Exercise: Create and build a Java project with Maven

In this exercise you create a Java project with the Maven command line and build this project.

5.1. Project generation

In this project we use the scaffolding functionality of Maven to create a Java project. To avoid the interactive mode, all required properties are passed directly to the command. Otherwise Maven asks for all the required parameters. Enter the following into one line in the command shell(the backslash masks the line breaks).

mvn archetype:generate -DgroupId=com.vogella.build.maven.java \ -DartifactId=com.vogella.build.maven.java \ -DarchetypeArtifactId=maven-archetype-quickstart \ -DinteractiveMode=false

With this command Maven generates a Java project.

|  |  |
| --- | --- |
|  | If this is the first time you execute this goal, this may takes some time. It also produces additional output compared to a second run. This is because Maven first loads all required plug-ins and artifacts for the project generation from the Maven central repository. |

5.2. Review generated project

Validate that Maven generated a project on your file system similar to the following structure.

com.vogella.build.maven.java/ ├── pom.xml └── src ├── main │   └── java │   └── com │   └── vogella │   └── build │   └── maven │   └── java │   └── App.java └── test └── java └── com └── vogella └── build └── maven └── java └── AppTest.java

You have generated a whole Maven project structure with Java source code. Maven created a App.java class in the *./src/main/* folder, which is just a simple "Hello World" program. It also created an example test class in *./src/test/*. In the root folder there is a *pom.xml* file.

<project> <modelVersion>4.0.0</modelVersion> <groupId>com.vogella.build.maven.java</groupId> <artifactId>com.vogella.build.maven.java</artifactId> <packaging>jar</packaging> <version>1.0-SNAPSHOT</version> <name>com.vogella.build.maven.java</name> <url>http://maven.apache.org</url> <dependencies> <dependency> <groupId>junit</groupId> <artifactId>junit</artifactId> <version>3.8.1</version> <scope>test</scope> </dependency> </dependencies> </project>

5.3. Compile your sources

Now you want to compile your Java sources. For this switch on the command line into our projects root directory and trigger the following Maven command.

$ mvn compile

[INFO] Scanning for projects...

[INFO]

[INFO] ------------------------------------------------------------------------

[INFO] Building com.vogella.build.maven.java 1.0-SNAPSHOT

[INFO] ------------------------------------------------------------------------

[INFO]

[INFO] --- maven-resources-plugin:2.6:resources (default-resources) ...

[WARNING] Using platform encoding (UTF-8 actually) to copy filtered resources..

[INFO] skip non existing resourceDirectory .....

[INFO]

[INFO] --- maven-compiler-plugin:2.5.1:compile (default-compile)...

[INFO] Nothing to compile - all classes are up to date

[INFO] ------------------------------------------------------------------------

[INFO] BUILD SUCCESS

[INFO] ------------------------------------------------------------------------

[INFO] Total time: 0.755s

[INFO] Finished at: Thu Oct 17 00:59:13 CEST 2013

[INFO] Final Memory: 7M/106M

[INFO] ------------------------------------------------------------------------

Maven now runs through all life cycle phases, which were needed by the compile phase. It resolves the dependencies, loads the JUnit artifact and built the sources. It even executed the JUnit test.

5.4. Create a JAR file

Now you want to create an executable JAR file out of our project. The package goal creates a deployable JAR file.

To ensure previous build artifacts are removed, you can use the clean goal.

mvn clean package

Afterwards you can run the packed program.

$ java -cp target/com.vogella.build.maven.java-1.0-SNAPSHOT.jar \

com.vogella.build.maven.java.App

Hello World!

$ cd com.vogella.build.maven.java

$ mvn package

...

-------------------------------------------------------

T E S T S

-------------------------------------------------------

Running com.vogella.build.maven.java.AppTest

Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.014 sec

Results :

Tests run: 1, Failures: 0, Errors: 0, Skipped: 0

[INFO]

[INFO] --- maven-jar-plugin:2.4:jar (default-jar) @ com.vogella.build.maven.java ---

[INFO] ------------------------------------------------------------------------

[INFO] BUILD SUCCESS

[INFO] ------------------------------------------------------------------------

[INFO] Total time: 1.631s

[INFO] Finished at: Thu Oct 17 01:05:11 CEST 2013

[INFO] Final Memory: 11M/102M

[INFO] ------------------------------------------------------------------------

5.5. Running the test

Instead of running a full build with packaging, it is also possible to just run the test phases of the Maven life cycle.

$ mvn test

....

-------------------------------------------------------

T E S T S

-------------------------------------------------------

Running com.vogella.build.maven.java.AppTest

Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.003 sec

Results :

Tests run: 1, Failures: 0, Errors: 0, Skipped: 0

[INFO] ------------------------------------------------------------------------

[INFO] BUILD SUCCESS

[INFO] ------------------------------------------------------------------------

.....

5.6. Remove all build results / Clean the project

To clean the project and so remove the generated files in the *./target/* folder, run the following command.

$ mvn clean