

# MAVEN - BUILD & TEST PROJECT

<http://www.tutorialspoint.com>

Copyright © tutorialspoint.com

What we learnt in Project Creation chapter is how to create a Java application using Maven. Now we'll see how to build and test the application.

Go to C:/MVN directory where you've created your java application. Open *consumerBanking* folder. You will see the **POM.xml** file with following contents.

```
<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>com.companyname.projectgroup</groupId>
  <artifactId>project</artifactId>
  <version>1.0</version>
  <dependencies>
    <dependency>
      <groupId>junit</groupId>
      <artifactId>junit</artifactId>
      <version>3.8.1</version>
    </dependency>
  </dependencies>
</project>
```

Here you can see, Maven already added Junit as test framework. By default Maven adds a source file **App.java** and a test file **AppTest.java** in its default directory structure discussed in previous chapter.

Let's open command console, go the C:\MVN\consumerBanking directory and execute the following **mvn** command.

```
C:\MVN\consumerBanking>mvn clean package
```

Maven will start building the project.

```
[INFO] Scanning for projects...
[INFO]
[INFO] -----
[INFO] Building consumerBanking 1.0-SNAPSHOT
[INFO] -----
[INFO]
[INFO] --- maven-clean-plugin:2.5:clean (default-clean) @ consumerBanking ---
[INFO] Deleting C:\MVN\consumerBanking\target
[INFO]
[INFO] --- maven-resources-plugin:2.6:resources (default-resources) @ consumerBanking ---
[WARNING] Using platform encoding (Cp1252 actually) to copy filtered resources, i.e.
build is platform dependent!
[INFO] skip non existing resourceDirectory C:\MVN\consumerBanking\src\main\resources
[INFO]
[INFO] --- maven-compiler-plugin:3.1:compile (default-compile) @ consumerBanking ---
[INFO] Changes detected - recompiling the module!
[WARNING] File encoding has not been set, using platform encoding Cp1252, i.e. build is
platform dependent!
[INFO] Compiling 1 source file to C:\MVN\consumerBanking\target\classes
[INFO]
[INFO] --- maven-resources-plugin:2.6:testResources (default-testResources) @
consumerBanking ---
[WARNING] Using platform encoding (Cp1252 actually) to copy filtered resources,
i.e. build is platform dependent!
[INFO] skip non existing resourceDirectory C:\MVN\consumerBanking\src\test\resources
[INFO]
[INFO] --- maven-compiler-plugin:3.1:testCompile (default-testCompile) @ consumerBanking
```

```

---
[INFO] Changes detected - recompiling the module!
[WARNING] File encoding has not been set, using platform encoding Cp1252, i.e. b
uild is platform dependent!
[INFO] Compiling 1 source file to C:\MVN\consumerBanking\target\test-classes
[INFO]
[INFO] --- maven-surefire-plugin:2.12.4:test (default-test) @ consumerBanking ---
[INFO] Surefire report directory: C:\MVN\consumerBanking\target\surefire-reports

-----
T E S T S
-----
Running com.companyname.bank.AppTest
Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.063 sec

Results :

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

[INFO]
[INFO] --- maven-jar-plugin:2.4:jar (default-jar) @ consumerBanking ---
[INFO] Building jar: C:\MVN\consumerBanking\target\consumerBanking-1.0-SNAPSHOT.jar
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 14.406 s
[INFO] Finished at: 2015-09-27T17:58:06+05:30
[INFO] Final Memory: 14M/247M
[INFO] -----

```

You've built your project and created final jar file, following are the key learning concepts

- We give maven two goals, first to clean the target directory *clean* and then package the project build output as *jar* package.
- Packaged jar is available in consumerBanking\target folder as consumerBanking-1.0-SNAPSHOT.jar.
- Test reports are available in consumerBanking\target\surefire-reports folder.
- Maven compiled source code files and then test source code files.
- Then Maven run the test cases.
- Finally Maven created the package.

Now open command console, go the C:\MVN\consumerBanking\target\classes directory and execute the following java command.

```
C:\MVN\consumerBanking\target\classes>java com.companyname.bank.App
```

You will see the result

```
Hello World!
```

## Adding Java Source Files

Let's see how we can add additional Java files in our project. Open C:\MVN\consumerBanking\src\main\java\com\companyname\bank folder, create Util class in it as Util.java.

```

package com.companyname.bank;

public class Util
{
    public static void printMessage(String message){

```

```
        System.out.println(message);
    }
}
```

Update App class to use Util class.

```
package com.companyname.bank;

/**
 * Hello world!
 */
public class App
{
    public static void main( String[] args )
    {
        Util.printMessage("Hello World!");
    }
}
```

Now open command console, go the C:\MVN\consumerBanking directory and execute the following **mvn** command.

```
C:\MVN\consumerBanking>mvn clean compile
```

After Maven build is successful, go the C:\MVN\consumerBanking\target\classes directory and execute the following java command.

```
C:\MVN\consumerBanking\target\classes>java -cp com.companyname.bank.App
```

You will see the result

```
Hello World!
```

```
Loading [Mathjax]/jax/output/HTML-CSS/fonts/TeX/fontdata.js
```