MRJARs with Maven

1

Outline

- Overview of Maven plugins
- Configuring Maven Compiler plugin for MRJARs

2

Overview of Maven Plugins

3

How Maven Plugins Work ?

- **Maven is actually a plugin execution framework where every task is actually done by plugins. Maven Plugins are generally used to −**
  - compile code files
  - unit testing of code
  - create jar file
  - create war file
  - create project documentation
  - create project reports

- **A plugin generally provides a set of goals, which can be executed using the following syntax −**

  mvn [plugin-name]:[goal-name]

  For example, a Java project can be compiled with the maven-compiler-plugin's compile-goal using.

  mvn compiler:compile

4

How Maven Plugins Work ?

- **Exercice : configure the exec:java**

```
<plugin>
   <groupId>org.codehaus.mojo</groupId>
   <artifactId>exec-maven-plugin</artifactId>
   <version>3.0.0</version>
  <configuration>
    <mainClass>be.justice.MyApp</mainClass>
  </configuration>
</plugin>
```

  - So, to trigger the plugin from the command line, just run:

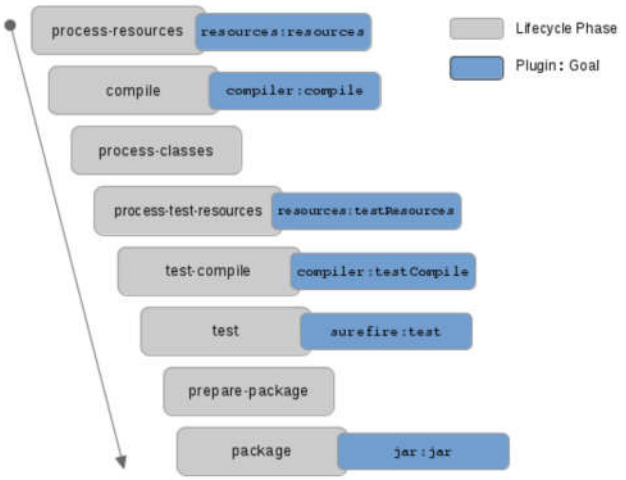
```
mvn exec:java
```

Now, if you want to execute the exec:java goal as part of your standard build, you'll need to bind the goal to a particular phase of the default lifecycle.

5

Maven default lifecycle

- **Lifecycle is composed of Phases**
- **For each phase, a specific Plugin:goal is performs the processing**



6

Maven default lifecycle

- **Compiler Pluging is bound to these phases:**
  - compile (goal compile)
  - test-compile (goal test-compile)
- **Exercise : Bind the exec plugin to the package phase**

```
<plugin>
   <groupId>org.codehaus.mojo</groupId>
   <artifactId>exec-maven-plugin</artifactId>
   <version>3.0.0</version>
  <executions>
    <execution>
      <id>my-execution</id>
      <phase>package</phase>
      <goals>
        <goal>java</goal>
      </goals>
    </execution>
  </executions>
  <configuration>
    <mainClass>be.justice.MyApp</mainClass>
  </configuration>
</plugin>
```

  - `mvn package` command witl trigger the plugin

7

Compiler plugin Configurations and Executions

- **Configure Compiler to use Java 9 for the goal compile**

```
<plugin>
  <artifactId>maven-compiler-plugin</artifactId>
  <version>3.8.0</version>
  <executions>
     <execution>
       <id>compile-java-9</id>
       <goals>
          <goal>compile</goal>
       </goals>
       <configuration>
          <release>9</release>
        </configuration>
      </execution>
   <executions>
<plugin>
```


8

Configure Compiler Plugin for MRJARs

9

Configure Compiler Plugin for MRJARs

- **Steps**
  1. Configure how Compiler Plugin should build the releases
     - Each release should have an execution entry
  2. Configure how JAR Plugin should produce the MRJAR

10

Step 1. Building the MRJAR

```
<executions>
    <execution>
       <id>compile-java-8</id>
       <goals>
           <goal>compile</goal>
       </goals>
       <configuration>
         <release>8</release>
       </configuration>
    </execution>
    <execution>
       <id>compile-java-9</id>
       <phase>compile</phase>
       <goals>
           <goal>compile</goal>
       </goals>
       <configuration>
           <release>9</release>
           <compileSourceRoots>
                <compileSourceRoot> ${project.basedir}/src/main/java9
                </compileSourceRoot>
           </compileSourceRoots>
           <outputDirectory>${project.build.outputDirectory}/META-INF/versions/9
           </outputDirectory>
       </configuration>
     </execution>
```

- **Invoke the plugin with :**
  `mvn compile`

11

12

Step 2. MR Jarring

- JAR Plugin configuration for MRJAR

```
<plugin>
   <artifactId>maven-jar-plugin</artifactId>
   <version>3.0.2</version>
   <configuration>
       <release>9</release>
       <compileSourceRoots>
           <compileSourceRoot>
                ${project.basedir}/src/main/java9
           </compileSourceRoot>
       </compileSourceRoots>
       <multiReleaseOutput>true</multiReleaseOutput>
    </configuration>
</plugin>
```

- Invoke the plugin with :
  `mvn package`

Testing the MRJAR

- The sample project : MyProject-Java-9-MRJAR-Maven
- Calling the java-9 release

```
java -cp MyProject-Java-9-MRJAR-Maven-0.0.1-SNAPSHOT.jar be.justice.App
```

- Calling the java-8 release

```
"C:\Program Files\OpenJDK\openjdk-8u292-b10\bin\java"
 -cp MyProject-Java-9-MRJAR-Maven-0.0.1-SNAPSHOT.jar be.justice.App
```