



NARZĘDZIA I TECHNOLOGIE WSPOMAGAJĄCE PROGRAMOWANIE

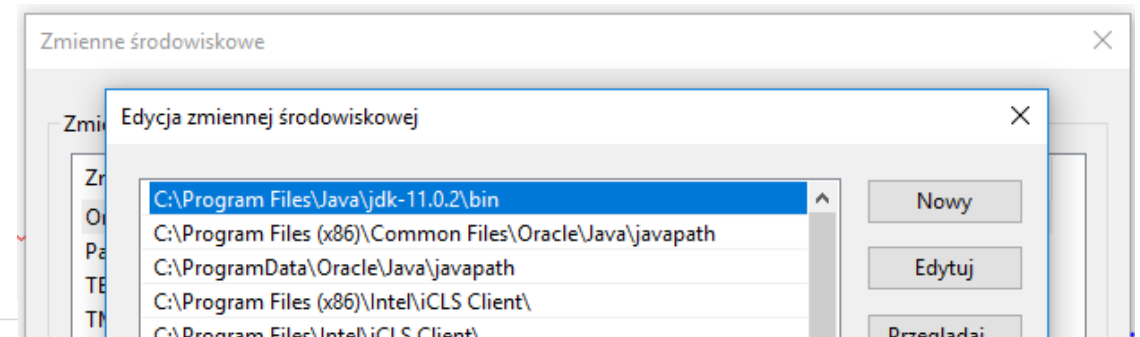
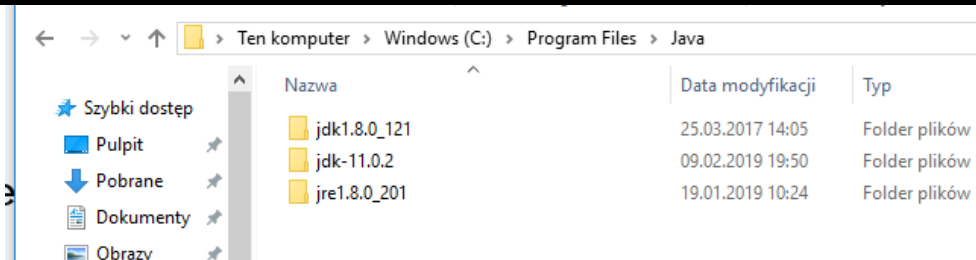
Edycja 2019

INTRODUCTORY EXERCISE

- Install newest Java (JDK 11)
- Set environment variable: JAVA_HOME in your operating system
- Using command line or terminal, type: `java -version`

```
C:\Users\Zbyszko>java -version
java version "11.0.2" 2019-01-15 LTS
Java(TM) SE Runtime Environment 18.9 (build 11.0.2+9-LTS)
Java HotSpot(TM) 64-Bit Server VM 18.9 (build 11.0.2+9-LTS, mixed mode)

C:\Users\Zbyszko>
```

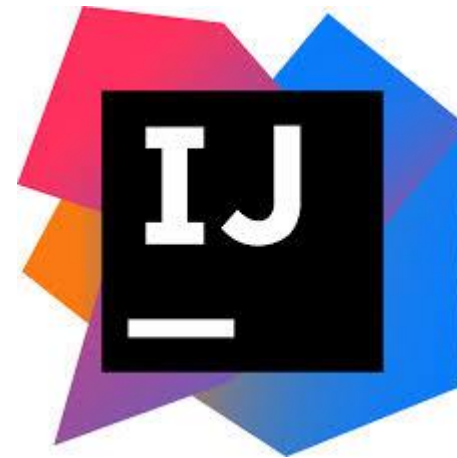


INSTALL YOUR IDE

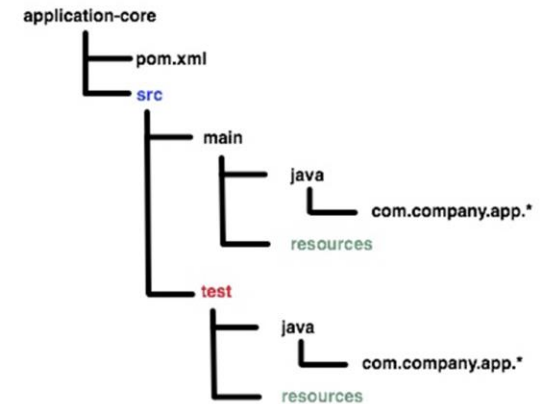
If you don't have Integrated Development Environment (IDE), it is a good time to install!

Recommendation: IntelliJ

Other: Eclipse or Netbeans



START YOUR DEVELOPMENT WITH MAVEN!



- Advanced build tool to support the developer at the whole process of a software project
 - Typical developer tasks: compile, tests, pack into JAR, run
- Automates creation of the initial folder structure for the Java application
- Management of dependencies (no need to manual download of external libraries)
- Repository: dependencies can be loaded from the local file system, from the Internet or public repositories
- Management of releases
- Simple usage: supply project templates (archetypes)
- Convention over configuration: avoid as much configuration as possible, by choosing real world default values
- Extensible (plugins)
- Empower pom.xml

SCAFFOLDING A PROJECT WITH MAVEN

- Maven supports project scaffolding, based on project templates called archetype
- Maven comes with number of „ready-to-go” archetypes
- Extremely speeds up the development preparation
- Type: *mvn archetype:generate*
- A pom should minimum have the following information:
 - `modelVersion`
 - `groupId`
 - `artifactId`
 - `version`

```
<project>  
  <modelVersion>4.0.0</modelVersion>  
  <groupId>com.mycompany.app</groupId>  
  <artifactId>my-app</artifactId>  
  <version>1</version>  
</project>
```

```
mvn archetype:generate -DarchetypeGroupId=org.apache.maven.archetypes -DarchetypeArtifactId=maven-archetype-quickstart -DarchetypeVersion=1.4
```

MAVEN LIFECYCLES, PHASES, GOALS

- 3 built-in build lifecycles:
 - **default** - project deployment
 - **clean** - project cleaning
 - **site** - creation of project's documentation
- Each lifecycles is defined by a different list of phases, wherein a phase represents a stage in the lifecycle.
- For example, the default lifecycle comprises of the following phases:
 - **validate** - check project correctness
 - **compile**
 - **test** - test the compiled source code using a unit testing
 - **package** - take the compiled code and package into distributable format (JAR/WAR)
 - **verify** - run any checks on results of integration tests to ensure quality criteria are met
 - **install** - install the package into the local repository
 - **deploy** - copies the final package to the remote repository for sharing with other developers and projects.
- Phase is made up of plugin goals

MAVEN EXERCISE

- Download maven package
- Set environment variable: `MAVEN_HOME`
- Using command line or terminal, check your maven version: `mvn -v` or `mvn --version`
- Generate project with archetype „quickstart” (default)
- Import project into IDE
- Review pom.xml
- Build package using Maven (using command line)
- Add project lombok dependency
- Create simple Java class with lombok annotation
- Build again using Maven

LOMBOK PROJECT

- Reduce boilerplate code
- Generation of constructors, getters/setters, equal and hashCode, toString methods via annotations:
 - @EqualsAndHashCode
 - @ToString
 - @AllArgsConstructor
 - @Getter
 - @Setter
- Automatic beans creation
 - @Data
- Builder design pattern
 - @Builder
- More: <https://projectlombok.org>

```
<dependency>  
  <groupId>org.projectlombok</groupId>  
  <artifactId>lombok</artifactId>  
  <version>1.16.20</version>  
  <scope>provided</scope>  
</dependency>
```


ANOTHER MAVEN'S KEY FEATURES THAT ARE WORTH TO BE KNOWN

- Wrappers
- Multi module projects (aggregator)
- Profile
- Own plugins
- Adding goals to life cycle phases
- Local/Remote/Central repositories
- Resolving conflicts using the dependency tree
 - mvn dependency:tree
- Read more: <http://maven.apache.org/index.html>