**BUILD JENKINS PLUGIN**

(*Using Eclipse IDE*)

**-ARKA DAS**

***Associate Consultant, Capgemini***

**Created On:**

# **INTRODUCTION**

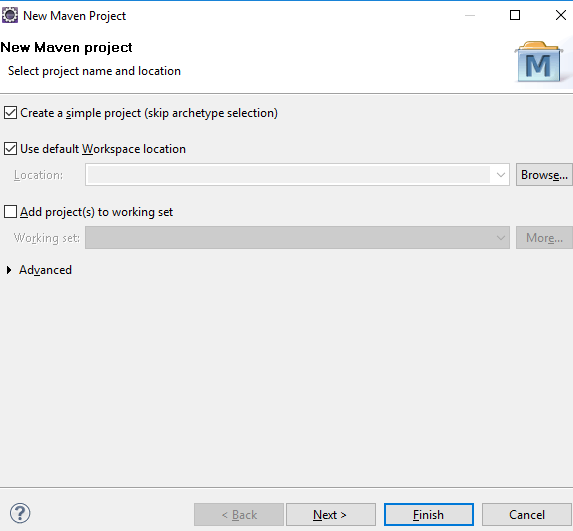
Using Eclipse IDE, we can easily build any plugin for Jenkins. Jenkins support only **HPI** package. So, we have to build HPI package.

\*\*You have to install **jenkins.war** in your system.

We have to use **MAVEN** project to build any HPI package. But in Eclipse, we can’t create HPI package from pom of a MAVEN project. We have to follow the below procedures to create HPI project structure within a MAVEN project-

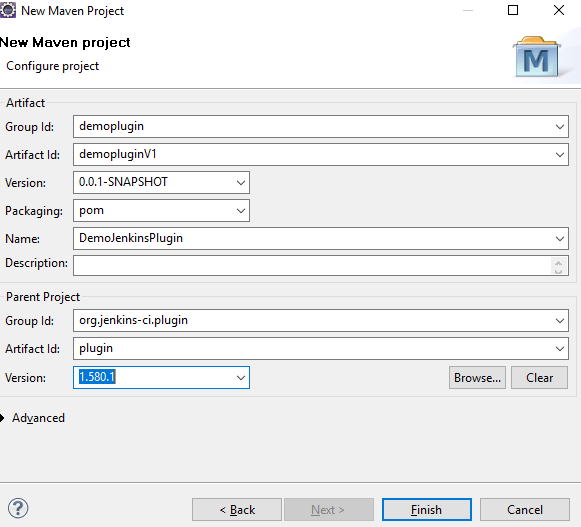
**STEP 1-**

We have to create MAVEN project following below screenshot.



**STEP 2-**

In next step, we have to provide Group Id, packaging type etc like below.



**STEP 4-**

After creating MAVEN project, we have to paste below pom.xml

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<project xmlns=*"http://maven.apache.org/POM/4.0.0"* xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"* xsi:schemaLocation=*"http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd"*>

<modelVersion>4.0.0</modelVersion>

<groupId>**YOUR PROJECT NAME**</groupId>

<artifactId> **YOUR PROJECT NAME** </artifactId>

<version>0.0.1-SNAPSHOT</version>

<packaging>pom</packaging>

<parent>

<groupId>org.jenkins-ci</groupId>

<artifactId>jenkins</artifactId>

<version>1.37</version>

</parent>

<name>**YOUR PLUGIN NAME**</name>

<description>**YOUR PLUGIN DESCRIPTION**</description>

<licenses>

<license>

<name>The Apache Software License, Version 2.0</name>

<url>http://www.apache.org/licenses/LICENSE-2.0.txt</url>

</license>

</licenses>

<scm>

<connection>scm:git:git://github.com/jenkinsci/maven-hpi-plugin.git</connection>

<developerConnection>scm:git:ssh://git@github.com/jenkinsci/maven-hpi-plugin.git</developerConnection>

<url>https://github.com/jenkinsci/maven-hpi-plugin</url>

<tag>HEAD</tag>

</scm>

<ciManagement>

<system>jenkins</system>

<url>https://github.com/jenkinsci/maven-hpi-plugin/</url>

</ciManagement>

<distributionManagement>

<site>

<!-- if this deployment fails, it's probably you are missing the entry to set the username in ~/.m2/settings.xml -->

<!-- <server>

<id>github.com</id>

<username>git</username>

</server> -->

<id>github.com</id>

<url>gitsite:git@github.com/jenkinsci/maven-hpi-plugin.git</url>

</site>

</distributionManagement>

<properties>

<!-- used for escape -->

<begin>${${$}}{</begin>

<end>$}</end>

<dollar>$</dollar>

<openbracket>{</openbracket>

<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>

<java.level>7</java.level>

</properties>

<repositories>

<repository>

<id>repo.jenkins-ci.org</id>

<url>http://repo.jenkins-ci.org/public/org/jvnet/maven-jellydoc-plugin/maven-jellydoc-plugin/1.5/</url>

</repository>

</repositories>

<pluginRepositories>

<pluginRepository>

<id>repo.jenkins-ci.org</id>

<url>http://repo.jenkins-ci.org/public/org/jvnet/maven-jellydoc-plugin/maven-jellydoc-plugin/1.5/</url>

</pluginRepository>

</pluginRepositories>

<dependencies>

<dependency>

<groupId>org.jenkins-ci.main</groupId>

<artifactId>jenkins-war</artifactId>

<type>war</type>

<version>2.89<!-- replace this with the version you want--></version>

<scope>test</scope>

</dependency>

<dependency>

<groupId>org.apache.hadoop</groupId>

<artifactId>hadoop-common</artifactId>

<version>2.7.2</version>

<exclusions>

<exclusion>

<groupId>jdk.tools</groupId>

<artifactId>jdk.tools</artifactId>

</exclusion>

</exclusions>

</dependency>

<!-- <dependency>

<groupId>org.apache.maven.plugin-tools</groupId>

<artifactId>maven-plugin-annotations</artifactId>

<version>3.5</version>

annotations are not needed for plugin execution, so exclude using provided scope

<scope>provided</scope>

</dependency> -->

<dependency>

<groupId>org.codehaus.plexus</groupId>

<artifactId>plexus-utils</artifactId>

<!-- generated help mojo has a dependency on plexus utils -->

<version>3.0.1</version>

</dependency>

<dependency>

<groupId>com.ibm.icu</groupId>

<artifactId>icu4j</artifactId>

<version>52.1</version>

<!-- stapler plugin pulls jaxen pulls ancient ver that blows up annotation processing -->

</dependency>

<dependency>

<groupId>com.sun.codemodel</groupId>

<artifactId>codemodel</artifactId>

<version>2.6</version>

</dependency>

<dependency>

<groupId>org.kohsuke.stapler</groupId>

<artifactId>stapler-groovy</artifactId>

<version>1.237</version>

</dependency>

<dependency>

<groupId>org.apache.maven</groupId>

<artifactId>maven-plugin-api</artifactId>

<version>2.2.0</version>

</dependency>

<dependency>

<groupId>org.apache.maven</groupId>

<artifactId>maven-project</artifactId>

<version>2.2.0</version>

</dependency>

<dependency>

<groupId>org.apache.maven</groupId>

<artifactId>maven-archiver</artifactId>

<version>2.0.1</version>

</dependency>

<dependency>

<groupId>org.apache.maven</groupId>

<artifactId>maven-artifact</artifactId>

<version>2.2.0</version>

</dependency>

<dependency>

<groupId>org.eclipse.jetty</groupId>

<artifactId>jetty-maven-plugin</artifactId>

<version>9.4.5.v20170502</version>

</dependency>

<dependency>

<groupId>net.java.sezpoz</groupId>

<artifactId>sezpoz</artifactId>

<version>1.9</version>

</dependency>

<!-- for copy-dependency-plugins -->

<dependency>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-dependency-plugin</artifactId>

<version>2.8</version>

</dependency>

<dependency>

<groupId>org.codehaus.plexus</groupId>

<artifactId>plexus-interactivity-api</artifactId>

<version>1.0-alpha-4</version>

</dependency>

<!-- for using annotation processor -->

<dependency>

<groupId>org.kohsuke.stapler</groupId>

<artifactId>maven-stapler-plugin</artifactId>

<version>1.16</version>

<!-- make sure to exclude servlet so that the one in Jetty (which has different groupId) will win -->

<exclusions>

<exclusion>

<groupId>javax.servlet</groupId>

<artifactId>servlet-api</artifactId>

</exclusion>

</exclusions>

</dependency>

<dependency>

<groupId>org.codehaus.plexus</groupId>

<artifactId>plexus-component-annotations</artifactId>

<version>1.6</version>

</dependency>

</dependencies>

<build>

<pluginManagement>

<plugins>

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-site-plugin</artifactId>

<version>3.6</version>

<dependencies>

<dependency>

<groupId>com.github.stephenc.wagon</groupId>

<artifactId>wagon-gitsite</artifactId>

<version>0.4.1</version>

</dependency>

<dependency> <!-- TODO https://maven.apache.org/plugins/maven-site-plugin/migrate.html 1.7 fails with: org.apache.velocity.exception.ParseErrorException: Encountered "source" at META-INF/maven/site.vm[line 1039, column 52] -->

<groupId>org.apache.velocity</groupId>

<artifactId>velocity</artifactId>

<version>1.5</version>

</dependency>

</dependencies>

</plugin>

<!--This plugin's configuration is used to store Eclipse m2e settings only. It has no influence on the Maven build itself.-->

<plugin>

<groupId>org.eclipse.m2e</groupId>

<artifactId>lifecycle-mapping</artifactId>

<version>1.0.0</version>

<configuration>

<lifecycleMappingMetadata>

<pluginExecutions>

<pluginExecution>

<pluginExecutionFilter>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-antrun-plugin</artifactId>

<versionRange>[1.3,)</versionRange>

<goals>

<goal>run</goal>

</goals>

</pluginExecutionFilter>

<action>

<ignore/>

</action>

</pluginExecution>

</pluginExecutions>

</lifecycleMappingMetadata>

</configuration>

</plugin>

</plugins>

</pluginManagement>

<plugins>

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-release-plugin</artifactId>

<version>2.5.1</version>

</plugin>

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-plugin-plugin</artifactId>

<version>3.5</version>

<configuration>

<goalPrefix>hpi</goalPrefix>

<skipErrorNoDescriptorsFound>true</skipErrorNoDescriptorsFound>

<!-- see MNG-5346 -->

</configuration>

<!-- <executions>

<execution>

<id>mojo-descriptor</id>

<goals>

<goal>descriptor</goal>

</goals>

</execution>

<execution>

<id>help-goal</id>

<goals>

<goal>helpmojo</goal>

</goals>

</execution>

</executions> -->

</plugin>

<plugin>

<groupId>org.codehaus.mojo</groupId>

<artifactId>mrm-maven-plugin</artifactId>

<version>1.1.0</version>

<configuration>

<propertyName>repository.proxy.url</propertyName>

<repositories>

<proxyRepo/>

</repositories>

</configuration>

</plugin>

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-invoker-plugin</artifactId>

<version>2.0.0</version>

<configuration>

<projectsDirectory>src/it</projectsDirectory>

<cloneProjectsTo>${project.build.directory}/its</cloneProjectsTo>

<localRepositoryPath>${basedir}/target/local-repo</localRepositoryPath>

<settingsFile>src/it/settings.xml</settingsFile>

<showErrors>true</showErrors>

<streamLogs>true</streamLogs>

<pomIncludes>

<pomInclude>\*/pom.xml</pomInclude>

</pomIncludes>

<postBuildHookScript>verify</postBuildHookScript>

<filterProperties>

<repository.proxy.url>${repository.proxy.url}</repository.proxy.url>

</filterProperties>

</configuration>

</plugin>

<plugin>

<groupId>org.codehaus.mojo</groupId>

<artifactId>animal-sniffer-maven-plugin</artifactId>

<version>1.15</version>

<executions>

<execution>

<goals>

<goal>check</goal>

</goals>

</execution>

</executions>

<configuration>

<signature>

<groupId>org.codehaus.mojo.signature</groupId>

<artifactId>java1${java.level}</artifactId>

<version>1.0</version>

</signature>

</configuration>

</plugin>

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-compiler-plugin</artifactId>

<version>3.6.1</version>

<configuration>

<source>1.7</source>

<target>1.7</target>

</configuration>

</plugin>

<plugin>

<groupId>org.codehaus.plexus</groupId>

<artifactId>plexus-component-metadata</artifactId>

<version>1.6</version>

<!-- <executions>

<execution>

<goals>

<goal>generate-metadata</goal>

</goals>

</execution>

</executions> -->

</plugin>

</plugins>

<extensions>

<extension>

<groupId>org.apache.maven.wagon</groupId>

<artifactId>wagon-ssh</artifactId>

<version>1.0-beta-7</version>

</extension>

</extensions>

</build>

<reporting>

<plugins>

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-plugin-plugin</artifactId>

<version>3.5</version>

</plugin>

</plugins>

</reporting>

<profiles>

<profile>

<id>run-its</id>

<build>

<plugins>

<plugin>

<groupId>org.codehaus.mojo</groupId>

<artifactId>mrm-maven-plugin</artifactId>

<executions>

<execution>

<goals>

<goal>start</goal>

<goal>stop</goal>

</goals>

</execution>

</executions>

</plugin>

<plugin>

<artifactId>maven-invoker-plugin</artifactId>

<executions>

<execution>

<id>integration-test</id>

<goals>

<goal>install</goal>

<goal>run</goal>

</goals>

<configuration>

<streamLogs>false</streamLogs>

<debug>true</debug>

</configuration>

</execution>

</executions>

</plugin>

</plugins>

</build>

</profile>

</profiles>

<modules>

<module>**YOUR MODULE NAME**</module>

</modules>

</project>

**STEP 5-**

We have to run this project with only the above pom (as **MAVEN Build**) to create the **hpi** project skeleton. Goal has given below-

hpi:create

**\*\*** We can generate out **hpi** package with below goal-

package

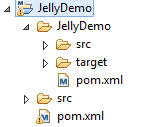
\*\* We can run our **hpi** package with below goal –

hpi:run

\*\*\*To Know more please visit to <https://jenkinsci.github.io/maven-hpi-plugin/>

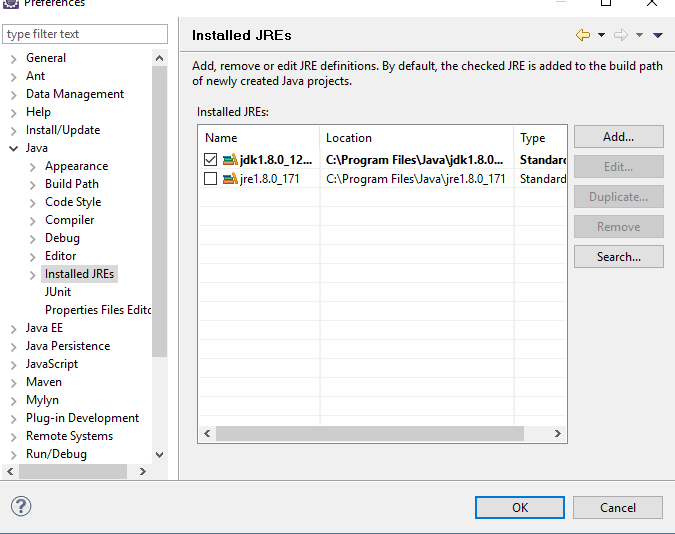
**STEP 6-**

After created HPI package your directory structure will look like below…



\*\***target** directory will appear once we generate package.

\*\*Before create package we have to change from **JRE** to **JDK** in **Installed JREs**.



# **Create HPI plugin for Jenkins**

## **Package Introduction**

HPI skeleton holds to major directive – 1. src and 2. Resource. Both directive is in one single directive called **main**. We can create custom package also.

src directive contains all class files (directly related with UI) and resource directive contains all UI files. UI files can design in either **jelly** or **groovy.**

## **Implement main/src package**

## **Implement main/resource package**

## **Build HPI package**

Run project as maven build with **package** goal. HPI package will create within **target** directive.

# **Integrate with Jenkins**

Deploy Jenkins war in your local server (available in internet). After that launch and setup Jenkins. Once installation has completed, open manage plugin option. Then click on **Advance** tab and upload your **hpi** package.

Once upload has completed successfully, we need to rerun our Jenkins to see this plugin as installed.

# **Sample Plugin Code**

### **Git Hub Link-**

<https://github.com/dasarka/Jenkins-Plugin-Demo>

# **References**

1. <https://jenkinsci.github.io/maven-hpi-plugin/jar-mojo.html>
2. <https://jenkins.io/doc/developer/tutorial/run/>
3. <https://wiki.jenkins.io/display/JENKINS/Plugin+tutorial>
4. <https://coderanch.com/t/592628/build-tools/packaging-jenkins-plugin-tutorial>
5. <https://hub.packtpub.com/maven-and-jenkins-plugin/>