# Assembly

Maven Assembly Plugin relies on the provided assembly descriptors to dictate its execution. Although there are already prefabricated descriptors available for use, they can only suffice some of the common assembly requirements.

So in order for you to customize the way the Assembly Plugin creates your assemblies, you need to know how to use the Assembly Descriptor.

This descriptor specifies the type of assembly archive to create, the contents of the assembly, and the ways in which dependencies or its modules are bundled with an assembly.

```
<assembly xmlns="http://maven.apache.org/ASSEMBLY/2.1.1" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/ASSEMBLY/2.1.1 https://maven.apache.org/xsd/assembly-2.1.1.xsd (https://maven.apache.org/xsd/assembly-2.1.1.xsd)  ">
  <id/>
  <formats/>
  <includeBaseDirectory/>
  <baseDirectory/>
  <includeSiteDirectory/>
  <containerDescriptorHandlers>
    <containerDescriptorHandler>
      <handlerName/>
      <configuration/>
    </containerDescriptorHandler>
  </containerDescriptorHandlers>
  <moduleSets>
    <moduleSet>
      <useAllReactorProjects/>
      <includeSubModules/>
      <includes/>
      <excludes/>
      <sources>
        <useDefaultExcludes/>
        <outputDirectory/>
        <includes/>
        <excludes/>
        <fileMode/>
        <directoryMode/>
        <fileSets>
          <fileSet>
            <useDefaultExcludes/>
            <outputDirectory/>
            <includes/>
            <excludes/>
            <fileMode/>
            <directoryMode/>
            <directory/>
            <lineEnding/>
            <filtered/>
            <nonFilteredFileExtensions/>
          </fileSet>
        </fileSets>
        <includeModuleDirectory/>
        <excludeSubModuleDirectories/>
        <outputDirectoryMapping/>
      </sources>
      <binaries>
        <outputDirectory/>
        <includes/>
        <excludes/>
        <fileMode/>
        <directoryMode/>
        <attachmentClassifier/>
        <includeDependencies/>
        <dependencySets>
          <dependencySet>
            <outputDirectory/>
            <includes/>
            <excludes/>
            <fileMode/>
            <directoryMode/>
            <useStrictFiltering/>
            <outputFileNameMapping/>
            <unpack/>
            <unpackOptions>
              <includes/>
              <excludes/>
              <filtered/>
              <nonFilteredFileExtensions/>
              <lineEnding/>
              <useDefaultExcludes/>
              <encoding/>
            </unpackOptions>
```

```
            <scope/>
            <useProjectArtifact/>
            <useProjectAttachments/>
            <useTransitiveDependencies/>
            <useTransitiveFiltering/>
          </dependencySet>
        </dependencySets>
        <unpack/>
        <unpackOptions>
          <includes/>
          <excludes/>
          <filtered/>
          <nonFilteredFileExtensions/>
          <lineEnding/>
          <useDefaultExcludes/>
          <encoding/>
        </unpackOptions>
        <outputFileNameMapping/>
      </binaries>
    </moduleSet>
  </moduleSets>
  <fileSets>
    <fileSet>
      <useDefaultExcludes/>
      <outputDirectory/>
      <includes/>
      <excludes/>
      <fileMode/>
      <directoryMode/>
      <directory/>
      <lineEnding/>
      <filtered/>
      <nonFilteredFileExtensions/>
    </fileSet>
  </fileSets>
  <files>
    <file>
      <source/>
      <sources/>
      <outputDirectory/>
      <destName/>
      <fileMode/>
      <lineEnding/>
      <filtered/>
    </file>
  </files>
  <dependencySets>
    <dependencySet>
      <outputDirectory/>
      <includes/>
      <excludes/>
      <fileMode/>
      <directoryMode/>
      <useStrictFiltering/>
      <outputFileNameMapping/>
      <unpack/>
      <unpackOptions>
        <includes/>
        <excludes/>
        <filtered/>
        <nonFilteredFileExtensions/>
        <lineEnding/>
        <useDefaultExcludes/>
        <encoding/>
      </unpackOptions>
      <scope/>
      <useProjectArtifact/>
      <useProjectAttachments/>
      <useTransitiveDependencies/>
      <useTransitiveFiltering/>
    </dependencySet>
  </dependencySets>
```

```
  <repositories>
    <repository>
      <outputDirectory/>
      <includes/>
      <excludes/>
      <fileMode/>
      <directoryMode/>
      <includeMetadata/>
      <groupVersionAlignments>
        <groupVersionAlignment>
          <id/>
          <version/>
          <excludes/>
        </groupVersionAlignment>
      </groupVersionAlignments>
      <scope/>
    </repository>
  </repositories>
  <componentDescriptors/>
</assembly>
```

# assembly

An assembly defines a collection of files usually distributed in an archive format such as zip, tar, or tar.gz that is generated from a project. For example, a project could produce a ZIP assembly which contains a project's JAR artifact in the root directory, the runtime dependencies in a lib/ directory, and a shell script to launch a stand-alone application.

| Element | Type | Description |
|---|---|---|
| `id` | `String` | Sets the id of this assembly. This is a symbolic name for a particular assembly of files from this project. Also, aside from being used to distinctly name the assembled package by attaching its value to the generated archive, the id is used as your artifact's classifier when deploying. |
| `formats/format*` | `List<String>` | **(Many)** Specifies the formats of the assembly. It is often better to specify the formats via the goal parameter rather than here. For example, that allows different profiles to generate different types of archives. Multiple formats can be supplied and the Assembly Plugin will generate an archive for each of the desired formats. When deploying your project, all file formats specified will also be deployed. A format is specified by supplying one of the following values in a <format> subelement:<br>• **"zip"** - Creates a ZIP file format<br>• **"tar"** - Creates a TAR format<br>• **"tar.gz"** or **"tgz"** - Creates a gzip'd TAR format<br>• **"tar.bz2"** or **"tbz2"** - Creates a bzip'd TAR format<br>• **"tar.snappy"** - Creates a snappy'd TAR format<br>• **"tar.xz"** or **"txz"** - Creates a xz'd TAR format<br>• **"jar"** - Creates a JAR format<br>• **"dir"** - Creates an exploded directory format<br>• **"war"** - Creates a WAR format |

| | | |
|---|---|---|
| `includeBaseDirectory` | `boolean` | Includes a base directory in the final archive. For example, if you are creating an assembly named "your-app", setting includeBaseDirectory to true will create an archive that includes this base directory. If this option is set to false the archive created will unzip its content to the current directory.<br>**Default value is**: `true`<br><br>. |
| `baseDirectory` | `String` | Sets the base directory of the resulting assembly archive. If this is not set and includeBaseDirectory == true, ${project.build.finalName} will be used instead. (Since 2.2-beta-1) |
| `includeSiteDirectory` | `boolean` | Includes a site directory in the final archive. The site directory location of a project is determined by the siteDirectory parameter of the Assembly Plugin.<br>**Default value is**: `false`<br><br>. |
| `containerDescriptorHandlers/containerDescriptorHandler*` | `List<ContainerDescriptorHandlerConfig>` | **(Many)** Set of components which filter various container descriptors out of the normal archive stream, so they can be aggregated then added. |
| `moduleSets/moduleSet*` | `List<ModuleSet>` | **(Many)** Specifies which module files to include in the assembly. A moduleSet is specified by providing one or more of <moduleSet> subelements. |
| `fileSets/fileSet*` | `List<FileSet>` | **(Many)** Specifies which groups of files to include in the assembly. A fileSet is specified by providing one or more of <fileSet> subelements. |
| `files/file*` | `List<FileItem>` | **(Many)** Specifies which single files to include in the assembly. A file is specified by providing one or more of <file> subelements. |
| `dependencySets/dependencySet*` | `List<DependencySet>` | **(Many)** Specifies which dependencies to include in the assembly. A dependencySet is specified by providing one or more of <dependencySet> subelements. |
| `repositories/repository*` | `List<Repository>` | **(Many) Deprecated since model version 2.1.1.** Specifies which repository files to include in the assembly. A repository is specified by providing one or more of <repository> subelements. |
| `componentDescriptors/componentDescriptor*` | `List<String>` | **(Many)** Specifies the shared components xml file locations to include in the assembly. The locations specified must be relative to the base location of the descriptor. If the descriptor was found via a <descriptorRef/> element in the classpath, any components it specifies will also be found on the classpath. If it is found by pathname via a <descriptor/> element the value here will be interpreted as a path relative to the project basedir. When multiple componentDescriptors are found, their contents are merged. Check out the descriptor components (assembly-component.html) for more information. A componentDescriptor is specified by providing one or more of <componentDescriptor> subelements. |

## containerDescriptorHandler

Configures a filter for files headed into the assembly archive, to enable aggregation of various types of descriptor fragments, such as components.xml, web.xml, etc.

| Element | Type | Description |
|---|---|---|
| `handlerName` | `String` | The handler's plexus role-hint, for lookup from the container. |
| `configuration` | `DOM` | Configuration options for the handler. |

## moduleSet

A moduleSet represent one or more project <module> present inside a project's pom.xml. This allows you to include sources or binaries belonging to a project's <modules>.

**NOTE:** When using <moduleSets> from the command-line, it is required to pass first the package phase by doing: "mvn package assembly:assembly". This bug/issue is scheduled to be addressed by Maven 2.1.

| Element | Type | Description |
|---|---|---|
| `useAllReactorProjects` | `boolean` | If set to true, the plugin will include all projects in the current reactor for processing in this ModuleSet. These will be subject to include/exclude rules. (Since 2.2)<br>**Default value is**: `false`<br><br>. |
| `includeSubModules` | `boolean` | If set to false, the plugin will exclude sub-modules from processing in this ModuleSet. Otherwise, it will process all sub-modules, each subject to include/exclude rules. (Since 2.2-beta-1)<br>**Default value is**: `true`<br><br>. |
| `includes/include*` | `List<String>` | **(Many)** When <include> subelements are present, they define a set of project coordinates to include. If none is present, then <includes> represents all valid values. Artifact coordinates may be given in simple groupId:artifactId form, or they may be fully qualified in the form groupId:artifactId:type[:classifier]:version. Additionally, wildcards can be used, as in *:maven-* |
| `excludes/exclude*` | `List<String>` | **(Many)** When <exclude> subelements are present, they define a set of project artifact coordinates to exclude. If none is present, then <excludes> represents no exclusions. Artifact coordinates may be given in simple groupId:artifactId form, or they may be fully qualified in the form groupId:artifactId:type[:classifier]:version. Additionally, wildcards can be used, as in *:maven-* |
| `sources` | `ModuleSources` | When this is present, the plugin will include the source files of the included modules from this set in the resulting assembly. |
| `binaries` | `ModuleBinaries` | When this is present, the plugin will include the binaries of the included modules from this set in the resulting assembly. |

## sources

Contains configuration options for including the source files of a project module in an assembly.

| Element | Type | Description |
|---|---|---|
| `useDefaultExcludes` | `boolean` | Whether standard exclusion patterns, such as those matching CVS and Subversion metadata files, should be used when calculating the files affected by this set. For backward compatibility, the default value is true. (Since 2.2-beta-1)<br>**Default value is**: `true`<br><br>. |
| `outputDirectory` | `String` | Sets the output directory relative to the root of the root directory of the assembly. For example, "log" will put the specified files in the log directory. |
| `includes/include*` | `List<String>` | **(Many)** When <include> subelements are present, they define a set of files and directory to include. If none is present, then <includes> represents all valid values. |
| `excludes/exclude*` | `List<String>` | **(Many)** When <exclude> subelements are present, they define a set of files and directory to exclude. If none is present, then <excludes> represents no exclusions. |

| | | |
|---|---|---|
| `fileMode` | `String` | Similar to a UNIX permission, sets the file mode of the files included. THIS IS AN OCTAL VALUE. Format: (User)(Group)(Other) where each component is a sum of Read = 4, Write = 2, and Execute = 1. For example, the value 0644 translates to User read-write, Group and Other read-only. The default value is 0644 (more on unix-style permissions) (http://www.onlamp.com/pub/a/bsd/2000/09/06/FreeBSD_Basics.html) |
| `directoryMode` | `String` | Similar to a UNIX permission, sets the directory mode of the directories included. THIS IS AN OCTAL VALUE. Format: (User)(Group)(Other) where each component is a sum of Read = 4, Write = 2, and Execute = 1. For example, the value 0755 translates to User read-write, Group and Other read-only. The default value is 0755. (more on unix-style permissions) (http://www.onlamp.com/pub/a/bsd/2000/09/06/FreeBSD_Basics.html) |
| `fileSets/fileSet*` | `List<FileSet>` | **(Many)** Specifies which groups of files from each included module to include in the assembly. A fileSet is specified by providing one or more of <fileSet> subelements. (Since 2.2-beta-1) |
| `includeModuleDirectory` | `boolean` | Specifies whether the module's finalName should be prepended to the outputDirectory values of any fileSets applied to it. (Since 2.2-beta-1)<br>**Default value is**: `true`<br>. |
| `excludeSubModuleDirectories` | `boolean` | Specifies whether sub-module directories below the current module should be excluded from fileSets applied to that module. This might be useful if you only mean to copy the sources for the exact module list matched by this ModuleSet, ignoring (or processing separately) the modules which exist in directories below the current one. (Since 2.2-beta-1)<br>**Default value is**: `true`<br>. |
| `outputDirectoryMapping` | `String` | Sets the mapping pattern for all module base-directories included in this assembly. NOTE: This field is only used if includeModuleDirectory == true. Default is the module's ${artifactId} in 2.2-beta-1, and ${module.artifactId} in subsequent versions. (Since 2.2-beta-1)<br>**Default value is**: `${module.artifactId}`<br>. |

# fileSet

A fileSet allows the inclusion of groups of files into the assembly.

| Element | Type | Description |
|---|---|---|
| `useDefaultExcludes` | `boolean` | Whether standard exclusion patterns, such as those matching CVS and Subversion metadata files, should be used when calculating the files affected by this set. For backward compatibility, the default value is true. (Since 2.2-beta-1)<br>**Default value is**: `true`<br>. |
| `outputDirectory` | `String` | Sets the output directory relative to the root of the root directory of the assembly. For example, "log" will put the specified files in the log directory. |
| `includes/include*` | `List<String>` | **(Many)** When <include> subelements are present, they define a set of files and directory to include. If none is present, then <includes> represents all valid values. |
| `excludes/exclude*` | `List<String>` | **(Many)** When <exclude> subelements are present, they define a set of files and directory to exclude. If none is present, then <excludes> represents no exclusions. |
| `fileMode` | `String` | Similar to a UNIX permission, sets the file mode of the files included. THIS IS AN OCTAL VALUE. Format: (User)(Group)(Other) where each component is a sum of Read = 4, Write = 2, and Execute = 1. For example, the value 0644 translates to User read-write, Group and Other read-only. The default value is 0644. (more on unix-style permissions) (http://www.onlamp.com/pub/a/bsd/2000/09/06/FreeBSD_Basics.html) |
| `directoryMode` | `String` | Similar to a UNIX permission, sets the directory mode of the directories included. THIS IS AN OCTAL VALUE. Format: (User)(Group)(Other) where each component is a sum of Read = 4, Write = 2, and Execute = 1. For example, the value 0755 translates to User read-write, Group and Other read-only. The default value is 0755. (more on unix-style permissions) (http://www.onlamp.com/pub/a/bsd/2000/09/06/FreeBSD_Basics.html) |

| | | |
|---|---|---|
| `directory` | `String` | Sets the absolute or relative location from the module's directory. For example, "src/main/bin" would select this subdirectory of the project in which this dependency is defined. |
| `lineEnding` | `String` | Sets the line-endings of the files in this fileSet. Valid values:<br>• **"keep"** - Preserve all line endings<br>• **"unix"** - Use Unix-style line endings (i.e. "\n")<br>• **"lf"** - Use a single line-feed line endings (i.e. "\n")<br>• **"dos"** - Use DOS-/Windows-style line endings (i.e. "\r\n")<br>• **"windows"** - Use DOS-/Windows-style line endings (i.e. "\r\n")<br>• **"crlf"** - Use carriage-return, line-feed line endings (i.e. "\r\n") |
| `filtered` | `boolean` | Whether to filter symbols in the files as they are copied, using properties from the build configuration. (Since 2.2-beta-1)<br>**Default value is**: `false`<br>. |
| `nonFilteredFileExtensions/nonFilteredFileExtension*` | `List<String>` | **(Many)** Additional file extensions to not apply filtering (Since 3.2.0) |

# binaries

Contains configuration options for including the binary files of a project module in an assembly.

| Element | Type | Description |
|---|---|---|
| `outputDirectory` | `String` | Sets the output directory relative to the root of the root directory of the assembly. For example, "log" will put the specified files in the log directory, directly beneath the root of the archive. |
| `includes/include*` | `List<String>` | **(Many)** When <include> subelements are present, they define a set of artifact coordinates to include. If none is present, then <includes> represents all valid values. Artifact coordinates may be given in simple groupId:artifactId form, or they may be fully qualified in the form groupId:artifactId:type[:classifier]:version. Additionally, wildcards can be used, as in *:maven-* |
| `excludes/exclude*` | `List<String>` | **(Many)** When <exclude> subelements are present, they define a set of dependency artifact coordinates to exclude. If none is present, then <excludes> represents no exclusions. Artifact coordinates may be given in simple groupId:artifactId form, or they may be fully qualified in the form groupId:artifactId:type[:classifier]:version. Additionally, wildcards can be used, as in *:maven-* |
| `fileMode` | `String` | Similar to a UNIX permission, sets the file mode of the files included. THIS IS AN OCTAL VALUE. Format: (User)(Group)(Other) where each component is a sum of Read = 4, Write = 2, and Execute = 1. For example, the value 0644 translates to User read-write, Group and Other read-only. The default value is 0644 (more on unix-style permissions) (http://www.onlamp.com/pub/a/bsd/2000/09/06/FreeBSD_Basics.html) |
| `directoryMode` | `String` | Similar to a UNIX permission, sets the directory mode of the directories included. THIS IS AN OCTAL VALUE. Format: (User)(Group)(Other) where each component is a sum of Read = 4, Write = 2, and Execute = 1. For example, the value 0755 translates to User read-write, Group and Other read-only. The default value is 0755. (more on unix-style permissions) (http://www.onlamp.com/pub/a/bsd/2000/09/06/FreeBSD_Basics.html) |
| `attachmentClassifier` | `String` | When specified, the attachmentClassifier will cause the assembler to look at artifacts attached to the module instead of the main project artifact. If it can find an attached artifact matching the specified classifier, it will use it; otherwise, it will throw an exception. (Since 2.2-beta-1) |
| `includeDependencies` | `boolean` | If set to true, the plugin will include the direct and transitive dependencies of of the project modules included here. Otherwise, it will only include the module packages only.<br>**Default value is**: `true`<br>. |
| `dependencySets/dependencySet*` | `List<DependencySet>` | **(Many)** Specifies which dependencies of the module to include in the assembly. A dependencySet is specified by providing one or more of <dependencySet> subelements. (Since 2.2-beta-1) |
| `unpack` | `boolean` | If set to true, this property will unpack all module packages into the specified output directory. When set to false module packages will be included as archives (jars).<br>**Default value is**: `true`<br>. |
| `unpackOptions` | `UnpackOptions` | Allows the specification of includes and excludes, along with filtering options, for items unpacked from a module artifact. (Since 2.2-beta-1) |

| | | |
|---|---|---|
| `outputFileNameMapping` | `String` | Sets the mapping pattern for all NON-UNPACKED dependencies included in this assembly. (Since 2.2-beta-2; 2.2-beta-1 uses ${artifactId}-${version}${dashClassifier?}.${extension} as default value) NOTE: If the dependencySet specifies unpack == true, outputFileNameMapping WILL NOT BE USED; in these cases, use outputDirectory. See the plugin FAQ for more details about entries usable in the outputFileNameMapping parameter.<br>**Default value is**:<br>`${module.artifactId}-${module.version}${dashClassifier?}.${module.extension}`<br>. |

# dependencySet

A dependencySet allows inclusion and exclusion of project dependencies in the assembly.

| Element | Type | Description |
|---|---|---|
| `outputDirectory` | `String` | Sets the output directory relative to the root of the root directory of the assembly. For example, "log" will put the specified files in the log directory, directly beneath the root of the archive. |
| `includes/include*` | `List<String>` | **(Many)** When <include> subelements are present, they define a set of artifact coordinates to include. If none is present, then <includes> represents all valid values. Artifact coordinates may be given in simple groupId:artifactId form, or they may be fully qualified in the form groupId:artifactId:type[:classifier]:version. Additionally, wildcards can be used, as in *:maven-* |
| `excludes/exclude*` | `List<String>` | **(Many)** When <exclude> subelements are present, they define a set of dependency artifact coordinates to exclude. If none is present, then <excludes> represents no exclusions. Artifact coordinates may be given in simple groupId:artifactId form, or they may be fully qualified in the form groupId:artifactId:type[:classifier]:version. Additionally, wildcards can be used, as in *:maven-* |
| `fileMode` | `String` | Similar to a UNIX permission, sets the file mode of the files included. THIS IS AN OCTAL VALUE. Format: (User)(Group)(Other) where each component is a sum of Read = 4, Write = 2, and Execute = 1. For example, the value 0644 translates to User read-write, Group and Other read-only. The default value is 0644 (more on unix-style permissions) (http://www.onlamp.com/pub/a/bsd/2000/09/06/FreeBSD_Basics.html) |
| `directoryMode` | `String` | Similar to a UNIX permission, sets the directory mode of the directories included. THIS IS AN OCTAL VALUE. Format: (User)(Group)(Other) where each component is a sum of Read = 4, Write = 2, and Execute = 1. For example, the value 0755 translates to User read-write, Group and Other read-only. The default value is 0755. (more on unix-style permissions) (http://www.onlamp.com/pub/a/bsd/2000/09/06/FreeBSD_Basics.html) |
| `useStrictFiltering` | `boolean` | When specified as true, any include/exclude patterns which aren't used to filter an actual artifact during assembly creation will cause the build to fail with an error. This is meant to highlight obsolete inclusions or exclusions, or else signal that the assembly descriptor is incorrectly configured. (Since 2.2)<br>**Default value is**: `false`<br>. |
| `outputFileNameMapping` | `String` | Sets the mapping pattern for all dependencies included in this assembly. (Since 2.2-beta-2; 2.2-beta-1 uses ${artifactId}-${version}${dashClassifier?}.${extension} as default value). See the plugin FAQ for more details about entries usable in the outputFileNameMapping parameter.<br>**Default value is**:<br>`${artifact.artifactId}-${artifact.version}${dashClassifier?}.${artifact.extension}`<br>. |
| `unpack` | `boolean` | If set to true, this property will unpack all dependencies into the specified output directory. When set to false dependencies will be includes as archives (jars). Can only unpack jar, zip, tar.gz, and tar.bz archives.<br>**Default value is**: `false`<br>. |
| `unpackOptions` | `UnpackOptions` | Allows the specification of includes and excludes, along with filtering options, for items unpacked from a dependency artifact. (Since 2.2-beta-1) |
| `scope` | `String` | Sets the dependency scope for this dependencySet.<br>**Default value is**: `runtime`<br>. |

| | | |
|---|---|---|
| `useProjectArtifact` | `boolean` | Determines whether the artifact produced during the current project's build should be included in this dependency set. (Since 2.2-beta-1)<br>**Default value is**: `true`<br><br>. |
| `useProjectAttachments` | `boolean` | Determines whether the attached artifacts produced during the current project's build should be included in this dependency set. (Since 2.2-beta-1)<br>**Default value is**: `false`<br><br>. |
| `useTransitiveDependencies` | `boolean` | Determines whether transitive dependencies will be included in the processing of the current dependency set. If true, includes/excludes/useTransitiveFiltering will apply to transitive dependency artifacts in addition to the main project dependency artifacts. If false, useTransitiveFiltering is meaningless, and includes/excludes only affect the immediate dependencies of the project. By default, this value is true. (Since 2.2-beta-1)<br>**Default value is**: `true`<br><br>. |
| `useTransitiveFiltering` | `boolean` | Determines whether the include/exclude patterns in this dependency set will be applied to the transitive path of a given artifact. If true, and the current artifact is a transitive dependency brought in by another artifact which matches an inclusion or exclusion pattern, then the current artifact has the same inclusion/exclusion logic applied to it as well. By default, this value is false, in order to preserve backward compatibility with version 2.1. This means that includes/excludes only apply directly to the current artifact, and not to the transitive set of artifacts which brought it in. (Since 2.2-beta-1)<br>**Default value is**: `false`<br><br>. |

## unpackOptions

Specifies options for including/excluding/filtering items extracted from an archive. (Since 2.2-beta-1)

| Element | Type | Description |
|---|---|---|
| `includes/include*` | `List<String>` | **(Many)** Set of file and/or directory patterns for matching items to be included from an archive as it is unpacked. Each item is specified as <include>some/path</include> (Since 2.2-beta-1) |
| `excludes/exclude*` | `List<String>` | **(Many)** Set of file and/or directory patterns for matching items to be excluded from an archive as it is unpacked. Each item is specified as <exclude>some/path</exclude> (Since 2.2-beta-1) |
| `filtered` | `boolean` | Whether to filter symbols in the files as they are unpacked from the archive, using properties from the build configuration. (Since 2.2-beta-1)<br>**Default value is**: `false`<br><br>. |
| `nonFilteredFileExtensions/nonFilteredFileExtension*` | `List<String>` | **(Many)** Additional file extensions to not apply filtering (Since 3.2.0) |
| `lineEnding` | `String` | Sets the line-endings of the files. (Since 2.2) Valid values:<br>• **"keep"** - Preserve all line endings<br>• **"unix"** - Use Unix-style line endings<br>• **"lf"** - Use a single line-feed line endings<br>• **"dos"** - Use DOS-style line endings<br>• **"crlf"** - Use Carraige-return, line-feed line endings |
| `useDefaultExcludes` | `boolean` | Whether standard exclusion patterns, such as those matching CVS and Subversion metadata files, should be used when calculating the files affected by this set. For backward compatibility, the default value is true. (Since 2.2)<br>**Default value is**: `true`<br><br>. |
| `encoding` | `String` | Allows to specify the encoding to use when unpacking archives, for unarchivers that support specifying encoding. If unspecified, archiver default will be used. Archiver defaults generally represent sane (modern) values. |

## file

A file allows individual file inclusion with the option to change the destination filename not supported by fileSets. Note: either source or sources is required

| Element | Type | Description |
|---|---|---|
| `source` | `String` | Sets the absolute or relative path from the module's directory of the file to be included in the assembly. |
| `sources/source*` | `List<String>` | **(Many)** Set of absolute or relative paths from the module's directory of the files be combined and included in the assembly. |
| `outputDirectory` | `String` | Sets the output directory relative to the root of the root directory of the assembly. For example, "log" will put the specified files in the log directory. |
| `destName` | `String` | Sets the destination filename in the outputDirectory. Default is the same name as the source's file. |
| `fileMode` | `String` | Similar to a UNIX permission, sets the file mode of the files included. THIS IS AN OCTAL VALUE. Format: (User)(Group)(Other) where each component is a sum of Read = 4, Write = 2, and Execute = 1. For example, the value 0644 translates to User read-write, Group and Other read-only. The default value is 0644 (more on unix-style permissions) (http://www.onlamp.com/pub/a/bsd/2000/09/06/FreeBSD_Basics.html) |
| `lineEnding` | `String` | Sets the line-endings of the files in this file. Valid values are:<br>• **"keep"** - Preserve all line endings<br>• **"unix"** - Use Unix-style line endings (i.e. "\n")<br>• **"lf"** - Use a single line-feed line endings (i.e. "\n")<br>• **"dos"** - Use DOS-/Windows-style line endings (i.e. "\r\n")<br>• **"windows"** - Use DOS-/Windows-style line endings (i.e. "\r\n")<br>• **"crlf"** - Use carriage-return, line-feed line endings (i.e. "\r\n") |
| `filtered` | `boolean` | Sets whether to determine if the file is filtered.<br>**Default value is**: `false`<br>. |

# repository

**Deprecated since model version 2.1.1.** Defines a Maven repository to be included in the assembly. The artifacts available to be included in a repository are your project's dependency artifacts. The repository created contains the needed metadata entries and also contains both sha1 and md5 checksums. This is useful for creating archives which will be deployed to internal repositories.

**NOTE:** Currently, only artifacts from the central repository are allowed.

| Element | Type | Description |
|---|---|---|
| `outputDirectory` | `String` | Sets the output directory relative to the root of the root directory of the assembly. For example, "log" will put the specified files in the log directory, directly beneath the root of the archive. |
| `includes/include*` | `List<String>` | **(Many)** When <include> subelements are present, they define a set of artifact coordinates to include. If none is present, then <includes> represents all valid values. Artifact coordinates may be given in simple groupId:artifactId form, or they may be fully qualified in the form groupId:artifactId:type[:classifier]:version. Additionally, wildcards can be used, as in *:maven-* |
| `excludes/exclude*` | `List<String>` | **(Many)** When <exclude> subelements are present, they define a set of dependency artifact coordinates to exclude. If none is present, then <excludes> represents no exclusions. Artifact coordinates may be given in simple groupId:artifactId form, or they may be fully qualified in the form groupId:artifactId:type[:classifier]:version. Additionally, wildcards can be used, as in *:maven-* |
| `fileMode` | `String` | Similar to a UNIX permission, sets the file mode of the files included. THIS IS AN OCTAL VALUE. Format: (User)(Group)(Other) where each component is a sum of Read = 4, Write = 2, and Execute = 1. For example, the value 0644 translates to User read-write, Group and Other read-only. The default value is 0644 (more on unix-style permissions) (http://www.onlamp.com/pub/a/bsd/2000/09/06/FreeBSD_Basics.html) |
| `directoryMode` | `String` | Similar to a UNIX permission, sets the directory mode of the directories included. THIS IS AN OCTAL VALUE. Format: (User)(Group)(Other) where each component is a sum of Read = 4, Write = 2, and Execute = 1. For example, the value 0755 translates to User read-write, Group and Other read-only. The default value is 0755. (more on unix-style permissions) (http://www.onlamp.com/pub/a/bsd/2000/09/06/FreeBSD_Basics.html) |

| | | |
|---|---|---|
| `includeMetadata` | `boolean` | If set to true, this property will trigger the creation of repository metadata which will allow the repository to be used as a functional remote repository.<br>**Default value is**: `false`<br><br>. |
| `groupVersionAlignments/groupVersionAlignment*` | `List<GroupVersionAlignment>` | **(Many)** Specifies that you want to align a group of artifacts to a specified version. A groupVersionAlignment is specified by providing one or more of <groupVersionAlignment> subelements. |
| `scope` | `String` | Specifies the scope for artifacts included in this repository. (Since 2.2-beta-1)<br>**Default value is**: `runtime`<br><br>. |

## groupVersionAlignment

Allows a group of artifacts to be aligned to a specified version.

| Element | Type | Description |
|---|---|---|
| `id` | `String` | The groupId of the artifacts for which you want to align the versions. |
| `version` | `String` | The version you want to align this group to. |
| `excludes/exclude*` | `List<String>` | **(Many)** When <exclude> subelements are present, they define the artifactIds of the artifacts to exclude. If none is present, then <excludes> represents no exclusions. An exclude is specified by providing one or more of <exclude> subelements. |