Programming in Java with IntelliJ

SWEN1 | BIF3

Compiling Code

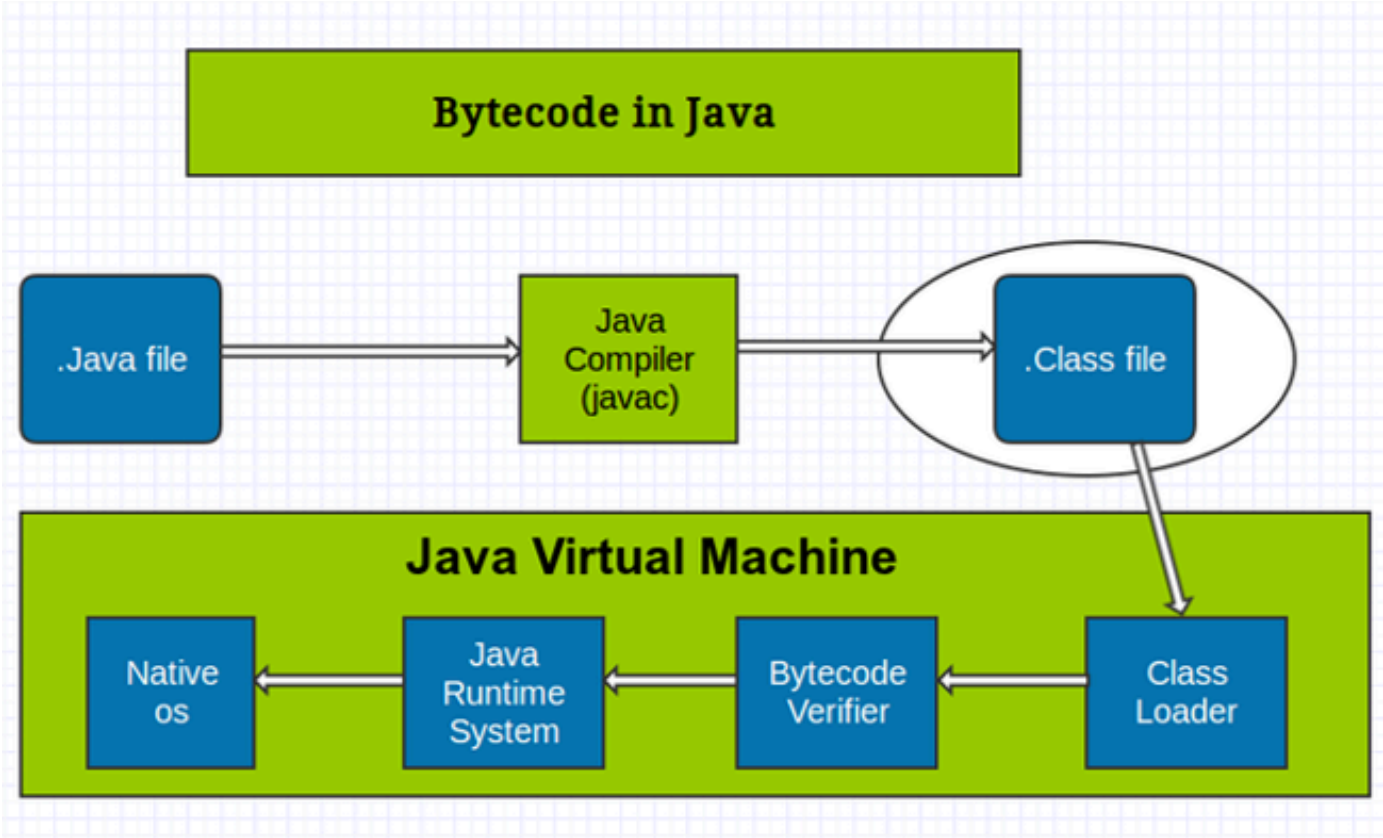
The Big Picture

Compile in Command line:

```
javac helloworld.java
```

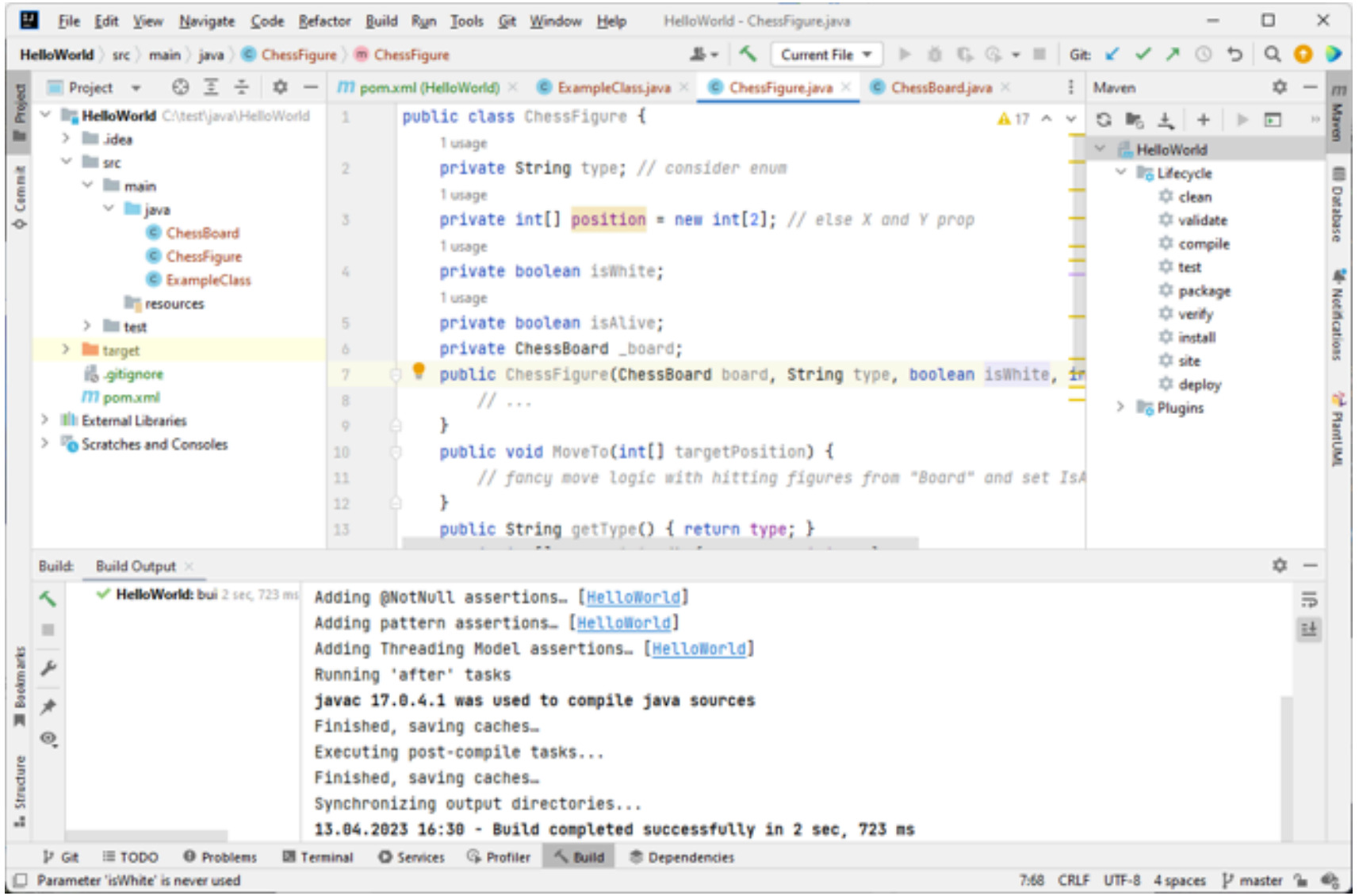
Execute in Command line:

```
java helloworld
```



//java.meritcampus.com/core-java-topics/java-features-java-buzzwords

JetBrains IntelliJ



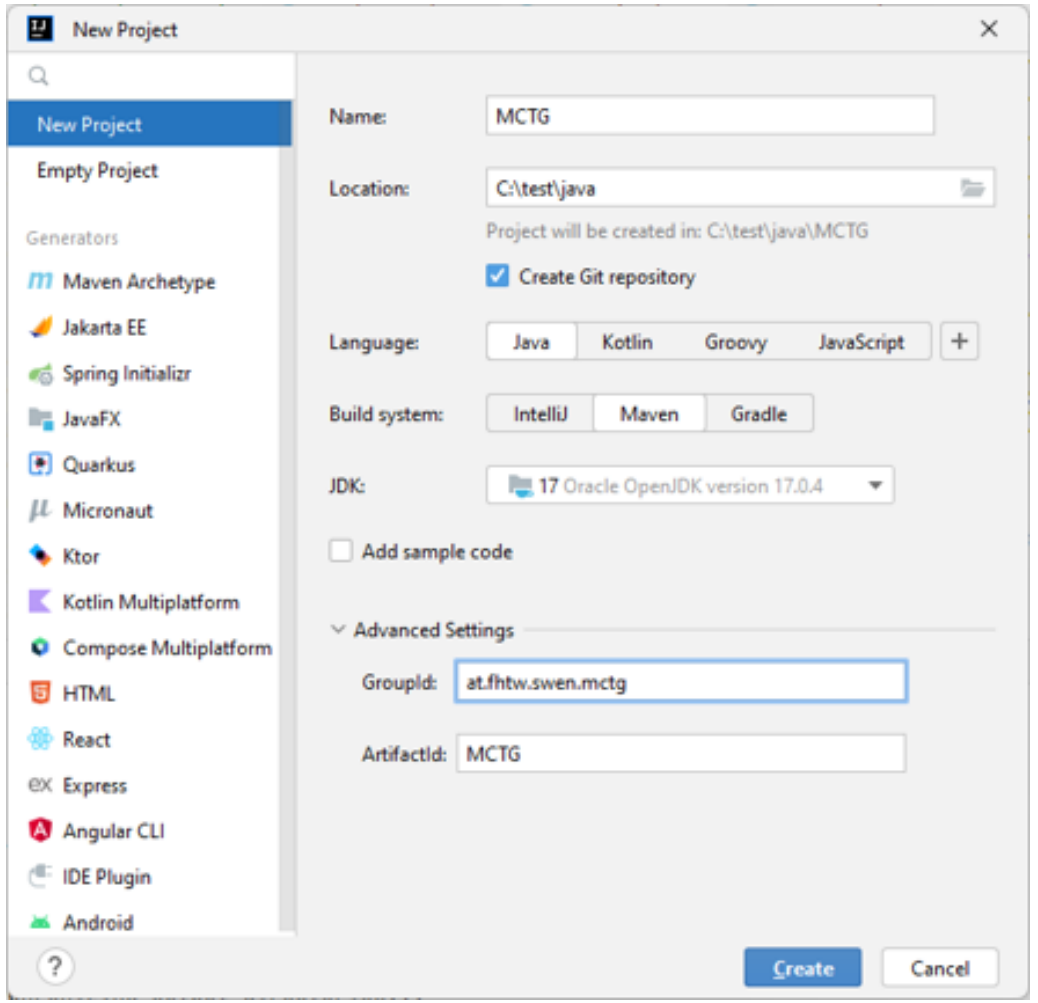
IntelliJ - Create a Project

Project-Templates for many kinds of applications, e.g.:

- Console App
- JavaFX App
- Spring Initializr

Build Systems:

- IntelliJ (integrated)
- **Maven**
- Gradle



IntelliJ Tips & Tricks - Source Editing

Type word in the editor, then hit `Tab`-key.

- `psvm` ... public static void main()
- `sout` ... System.out.println()
- `for`
- `do`
- `if`

...

see also:
https://www.jetbrains.com/help/rider/Templates__Applying_Templates__Creating_Source_Code_Using_Live_Templates.html#selecting

IntelliJ - Default Keyboard Shortcuts

Action	Windows	OS X
SEARCH		
Find usages	Ctrl + Alt + F7	⌘ + F7
Find usages (results)	Ctrl + Alt + Shift + F7	⌘ + Alt + F7
Find / Replace in file	Ctrl + F	⌘ + F / ⌘ + R
Find / Replace in projects	Ctrl + Shift + F	⌘ + Shift + F ⌘ + Shift + R
Find next	F3	F3
FILE NAVIGATION		
Open resource / Navigate to file	Ctrl + Shift + N	⌘ + Shift + O
Open type	Ctrl + N	⌘ + O
Go to symbol	Ctrl + Alt + Shift + N	⌘ + Alt + G
Go to line	Ctrl + G	⌘ + L
Recent files	Ctrl + E	⌘ + E
Tab / File switcher	Ctrl + Tab	⌘ + Shift + [/]
WINDOWS ACTIONS		
Maximize active window	Ctrl + Shift + F12	⌘ + Shift + F12
Next view (editor)	Alt + Left / Right	Ctrl + Left / Right
Quick switch editor	Ctrl + E	⌘ + E
Back	Ctrl + [	⌘ + [
Forward	Ctrl +]	⌘ +]
Show UML popup	Ctrl + Alt + U	⌘ + Alt + U
Activate editor	Ctrl + Tab	Ctrl + Tab
CODE COMPLETION		
Quick fix	Alt + Enter	Alt + Enter
Code completion	Ctrl + Space	Ctrl + Space
Smart code completion	Ctrl + Shift + Space	Ctrl + Shift + Space
Live templates	Ctrl + J	⌘ + J

Action	Windows	OS X
TEXT EDITING ACTIONS		
Move lines	Alt + Shift + Up/Down	Alt + Shift + Up/Down
Delete lines	Ctrl + Y	⌘ + Y
Copy / Duplicate lines	Ctrl + D	⌘ + D
Select identifier	Ctrl + W	Alt + Up
Format code	Ctrl + Alt + L	⌘ + Alt + L
Correct indentation	Ctrl + Alt + I	Ctrl + Alt + I
Structured selection	Ctrl + W	Alt + Up
CODE NAVIGATION		
Find usages / References in workspace	Alt + F7	Alt + F7
Find usages results	Ctrl + Alt + Shift + F7	⌘ + Alt + Shift + F7
Quick outline / File structure	Ctrl + F12	⌘ + F12
Inspect code hierachy	Ctrl + Alt + H	Ctrl + Alt + H
Open / Navigate to declaration	Ctrl + Alt + B	⌘ + Alt + B
Open / Navigate to type hierarchy	Ctrl + H	Ctrl + H
Open / Navigate to member hierarchy	Ctrl + Shift + H	⌘ + Shift + H
REFACTORING		
Refactor this	Ctrl + Alt + Shift + T	⌘ + Alt + Shift + T
Show quick refactoring menu		⌘ + Shift + T
Rename	Ctrl + Alt + R	Shift + F6
Surround with	Ctrl + Alt + T	⌘ + Alt + T
Extract local variable	Ctrl + Alt + V	⌘ + Alt + V
Extract / Assign to field	Ctrl + Alt + F	⌘ + Alt + F
Inline	Ctrl + Alt + N	⌘ + Alt + N
Extract method	Ctrl + Alt + M	⌘ + Alt + M
UNIVERSAL ACCESS		
Quick access / search everywhere	Ctrl + Shift + A	⌘ + Shift + A

Maven

Integration in IntelliJ

SWEN1 | BIF3

What is Maven?

Maven is a tool for building an managing any Java-based project:

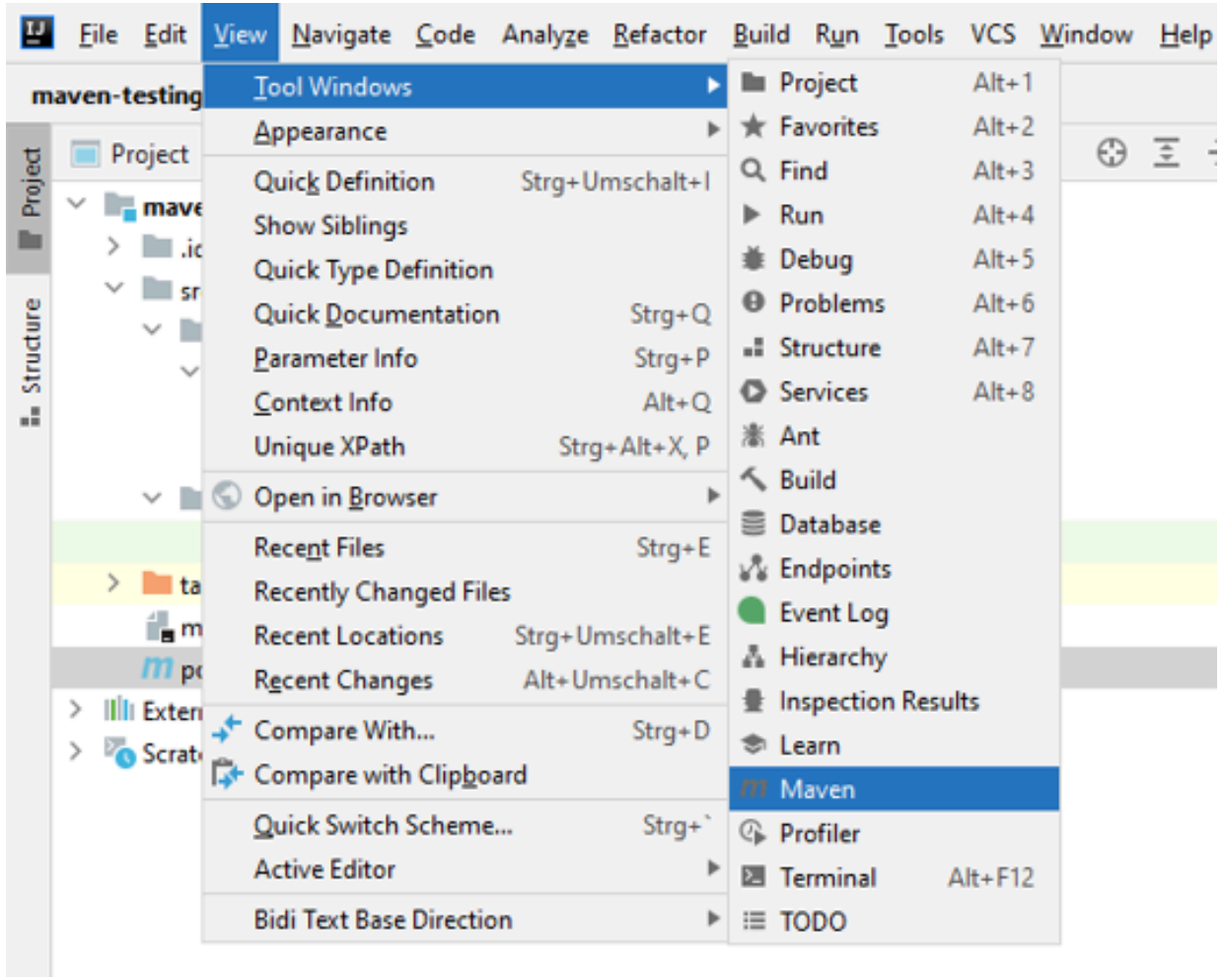
- Make the build process easy
- Provide a uniform build system
- Provide quality project information
- Encourage better development practices

see also: https://maven.apache.org/what-is-maven.html

Maven (mvn) in IntelliJ

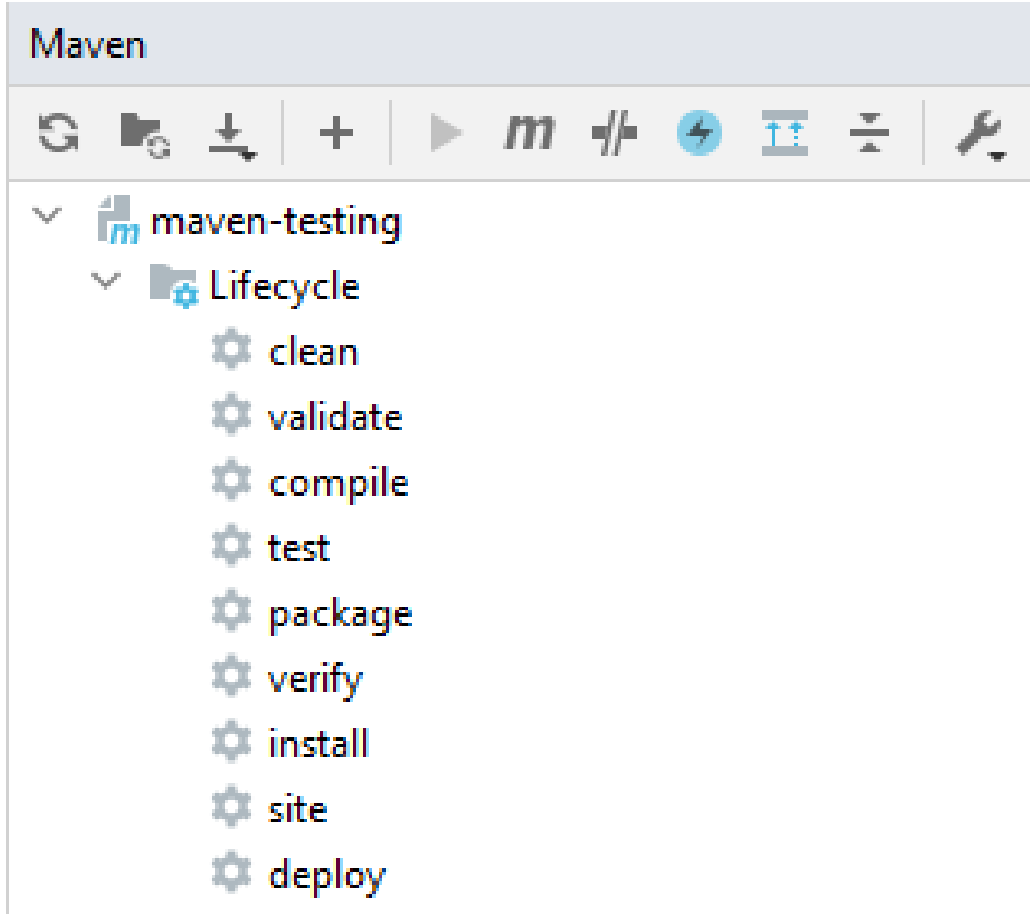
Maven in IntelliJ:

- Creates your project
- Drives your development (assumptions by mvn)



Maven Features - Build

- Build executables from source code
- Packaging of all executables, libs and resources
- Generation of documentation
- Automatic testing
- Automatic analysis of source-code
- Install and deploy on test- or productive-systems
- Reporting



Maven Features - Dependency Management

- dependency management is a core feature
- defining, creating, and maintaining reproducible builds with well-defined classpaths and library versions
- Transitive dependencies included automatically
- Dependency mediation – determines which version of an artifact will be chosen when multiple versions are encountered
- Dependency scope

Maven pom.xml

pom.xml:

- project metadata
- centralizing dependency information
- configuring plugins

see sample on the right side -->

```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>at.fhtw</groupId>
    <artifactId>shapelibdemo</artifactId>
    <version>1.0-SNAPSHOT</version>

    <properties>
        <java.version>21</java.version>
        <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
    </properties>

    <build>
        <plugins>
            <plugin>
                <groupId>org.apache.maven.plugins</groupId>
                <artifactId>maven-compiler-plugin</artifactId>
                <version>3.8.1</version>
                <configuration>
                    <source>${java.version}</source>
                    <target>${java.version}</target>
                    <release>${java.version}</release>
                </configuration>
            </plugin>
        </plugins>
    </build>

    <dependencies>
        <dependency>
            <groupId>org.projectlombok</groupId>
            <artifactId>lombok</artifactId>
            <version>1.18.28</version>
        </dependency>
    </dependencies>
</project>
```