Rocky Enterprise Linux 9.2 Manual Pages on command 'pack200.1'

$ man pack200.1

pack200(1) Java Deployment Tools pack200(1)

NAME

pack200 - Packages a JAR file into a compressed pack200 file for web deployment.

SYNOPSIS

pack200 [options] output-file JAR-file

Options can be in any order. The last option on the command line or in a properties file supersedes all previously specified options.

options

The command-line options. See Options.

output-file

Name of the output file.

JAR-file

Name of the input file.

DESCRIPTION

The pack200 command is a Java application that transforms a JAR file into a compressed pack200 file with the Java gzip compressor. The pack200 files are highly compressed files that can be directly deployed

to save bandwidth and reduce download time.

The pack200 command has several options to fine-tune and set the compression engine. The typical usage is shown in the following example, where myarchive.pack.gz is produced with the default pack200 command settings:

pack200 myarchive.pack.gz myarchive.jar

OPTIONS

-r, --repack

Produces a JAR file by packing and unpacking a JAR file. The resulting file can be used as an input to the jarsigner(1) tool.

The following example packs and unpacks the myarchive.jar file:

pack200 --repack myarchive-packer.jar myarchive.jar

pack200 --repack myarchive.jar

The following example preserves the order of files in the input file.

-g, --no-gzip

Produces a pack200 file. With this option, a suitable compressor must be used, and the target system must use a corresponding decompresser.

pack200 --no-gzip myarchive.pack myarchive.jar

-G, --strip-debug

Strips debugging attributes from the output. These include SourceFile, LineNumberTable, LocalVariableTable and LocalVariableTypeTable. Removing these attributes reduces the size of both downloads and installations, but reduces the usefulness of debuggers.

--keep-file-order

Preserve the order of files in the input file. This is the default behavior.

-O, --no-keep-file-order

The packer reorders and transmits all elements. The packer can also remove JAR directory names to reduce the download size. However, certain JAR file optimizations, such as indexing, might

not work correctly.

-Svalue , --segment-limit=value

The value is the estimated target size N (in bytes) of each archive segment. If a single input file requires more than N bytes, then its own archive segment is provided. As a special case, a value of -1 produces a single large segment with all input files, while a value of 0 produces 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 with the size of its name and other transmitted properties.

The default is -1, which means that the packer creates a single segment output file. In cases where extremely large output files are generated, users are strongly encouraged to use segmenting or break up the input file into smaller JARs.

A 10 MB JAR packed without this limit typically packs about 10 percent smaller, but the packer might require a larger Java heap (about 10 times the segment limit).

-Evalue , --effort=value

If the value is set to a single decimal digit, then the packer uses the indicated amount of effort in compressing the archive. Level 1 might produce somewhat larger size and faster compression speed, while level 9 takes much longer, but can produce better compression. The special value 0 instructs the pack200 command 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, to invest a modest amount of time to produce reasonable compression.

-Hvalue , --deflate-hint=value

Overrides the default, which preserves the input information,

but can cause the transmitted archive to be larger. The possible values are: true, false, or keep.

If the value is true or false, then the packer200 command sets the deflation hint accordingly in the output archive and does not transmit the individual deflation hints of archive elements.

The keep value preserves deflation hints observed in the input JAR. This is the default.

-mvalue , --modification-time=value

The possible values are latest and keep.

If the value is latest, then the packer attempts 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 that segment. This single value is transmitted as part of the segment and applied to all the entries in each segment. This can marginally decrease the transmitted size of the archive at the expense of setting all installed files to a single date.

If the value is keep, then modification times observed in the input JAR are preserved. This is the default.

-Pfile , --pass-file=file

Indicates that a file should be passed through bytewise with no compression. By repeating the option, multiple files can be specified. There is no pathname transformation, except that the system file separator is replaced by the JAR file separator forward slash (/). The resulting file names must match exactly as strings with their occurrences in the JAR file. If file is a directory name, then all files under that directory are passed.

-Uaction , --unknown-attribute=action

Overrides the default behavior, which means that the class file that contains the unknown attribute is passed through with the specified action. The possible values for actions are error, strip, or pass.

If the value is error, then the entire pack200 command operation

fails with a suitable explanation.

If the value is strip, then the attribute is dropped. Removing the required Java Virtual Machine (JVM) attributes can cause class loader failures.

If the value is pass, then the entire class is transmitted as though it is a resource.

-Cattribute-name=layout , --class-attribute=attribute-name=action

See next option.

-Fattribute-name=layout , --field-attribute=attribute-name=action

See next option.

-Mattribute-name=layout , --method-attribute=attribute-name=action

See next option.

-Dattribute-name=layout , --code-attribute=attribute-name=action

With the previous four options, the attribute layout can be specified for a class entity, such as class-attribute, field-attribute, method-attribute, and code-attribute. The attribute-name is the name of the attribute for which the layout or action is being defined. The possible values for action are some-layout-string, error, strip, pass.

some-layout-string: The layout language is defined in the JSR 200 specification, for example: --class-attribute=SourceFile=RUH.

If the value is error, then the pack200 operation fails with an explanation.

If the value is strip, then the attribute is removed from the output. Removing JVM-required attributes can cause class loader failures. For example, --class-attribute=CompilationID=pass causes the class file that contains this attribute to be passed through without further action by the packer.

If the value is pass, then the entire class is transmitted as though it is a resource.

-f pack.properties , --config-file=pack.properties

A configuration file, containing Java properties to initialize

the packer, can be specified on the command line.

pack200 -f pack.properties myarchive.pack.gz myarchive.jar

more pack.properties

Generic properties for the packer.

modification.time=latest

deflate.hint=false

keep.file.order=false

This option will cause the files bearing new attributes to

be reported as an error rather than passed uncompressed.

unknown.attribute=error

Change the segment limit to be unlimited.

segment.limit=-1

-v, --verbose

Outputs minimal messages. Multiple specification of this option will create more verbose messages.

-q, --quiet

Specifies quiet operation with no messages.

-lfilename , --log-file=filename

Specifies a log file to output messages.

-?, -h, --help

Prints help information about this command.

-V, --version

Prints version information about this command.

-Joption

Passes the specified option to the Java Virtual Machine. For more information, see the reference page for the java(1) command. For example, -J-Xms48m sets the startup memory to 48 MB.

EXIT STATUS

The following exit values are returned: 0 for successful completion and a number greater than 0 when an error occurs.

NOTES

This command should not be confused with pack(1). The pack and pack200

commands are separate products.

The Java SE API Specification provided with the JDK is the superseding authority, when there are discrepancies.

SEE ALSO

? unpack200(1)

? jar(1)

? jarsigner(1)

JDK 8 21 November 2013 pack200(1)