Creating Maven Project

This lesson gives a brief introduction to Maven. It is more like a guide to set a Maven project on our system, depending on its requirements.

WE'LL COVER THE FOLLOWING

What is Maven?

Apache Maven is a software project management and comprehension tool that can manage a project's build, reporting, and documentation from a central piece of information, making it a complete build lifecycle framework.

System requirements

Maven 3.3+ requires **JDK 1.7** or above to execute. They still allow you to build against 1.3 and other JDK versions.

Installation

The current latest Maven can be downloaded from apache-maven-3.6.3.

To find the latest version, please follow the link.

The downloaded binary needs to be added to classpath after extraction.

Windows

```
export PATH=%PATH%;<path\apache-maven-3.6.3\bin>;
```

Linux / Mac

```
export PATH=$PATH:<path/apache-maven-3.6.3/bin>
```

Alternatively, we can install using `brew`:

```
brew install maven
```

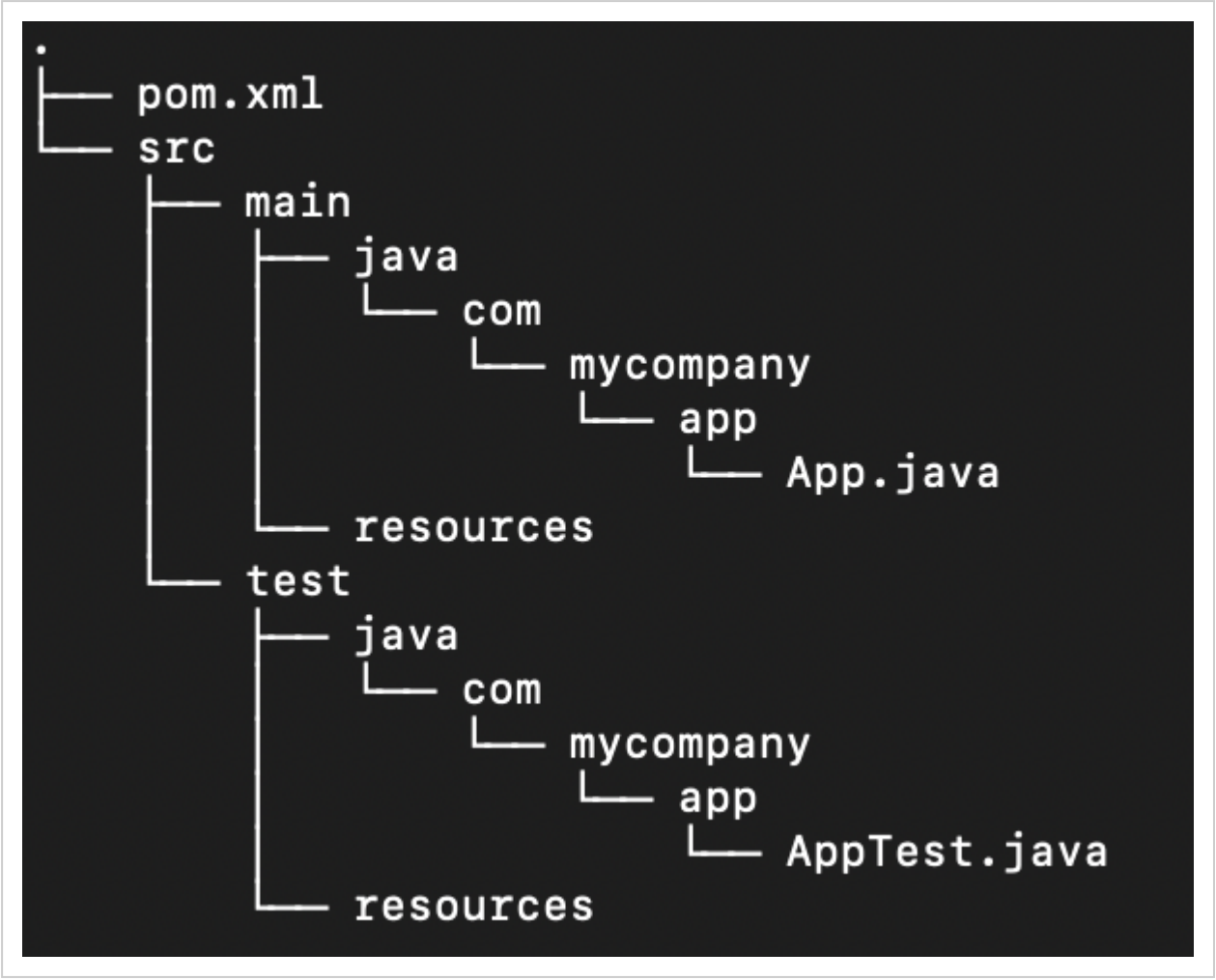
For installing `brew`, please follow the link.

Creating Maven project from the command line

```
mvn archetype:generate -DgroupId=com.mycompany.app -DartifactId=my-app -Da
rchetypeArtifactId=maven-archetype-quickstart -DarchetypeVersion=1.4 -Dint
eractiveMode=false
```

- `archetypeArtifactId`: a starter template to use for creating the basic project structure.
- `groupId`, `artifactId`, and `version`: this combination enables you to identify an artifact uniquely. groupId is mostly the company's domain reversed. For example, `groupId` of `example.com` is created as com.example. Version can be in format as `<major.version>.<minor.version>.<patch.number>-<RELEASE/SNAPSHOT>`. For example, `1.0.0-SNAPSHOT` or `1.0.0-RELEASE`. Please follow the link for more information.

By running the above command, a basic project structure will be created.



Building project and running test

```
mvn clean compile test surefire-report:report
```

Running the above command, the `build` folder (**target** incase of Maven) will be cleaned up, the project will be *compiled* after downloading all the dependencies mentioned in ***pom.xml*** and cached in `${user.home}/.m2` folder and *test* will be run with a generation of **sure-fire** reports at the end of the test run.

For generating `sure-fire` reports, please ensure the plugin is added in ***pom.xml*** as:

```
<plugin>
    <artifactId>maven-surefire-plugin</artifactId>
    <version>2.22.1</version>
</plugin>
```

The generated report can be found at `${project.dir}/target/site/surefire-`

`report.html`

See what a basic SureFire HTML Report looks like.

Surefire Report

Summary

[Summary] [Package List] [Test Cases]

Tests	Errors	Failures	Skipped	Success Rate	Time
1	0	0	0	100%	0.022

Note: failures are anticipated and checked for with assertions while errors are unanticipated.

Package List

[Summary] [Package List] [Test Cases]

Package	Tests	Errors	Failures	Skipped	Success Rate	Time
com.mycompany.app	1	0	0	0	100%	0.022

Note: package statistics are not computed recursively, they only sum up all of its testsuites numbers.

com.mycompany.app

	Class	Tests	Errors	Failures	Skipped	Success Rate	Time
	AppTest	1	0	0	0	100%	0.022

Test Cases

[Summary] [Package List] [Test Cases]

AppTest

	shouldAnswerWithTrue	0.001

That is all for building the Maven project and running it. In the next lesson, you'll learn how to build a project in the Gradle environment.