archetype:create-from-project

Full name:

org.apache.maven.plugins:maven-archetype-plugin:2.3:create-from-project

Description:

Creates an archetype project from the current project.

This goal reads your source and resource files, the values of its parameters, and properties you specify in a `.property` file, and uses them to create a Maven archetype project using the maven-archetype packaging. If you build the resulting project, it will create the archetype. You can then use this archetype to create new projects that resemble the original.

The maven-archetype-plugin uses Velocity to expand template files, and this documentation talks about 'Velocity Properties', which are values substituted into Velocity templates. See The Velocity User's Guide 🌐 for more information.

This goal modifies the text of the files of the current project to form the Velocity template files that make up the archetype.

GAV
The GAV values for the current project are replaced by properties: groupId, artifactId, and version. The user chooses new values for these when generating a project from the archetype.

package
All the files under one specified Java (or cognate) package are relocated to a project that the user chooses when generating a project. References to the class name are replaced by a property reference. For example, if the current project's sources are in the package `org.apache.saltedpeanuts`, then any example of the string `org.apache.saltedpeanuts` is replaced with the Velocity property reference `${packageName}`. When the user generates a project, this is in turn replaced by his or her choice of a package.

custom properties
You may identify additional strings that should be replaced by parameters. To add custom properties, you must use the `propertyFile` parameter to specify a property file. See the documentation for `propertyFile` for the details.

Note that you may need to edit the results of this goal. This goal has no way to exclude unwanted files, or add copyright notices to the Velocity templates, or add more complex elements to the archetype metadata file.

This goal also generates a simple integration-test that exercises the generated archetype.

Attributes:

- Requires a Maven project to be executed.   Maven
- Executes as an aggregator plugin.
- Invokes the execution of the lifecycle phase 🌐 `generate-sources` prior to executing itself.

Optional Parameters

Name	Type	Since	Description
archetypeFilteredExtentions	`String`	–	File extensions which are checked for project's text files (vs binary files). **User property is**: `archetype.filteredExtentions`.

archetypeLanguages	`String`	–	Directory names which are checked for project's sources main package. **User property is**: `archetype.languages`.
archetypePostPhase	`String`	–	The property telling which phase to call on the generated archetype. Interesting values are: `package`, `integration-test`, `install` and `deploy`. **Default value is**: `package`. **User property is**: `archetype.postPhase`.
archetypeRegistryFile	`File`	–	The location of the registry file. **Default value is**: `${user.home}/.m2/archetype.xml`.
defaultEncoding	`String`	–	Velocity templates encoding. Velocity **Default value is**: `UTF-8`. **User property is**: `archetype.encoding`.
interactive	`boolean`	–	Enable the interactive mode to define the archetype from the project. **Default value is**: `false`. **User property is**: `interactive`.
keepParent	`boolean`	–	POMs in archetype are created with their initial parent. This property is ignored when preserveCData is true. **User property is**: `archetype.keepParent`.
outputDirectory	`File`	–	The directory where the archetype should be created. **Default value is**: `${project.build.directory}/generated-sources/archetype`.
packageName	`String`	–	The package name for Java source files to be incorporated in the archetype and and relocated to the package that the user selects. **User property is**: `packageName`.
partialArchetype	`boolean`	–	Create a partial archetype. **User property is**: `archetype.partialArchetype`.
preserveCData	`boolean`	–	Create pom's velocity templates with CDATA preservation. This uses the `String.replaceAll()` method and risks to have some overly replacement capabilities (beware of '1.0' value). **User property is**: `archetype.preserveCData`.
propertyFile	`File`	–	The property file that holds the plugin configuration. If this is provided, then the plugin reads properties from here. The properties in here can be standard properties listed below or custom properties for this archetype. The standard properties are below. Several of them overlap

parameters of this goal; it's better to just set the parameter.

package
 See the packageName parameter.
archetype.languages
 See the archetypeLanguages parameter.
groupId
 The default groupId of the generated project.
artifactId
 The default artifactId of the generated project.
version
 The default version of the generated project.
archetype.filteredExtensions
 See the filteredExensions parameter.

Custom Properties
 Custom properties allow you to replace some constant values in the project's files with Velocity macro references. When a user generates a project from your archetype he or she gets the opportunity to replace the value from the source project.

Custom property names **may not contain the '.' character**.

For example, if you include a line like the following in your property file:

```
cxf-version=2.5.1-SNAPSHOT
```

the plugin will search your files for the string

```
2.5.1-SNAPSHOT
```

and replace them with references to a velocity macro

```
cxf-version
```

. It will then list

```
cxf-version
```

as a

```
requiredProperty
```

in the archetype-metadata.xml, with

```
2.5.1-SNAPSHOT
```

as the default value.

User property is: `archetype.properties`.

testMode   `boolean`   `-`   (no description)
User property is: `testMode`.