util/jar/Pack200.Packer.html#MODIFICATION_TIME), [Constant Field Values](http://docs.google.com/constant-values.html#java.util.jar.Pack200.Packer.LATEST)

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

### properties

[SortedMap](http://docs.google.com/java/util/SortedMap.html)<[String](http://docs.google.com/java/lang/String.html),[String](http://docs.google.com/java/lang/String.html)> **properties**()

Get the set of this engine's properties. This set is a "live view", so that changing its contents immediately affects the Packer engine, and changes from the engine (such as progress indications) are immediately visible in the map.

The property map may contain pre-defined implementation specific and default properties. Users are encouraged to read the information and fully understand the implications, before modifying pre-existing properties.

Implementation specific properties are prefixed with a package name associated with the implementor, beginning with com. or a similar prefix. All property names beginning with pack. and unpack. are reserved for use by this API.

Unknown properties may be ignored or rejected with an unspecified error, and invalid entries may cause an unspecified error to be thrown.

The returned map implements all optional [SortedMap](http://docs.google.com/java/util/SortedMap.html) operations

**Returns:**A sorted association of property key strings to property values.

### pack

void **pack**([JarFile](http://docs.google.com/java/util/jar/JarFile.html) in,  
 [OutputStream](http://docs.google.com/java/io/OutputStream.html) out)  
 throws [IOException](http://docs.google.com/java/io/IOException.html)

Takes a JarFile and converts it into a Pack200 archive.

Closes its input but not its output. (Pack200 archives are appendable.)

**Parameters:**in - a JarFileout - an OutputStream **Throws:** [IOException](http://docs.google.com/java/io/IOException.html) - if an error is encountered.

### pack

void **pack**([JarInputStream](http://docs.google.com/java/util/jar/JarInputStream.html) in,  
 [OutputStream](http://docs.google.com/java/io/OutputStream.html) out)  
 throws [IOException](http://docs.google.com/java/io/IOException.html)

Takes a JarInputStream and converts it into a Pack200 archive.

Closes its input but not its output. (Pack200 archives are appendable.)

The modification time and deflation hint attributes are not available, for the JAR manifest file and its containing directory.

**Parameters:**in - a JarInputStreamout - an OutputStream **Throws:** [IOException](http://docs.google.com/java/io/IOException.html) - if an error is encountered.**See Also:**[MODIFICATION\_TIME](http://docs.google.com/java/util/jar/Pack200.Packer.html#MODIFICATION_TIME), [DEFLATE\_HINT](http://docs.google.com/java/util/jar/Pack200.Packer.html#DEFLATE_HINT)

### addPropertyChangeListener

void **addPropertyChangeListener**([PropertyChangeListener](http://docs.google.com/java/beans/PropertyChangeListener.html) listener)

Registers a listener for PropertyChange events on the properties map. This is typically used by applications to update a progress bar.

**Parameters:**listener - An object to be invoked when a property is changed.**See Also:**[properties()](http://docs.google.com/java/util/jar/Pack200.Packer.html#properties()), [PROGRESS](http://docs.google.com/java/util/jar/Pack200.Packer.html#PROGRESS)

### removePropertyChangeListener

void **removePropertyChangeListener**([PropertyChangeListener](http://docs.google.com/java/beans/PropertyChangeListener.html) listener)

Remove a listener for PropertyChange events, added by the [addPropertyChangeListener(java.beans.PropertyChangeListener)](http://docs.google.com/java/util/jar/Pack200.Packer.html#addPropertyChangeListener(java.beans.PropertyChangeListener)).

**Parameters:**listener - The PropertyChange listener to be removed.**See Also:**[addPropertyChangeListener(java.beans.PropertyChangeListener)](http://docs.google.com/java/util/jar/Pack200.Packer.html#addPropertyChangeListener(java.beans.PropertyChangeListener))

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/Pack200.Packer.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/jar/Pack200.html)   [**NEXT CLASS**](http://docs.google.com/java/util/jar/Pack200.Unpacker.html) | [**FRAMES**](http://docs.google.com/index.html?java/util/jar/Pack200.Packer.html)    [**NO FRAMES**](http://docs.google.com/Pack200.Packer.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | [FIELD](#3znysh7) | CONSTR | [METHOD](#2et92p0) | DETAIL: [FIELD](#tyjcwt) | CONSTR | [METHOD](#2bn6wsx) |

[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).