



Swing, SOA και Dependency Management με Apache Maven

Αθ. Ανδρούτσος



Dependencies

- Στο Swing Project του προηγούμενου κεφαλαίου για να συνδεθούμε με τη ΒΔ εισάγαμε στο Build Path ένα Connector/J αρχείο που είναι η υλοποίηση του JDBC από την MySQL
- Αυτό είναι ένα **dependency**, δηλαδή ένα εξωτερικό (third party) αρχείο σε μορφή .JAR (Java ARchive)
- **Όλα τα προγράμματα έχουν εξαρτήσεις**, δηλαδή χρειάζονται άλλα προγράμματα και βιβλιοθήκες που να προσθέτουν λειτουργικότητα
- Τυπικά, οι βιβλιοθήκες αυτές έρχονται σε μορφή jar (Java **A**rchive) δηλαδή zip μορφή compiled αρχείων java (class files) αλλά και audio & image files



Dependencies Management

Full Stack, SOA & Maven

- Η **διαχείριση εξαρτήσεων** είναι ένα θέμα που πρέπει να μας απασχολήσει. Είδαμε πως για να βρούμε τον Connector/J έπρεπε είτε να κατεβάσουμε το Workbench ή να ψάξουμε ανεξάρτητα και να κατεβάσουμε το αρχείο του Connector και να το εισάγουμε στο build path
- Συνήθως τα προγράμματα έχουν πολλά dependencies ενώ πολλές φορές τα dependencies έχουν και τα ίδια άλλα dependencies (**transitive dependencies**) που εμείς δεν γνωρίζουμε
- Επομένως, θα πρέπει να αναζητήσουμε **μία λύση για την εύκολη διαχείριση των dependencies**



Maven (1)

- Το **Apache Maven** είναι ένα Framework που μας διευκολύνει στη διαχείριση των Dependencies
- Όπως αναφέραμε, τα περισσότερα προγράμματα Java χρησιμοποιούν βιβλιοθήκες κλάσεων τρίτων σε μορφή .jar
- Αυτά τα .jar ονομάζονται dependencies και πρέπει να ψάχνουμε να τα βρίσκουμε και να τα κάνουμε add στο Project μας



Maven (2)

- Κάτι τέτοιο δεν είναι και πολύ παραγωγικό όταν το κάνουμε μόνοι μας
- Το Maven κάνει αυτόματα την παραπάνω διαδικασία με τη χρήση ενός αρχείου xml, του **POM.xml** στο οποίο ορίζουμε τα dependencies που θέλουμε
- Επιπλέον το **Maven δημιουργεί αυτόματα μία δομή φακέλων** ώστε όλα τα μέλη της ομάδας να δουλεύουν με την ίδια οργάνωση φακέλων
- Το Maven website είναι το <https://maven.apache.org/>



App Development stages

Full Stack, SOA & Maven

- Πριν δούμε περισσότερες λεπτομέρειες για το Maven, να κάνουμε μία ανασκόπηση των βασικών σταδίων στην ανάπτυξη και εκτέλεση μίας εφαρμογής
- Τα βασικά στάδια είναι: 1) *Codebase* (Συγγραφή), 2) *Build* (Μεταγλώττιση), 3) Έλεγχος (test), 4) *Release* (.jar αρχείο) 5) *Deploy* (Install software & Run σε παραγωγικό περιβάλλον)
- Αυτή η σειρά βημάτων ονομάζεται **pipeline** γιατί κάθε επόμενη φάση λαμβάνει input από την προηγούμενη



- Όσο αφορά την επιχειρησιακή οργάνωση των προηγούμενων φάσεων μπορούμε να τις κατηγοριοποιήσουμε σε δύο κατηγορίες:
 1. Development, που είναι οι φάσεις μέχρι το Deploy, και
 2. Operation, που είναι η φάση της παραγωγικής εκτέλεσης της εφαρμογής



Maven Pipeline

Full Stack, SOA & Maven

- Το Maven αυτοματοποιεί όλη σχεδόν τη φάση του Development, δηλαδή
 - 1. Κατεβάζει τα dependencies που έχουμε ορίσει στο POM.xml
 - Κάνει build, δηλαδή μεταγλωττίζει και παράγει .class αρχεία
 - Κάνει Unit Testing με test frameworks όπως το JUnit και παράγει reports με το surefire plugin
 - Παράγει το τελικό release (αφού περάσει τα tests) και το κάνει package σε μορφή .jar ή .war



Deploy

- Τέλος μπορεί να γίνει το deployment είτε σε περιβάλλον που έχουμε εγκαταστήσει με το χέρι τα προαπαιτούμενα (π.χ. Apache Web Server) ή συνήθως σε περιβάλλον εικονικών μηχανών (VMs) και πλέον σε περιβάλλον containerized όπως Docker
- Αν έχουμε περισσότερα docker containers που αλληλεπιδρούν μπορούμε να οργανώσουμε την επικοινωνία με το Kubernetes



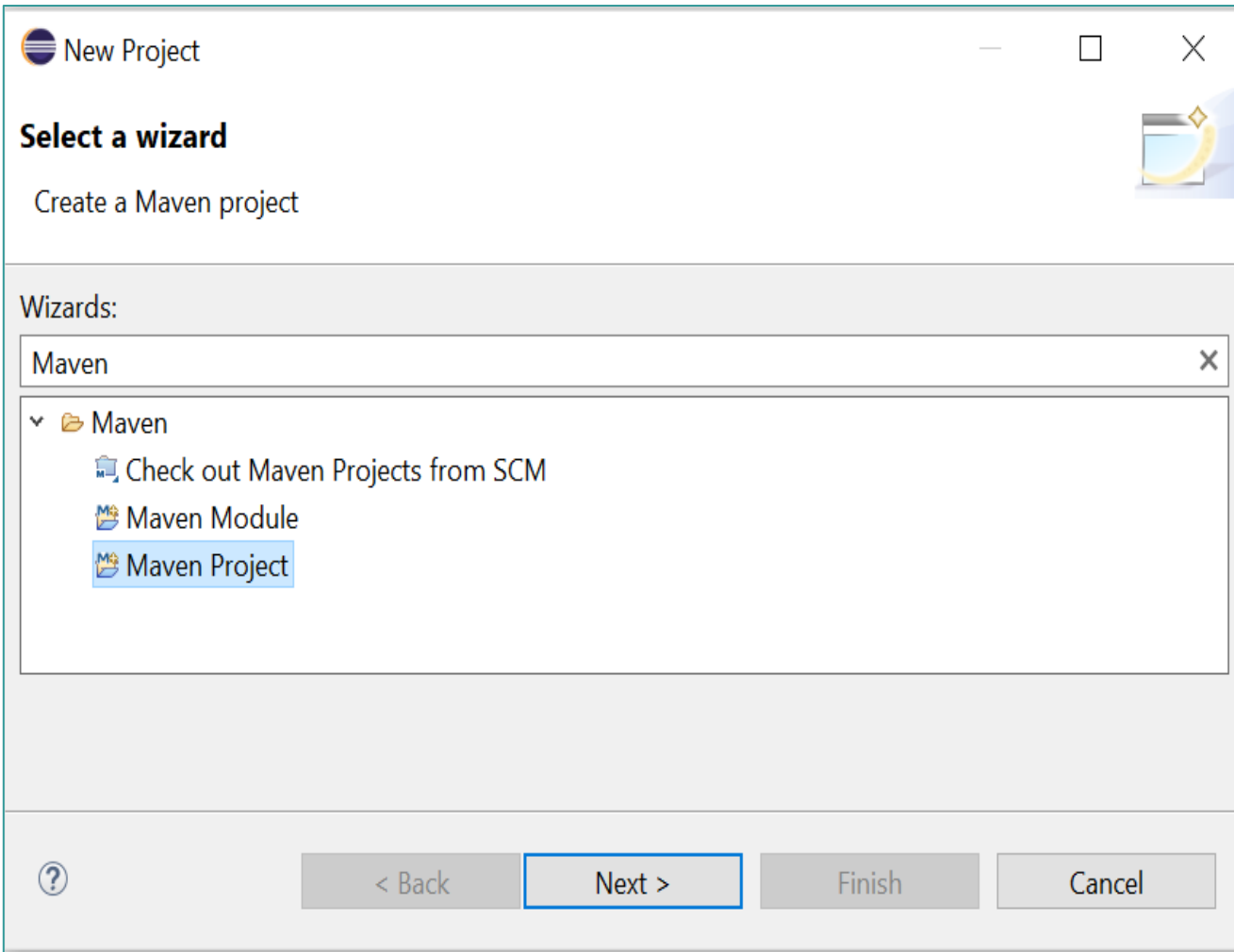
Maven as Build Tool

- Το Maven είναι πρωτίστως ένα build tool που αυτοματοποιεί όλο το pipeline build, test και release
- Όσο αφορά τη φάση του build παρέχει τη δυνατότητα για dependency management με το αρχείο POM.xml
- Όσο αφορά το Unit Test σε συνδυασμό με το JUnit (ή αντίστοιχα frameworks όπως TestNG, κ.α.) και σε συνδυασμό με το surefire plugin παρέχει ένα αυτοματοποιημένο test περιβάλλον και εξάγει reports



Eclipse / New Maven Project

Full Stack, SOA & Maven



- Θα δημιουργήσουμε ένα νέο Maven Project στο Eclipse με **File / New Project**
- Αναζητούμε για **Maven** και επιλέγουμε Maven Project



quickstart archetype (1)

Full Stack, SOA & Maven

- Για Swing Projects που δεν έχουν κάποια πιο ειδική δομή φακέλων μπορούμε είτε να κάνουμε skip archetype (το Maven μας δίνει τη default δομή φακέλων) ή να μην επιλέξουμε skip archetype και να επιλέξουμε στη συνέχεια το quickstart Archetype



quickstart archetype (2)

Full Stack, SOA & Maven

New Maven Project

New Maven project

Select an Archetype

Catalog: All Catalogs Configure...

Filter: org.apache.maven

Group Id	Artifact Id	Version
org.apache.maven.archetypes	maven-archetype-quickstart	1.0-alpha-2
org.apache.maven.archetypes	maven-archetype-quickstart	1.0-alpha-3
org.apache.maven.archetypes	maven-archetype-quickstart	1.0-alpha-4
org.apache.maven.archetypes	maven-archetype-quickstart	1.1
org.apache.maven.archetypes	maven-archetype-quickstart	1.3
org.apache.maven.archetypes	maven-archetype-quickstart	1.4
org.apache.maven.archetypes	maven-archetype-simple	1.3

An archetype which contains a sample Maven project.
<https://repo1.maven.org/maven2>

☐ Show the last version of Archetype only ☐ Include snapshot archetypes Add Archetype...

Advanced

? < Back Next > Finish Cancel

- Με skip archetype ή με quickstart archetype θα έχουμε τους βασικούς φακέλους: **src/main/java**, **src/test/java**



GAV -- Group Artifact Version

Full Stack, SOA & Maven

New Maven Project

New Maven project

Specify Archetype parameters

Group Id: gr.aueb.cf

Artifact Id: school-app-winter

Version: 0.0.1-SNAPSHOT

Package: gr.aueb.cf.schoolapp

☒ run archetype generation interactively

Properties available from archetype:

Name	Value

Advanced

< Back Next > Finish Cancel

- **Group Id:** Λογικό group που μπορεί να περιλαμβάνει πολλά artifact. Naming convention: reverse domain.
- **Artifact Id:** Το όνομα με το οποίο είναι γνωστό το Project (σύμβαση με μικρά γράμματα και παύλες) – αυτό θα είναι και το όνομα του .jar (.war) που θα εξαχθεί στο τελικό release
- **Version:** SNAPSHOT, qualifier που σημαίνει ότι το version αυτό είναι ακόμα unreleased (όταν θα είναι έτοιμο το version για release μπορεί να ονομαστεί 0.0.1)



Software Artifacts

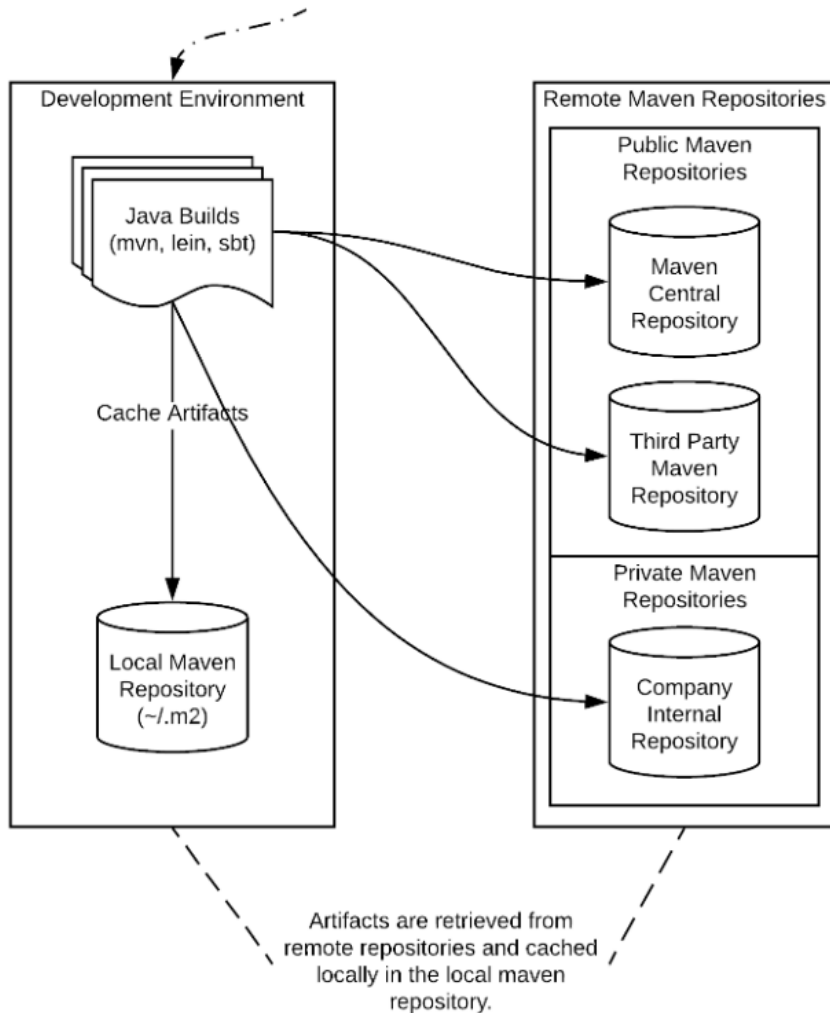
Full Stack, SOA & Maven

- Τα Maven artifacts είναι κάθε τύπος αρχείου που χρησιμοποιείται στο software development process
- Τα πιο γνωστά artifacts είναι τα αρχεία .jar
- Τα maven artifacts αποθηκεύονται σε repositories από όπου τα κατεβάζουμε και τα εισάγουμε στο project μας



Repositories (1)

A Maven Development Environment can be on a local developer machine, or part of a continuous integration and deployment environment



- Το Maven μπορεί να είναι εγκατεστημένο στο τοπικό μας μηχάνημα ή να είναι μέρος ενός CI/CD περιβάλλοντος όπως το Jenkins
- Σε κάθε περίπτωση υπάρχει ένα τοπικό repository το `~/.m2`, που λειτουργεί ως *cache*, οτιδήποτε δηλαδή κατεβάζουμε από remote repositories, αποθηκεύεται τοπικά ώστε αν το χρειαστούμε ξανά σε άλλο project να μην το ξανακατεβάζουμε από το Maven Central Repository



Repositories (2)

- Υπάρχουν επίσης απομακρυσμένα repositories, όπως repositories κάποιας εταιρείας ή οργανισμού, ή third-party repositories όπως το <https://mvnrepository.com/> ή και το central repository του Maven <https://repo1.maven.org/maven2/> ή <https://repo.maven.apache.org/maven2/>
- Τα τοπικά repositories λειτουργούν ως caches αλλά και ως deployment destinations για τα δικά μας (ή του οργανισμού) τα artifacts
- Κατά τα άλλα όλα τα δημόσια (software) artifacts είναι διαθέσιμα σε ένα κεντρικό repository του Maven



- Group / Artifact / Version (GAV) καθώς και το packaging (war / jar) είναι οι βασικές ιδιότητες του Maven για να διαχειρίζεται dependencies
- Οι τρεις GAV συντεταγμένες διαμορφώνουν ένα μοναδικό FQN (Fully Qualified Artifact Name) στο maven repository
- Defaults
 - Packaging type – default value: **jar**
 - Maven Repository:

repo.maven.apache.org/maven2

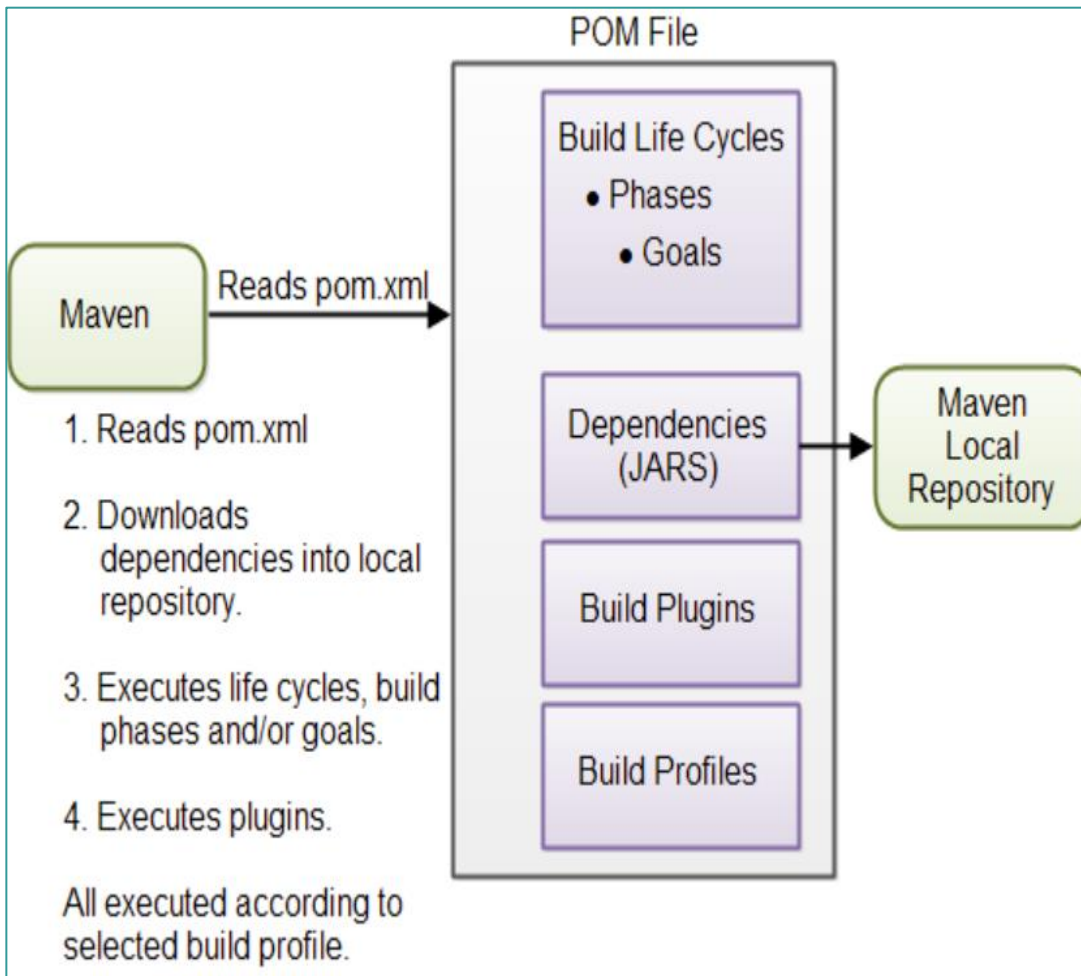


POM.xml (1)

- Το POM.xml (Project Object Model) είναι ένα XML representation των project resources όπως source code, test code, dependencies (external JARs used) κλπ.
- Το POM.xml περιέχει references σε όλα αυτά τα resources
- Το POM file πρέπει να βρίσκεται στο root directory του project στο οποίο ανήκει



POM.xml (2)



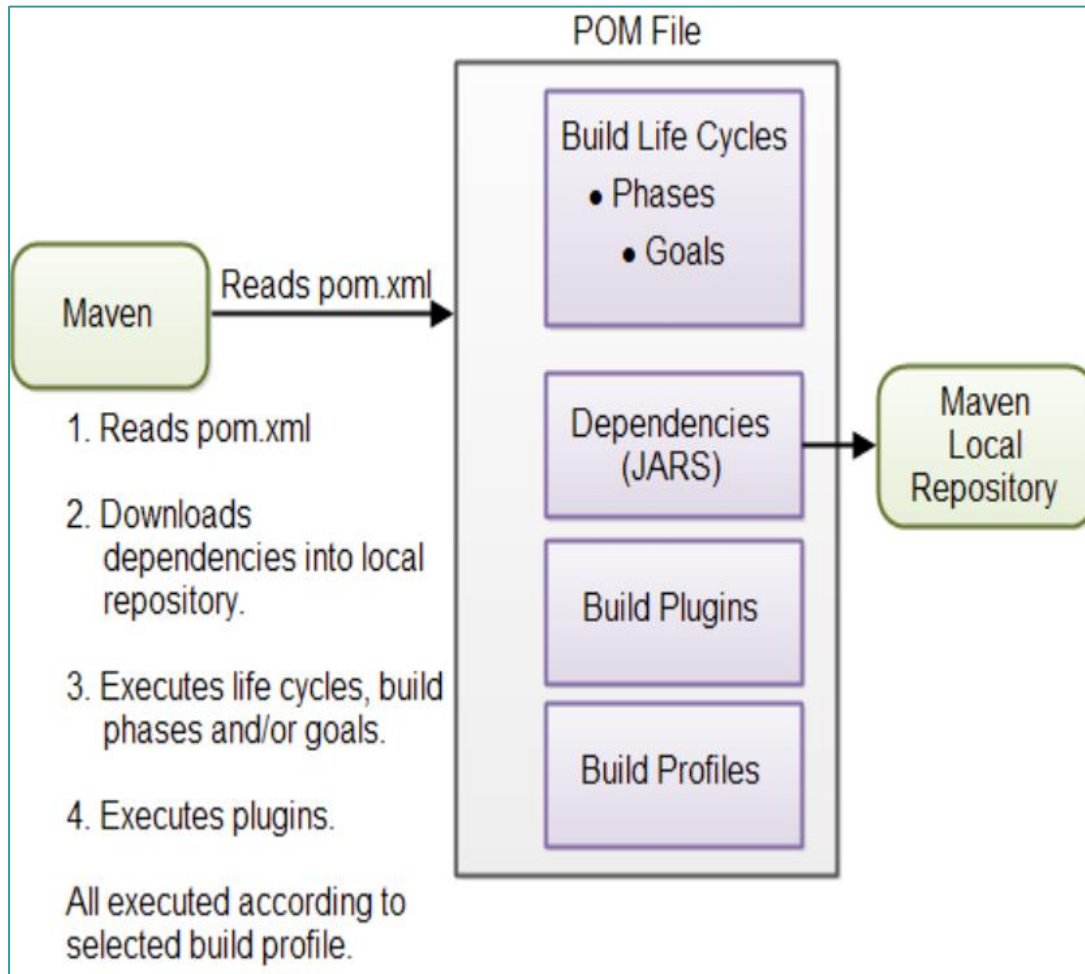
Όταν δίνουμε ένα Maven command, το command εκτελείται στα resources που περιγράφονται μέσα στο POM.xml

Το build process χωρίζεται σε phases και goals (κάθε phase αποτελείται από μία ακολουθία goals) ενώ και τα phases ομαδοποιούνται σε life cycles

Κάθε Maven command είναι το όνομα ενός life cycle, μιας phase ή ενός goal. Αν εκτελέσουμε ένα life cycle, εκτελούνται όλα τα phases του life cycle. Αν εκτελέσουμε ένα phase, εκτελούνται και όλα τα phases πριν από αυτό



POM.xml (3)



- Ένα από τα πρώτα goals που εκτελεί το Maven είναι να ελέγξει και να φέρει τα dependencies που ορίζονται στο POM.
- Τα Build plugins είναι extra goals που εκτελούνται σε κάποιο build phase. Το Maven έχει κάποια standard plugins (π.χ. Surefire)
- Τα build profiles χρησιμοποιούνται όταν θέλουμε να μπορούμε στο ίδιο project να έχουμε διαφορετικά builds



Declaring Dependencies

- Δηλώνουμε dependencies στο αρχείο POM.xml εισάγοντας ένα entry για κάθε dependency

```
<!-- https://mvnrepository.com/artifact/junit/junit -->
<dependency>
  <groupId>junit</groupId>
  <artifactId>junit</artifactId>
  <version>4.13.2</version>
  <scope>test</scope>
</dependency>
```



Minimal POM

```
1 <?xml version="1.0" encoding="UTF-8"?>
2
3 <project xmlns="http://maven.apache.org/POM/4.0.0"
4   xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
5     http://maven.apache.org/maven-v4_0_0.xsd"
6     <modelVersion>4.0.0</modelVersion>
7     <groupId>gr.aueb.cf</groupId>
8     <artifactId>school-app-winter22</artifactId>
9     <version>0.0.1-SNAPSHOT</version>
10
```

- Το *artifactId* είναι το όνομα του project. Είναι επίσης μέρος του ονόματος του .jar file ενώ δημιουργείται και sub-directory του *groupId*
- Το *version* περιέχει το version number του project όταν έχουμε πολλά releases, οπότε τότε είναι χρήσιμο να δώσουμε version στα builds. Είναι επίσης μέρος του ονόματος του .jar file ενώ δημιουργείται και sub-directory του *groupId/artifact*

- Το *modelVersion* είναι το version του POM (το version 4.0.0 κάνει match με το Maven version 2 και 3)
- Το *groupId* είναι ένα unique id του οργανισμού και συνήθως είναι ίδιο με το root package name του project. Στο maven repository το project αποθηκεύεται ως *~m2/gr/aueb/cf* δηλαδή τα tokens μεταξύ των dots (.) αντιστοιχούν σε directories



Executing commands (1)

- Η εκτέλεση εντολών γίνεται με την εντολή **mvn** και ως argument το όνομα ενός life cycle, ενός build phase ή ενός goal. Για παράδειγμα:
 - **mvn install** Εκτελεί το build phase που ονομάζεται *install* (που είναι μέρος του *default* life cycle) που κάνει build το project και κάνει copy το .jar στο τοπικό maven repository. Για την ακρίβεια εκτελούνται όλα τα phases πριν το install και τέλος το install build phase.
 - **mvn clean install** Μπορούμε να εκτελέσουμε και περισσότερα life cycles σε ένα command. Εδώ εκτελείται πρώτα το clean life cycle που διαγράφει τα .class αρχεία από τον φάκελο target



Executing commands (2)

Full Stack, SOA & Maven

- Μπορούμε επίσης να εκτελέσουμε ένα Maven goal περνώντας ως argument το build phase και το goal και συνενώνοντας τα με :
- Π.χ. `mvn dependency:copy-dependencies`



Standard directory structure

Full Stack, SOA & Maven









```
- src
  - main
    - java
    - resources
    - webapp
  - test
    - java
    - resources
- target
```

- Το mvn έχει ένα standard directory structure. Αν ακολουθήσουμε αυτό το directory structure δεν χρειάζεται να προσδιορίσουμε τα directories του source και test code στο POM.xml
- Το **src** είναι το root directory του source και test code. Το **main** directory περιέχει τον source code της εφαρμογής αυτής καθαυτής, όχι του test code. Το **test** directory περιέχει τον test κώδικα.
- Τα **Java** directories κάτω από το main και το test περιέχουν Java κώδικα για την εφαρμογή την ίδια (main/java) ή για τα tests (test/java).
- Το **resources** directory περιέχει άλλα αρχεία που χρειάζονται στο project, όπως property files για internationalization, αρχεία μέσων (εικόνες, κλπ.) κ.α.
- Το **webapp** directory περιέχει την Java Web Application αν το project είναι web application. Είναι το root directory της web εφαρμογής. Το webapp περιέχει το WEB-INF directory (περιέχει config xml files του Apache Web Server όπως web.xml, και άλλων application servers, π.χ. weblogic.xml)
- Το **target** directory δημιουργείται από το Maven κατά το build. Περιέχει όλες τις compiled κλάσεις, JAR files κλπ. που παράγονται από το Maven.



School-app

Full Stack, SOA & Maven

- ▼  school-app-winter22
 - >  src/main/java
 - >  src/test/java
 - >  JRE System Library [JavaSE-1.7]
 - >  Maven Dependencies
 - >  src
 -  target
 -  pom.xml

- Build directory: **target**
- Source directory: **src/main/java**
- Test directory: **src/test/java**
- POM File: **pom.xml**
- JRE System Library: Δείχνει στον compiler 1.7 ()

- Θα πρέπει να αλλάξουμε τον compiler από 1.7 σε 11 ή 17 ή γενικά στο JDK που έχουμε εγκατεστημένο



JDK version - properties

```
1 <?xml version="1.0" encoding="UTF-8"?>
2
3 <project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4     xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
5     <modelVersion>4.0.0</modelVersion>
6
7     <groupId>gr.aueb.cf</groupId>
8     <artifactId>school-app5</artifactId>
9     <version>1.0-SNAPSHOT</version>
10
11     <name>school-app5</name>
12     <!-- FIXME change it to the project's website -->
13     <url>http://www.example.com</url>
14
15     <properties>
16         <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
17         <maven.compiler.source>17</maven.compiler.source>
18         <maven.compiler.target>17</maven.compiler.target>
19     </properties>
```

- Μέσα στο node <properties> ορίζουμε τα source & target versions του java compiler. Κάνουμε Save & Update το project (στο Eclipse). Ή Maven / Reload Project (στο IntelliJ)



JDK version - plugin

```
<build>
  <pluginManagement><!-- lock down plugins versions to avoid
    <plugins>

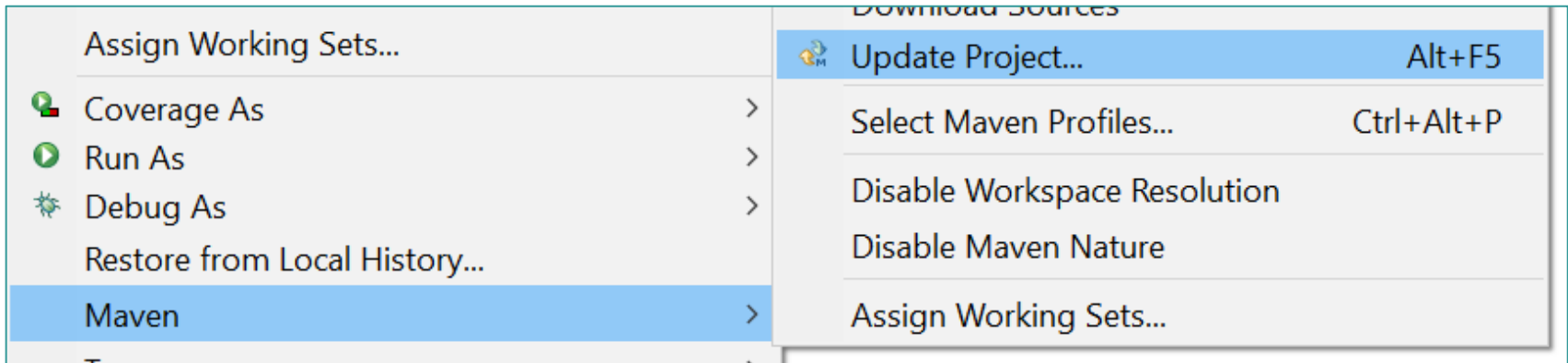
    <!-- https://mvnrepository.com/artifact/org.apache.maven.plugins/maven-compiler-plugin
    <plugin>
      <groupId>org.apache.maven.plugins</groupId>
      <artifactId>maven-compiler-plugin</artifactId>
      <version>3.10.1</version>
      <configuration>
        <source>11</source>
        <target>11</target>
      </configuration>
    </plugin>
```

- Εναλλακτικά μπορούμε να ορίσουμε τον compiler με το Maven plugin. Το maven-compiler-plugin έχει προτεραιότητα έναντι του <properties>. Σε κάθε περίπτωση ο compiler θα αναζητηθεί στο JAVA_HOME environmental variable

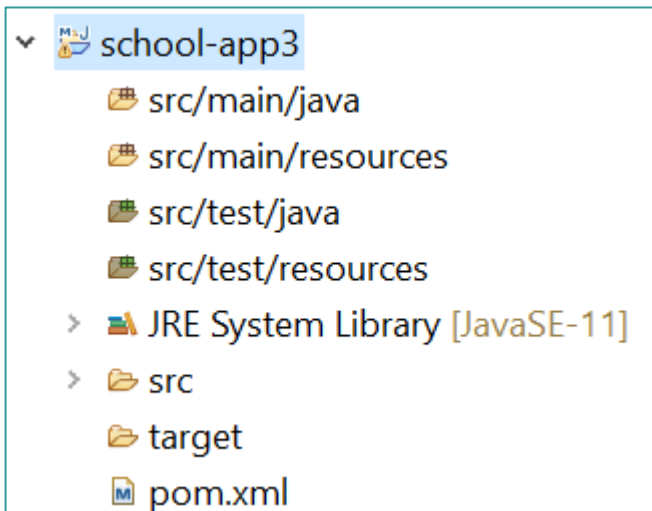


Save & Update Project (Eclipse)

Full Stack, SOA & Maven



- Save το POM.xml και μετά Maven/Update Project και αλλάζει το version του compiler





Dependencies

The screenshot shows the Maven Repository website with the search query 'mysql connector J'. The search results show 9540 results, sorted by relevance. The top result is 'MySQL Connector/J' with 727 usages. The description states it is a JDBC Type 4 driver, which means it is a pure Java implementation of the MySQL protocol and does not rely on the MySQL client libraries. It supports auto-registration with the Driver Manager, standardized validity checks, categorized SQLExceptions, support for large update counts, support for local and offset date-time variants from the java.time package, support for JDBC-4.x XML processing, support for per connection client information and support for the NCHAR, NVARCHAR ...

Repository

Repository	Count
Central	6.9k
Sonatype	499
Mulesoft	445
Mulesoft Releases	443
BeDataDriven	320
JCenter	269
JBoss Releases	157
WSO2 Releases	151

Group

Found 9540 results

Sort: **relevance** | popular | newest

1. **MySQL Connector/J** 727 usages

[com.mysql » mysql-connector-j](#)

MySQL Connector/J is a JDBC Type 4 driver, which means that it is pure Java implementation of the MySQL protocol and does not rely on the MySQL client libraries. This driver supports auto-registration with the Driver Manager, standardized validity checks, categorized SQLExceptions, support for large update counts, support for local and offset date-time variants from the java.time package, support for JDBC-4.x XML processing, support for per connection client information and support for the NCHAR, NVARCHAR ...

Last Release on Jul 2, 2024

- Αναζητούμε τα Dependencies του MySQL Connector/J στο mvnrepository.com



MySQL Connector/J

Full Stack, SOA & Maven

mvnrepository.com/artifact/com.mysql/mysql-connector-j/8.3.0

5.3.... Μεταγλωττιστές Electronic library. D... (All Streams) - One... OIKOPEDA Android Hibernate Java aocAI - OneDrive

Search for groups, artifacts, categories Search

Home » com.mysql » mysql-connector-j » 8.3.0

MySQL Connector/J » 8.3.0

MySQL Connector/J is a JDBC Type 4 driver, which means that it is pure Java implementation of the MySQL protocol and does not rely on the MySQL client libraries. This driver supports auto-registration with the Driver Manager, standardized validity checks, categorized SQLExceptions, support for large update counts, support for local and offset date-time variants from the java.time package, support for JDBC-4.x XML processing, support for per connection client information and support for the NCHAR, NVARCHAR ...

Categories	JDBC Drivers
Tags	database sql jdbc driver connector rdbms mysql connection
Organization	Oracle Corporation
HomePage	http://dev.mysql.com/doc/connector-j/en/
Date	Jan 16, 2024
Files	pom (3 KB) jar (2.4 MB) View All
Repositories	Central
Ranking	#684 in MvnRepository (See Top Artifacts) #9 in JDBC Drivers
Used By	727 artifacts

Note: There is a new version for this artifact

New Version	9.0.0
-------------	-------

Maven Gradle Gradle (Short) Gradle (Kotlin) SBT Ivy Grape Leiningen Buildr

```
<!-- https://mvnrepository.com/artifact/com.mysql/mysql-connector-j -->
<dependency>
  <groupId>com.mysql</groupId>
  <artifactId>mysql-connector-j</artifactId>
  <version>8.3.0</version>
</dependency>
```




POM.xml

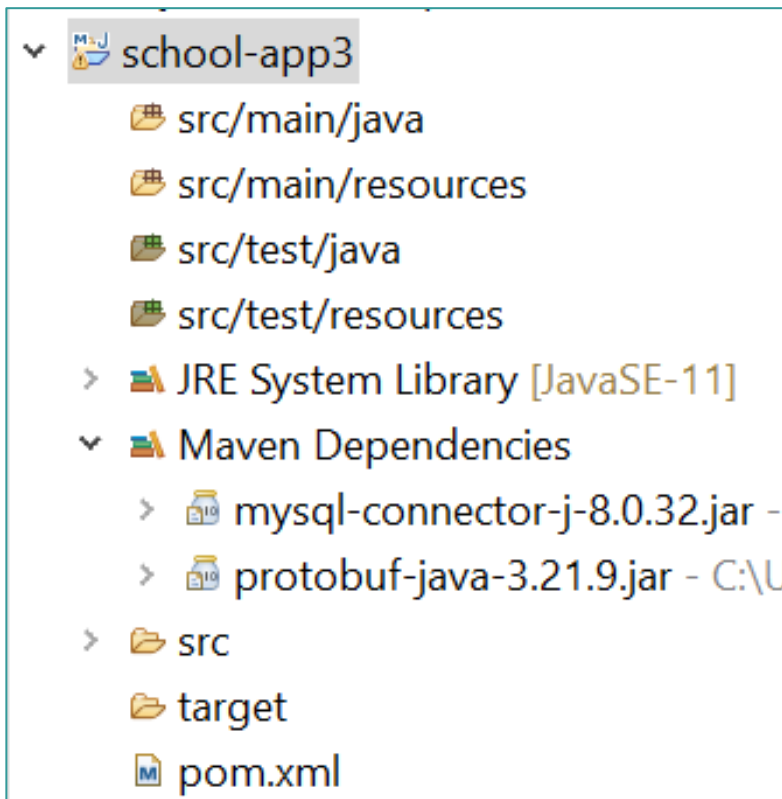
```
7 <groupId>gr.aueb.cf</groupId>
8 <artifactId>school6-test</artifactId>
9 <version>1.0-SNAPSHOT</version>
10
11 <name>school6-test</name>
12 <!-- FIXME change it to the project's website -->
13 <url>http://www.example.com</url>
14
15 <properties>
16   <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
17   <maven.compiler.source>17</maven.compiler.source>
18   <maven.compiler.target>17</maven.compiler.target>
19 </properties>
20
21 <dependencies>
22   <!-- https://mvnrepository.com/artifact/com.mysql/mysql-connector-j -->
23   <dependency>
24     <groupId>com.mysql</groupId>
25     <artifactId>mysql-connector-j</artifactId>
26     <version>8.3.0</version>
27   </dependency>
28
```

- Εισάγουμε το dependency στο POM.xml και κάνουμε save



Maven Dependencies

Full Stack, SOA & Maven



- Όταν κάνουμε save το POM.xml στο φάκελο Maven Dependencies κατεβαίνει το mysql-connector-j.jar
- Το Google protobuf (protocol buffer) είναι ένα transitive dependency της MySQL. Κάνει serialization / deserialization σε binary form και είναι χρήσιμο για την αποθήκευση BLOB (Binary Large Objects) στην MySQL
- Είναι language-agnostic, μπορεί να χρησιμοποιηθεί στην Java καθώς και σε άλλες γλώσσες



Full Stack Project

Full Stack, SOA & Maven

- Θα αναπτύξουμε το project του προηγούμενου κεφαλαίου, σε Swing, αλλά τώρα με maven και αρχιτεκτονική N-Tier, SOA
- Η ανάπτυξη θα γίνει με το IntelliJ, αλλά μπορούμε να χρησιμοποιήσουμε τον κώδικα των φορμών (View) του προηγούμενου project



Δημιουργία Maven Project

Full Stack, SOA & Maven

New Project

To create a general Maven project, go to the [New Project](#) page.

Name:

Location:

Project will be created in: ~\IdeaProjects\idea1023\school6-app

☐ Create Git repository

JDK:

Catalog: [Manage catalogs...](#)

Archetype:

Version:

Additional Properties

No properties

Advanced Settings

GroupId:

ArtifactId:

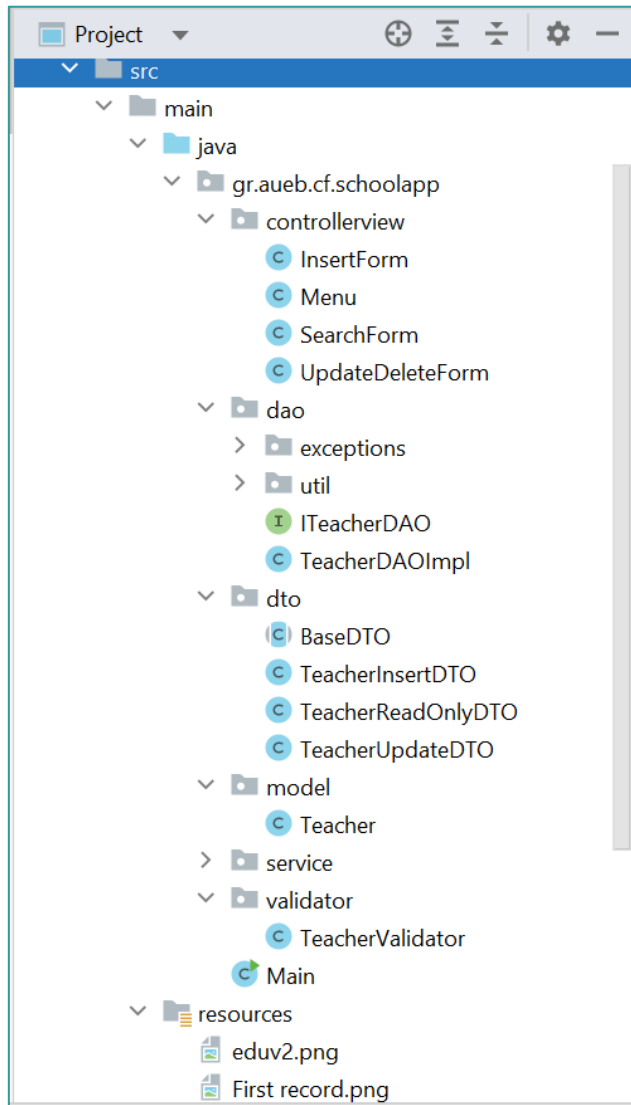
Version:

- Επιλέγουμε quickstart archetype
- Το naming convention στα maven projects είναι kebab-lowercase



Δομή Project στο IntelliJ

Full Stack, SOA & Maven



- Αριστερά απεικονίζονται τα tiers / packages στο πλαίσιο του SOA (Service Oriented Architecture) της τελικής μορφής του project
- Καθώς και τα resources



POM.xml

```
1 <?xml version="1.0" encoding="UTF-8"?>
2
3 <project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4     xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/maven-v4_0_0.xsd">
5     <modelVersion>4.0.0</modelVersion>
6
7     <groupId>gr.aueb.cf</groupId>
8     <artifactId>school6-test</artifactId>
9     <version>1.0-SNAPSHOT</version>
10
11     <name>school6-test</name>
12     <!-- FIXME change it to the project's website -->
13     <url>http://www.example.com</url>
14
15     <properties>
16         <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
17         <maven.compiler.source>17</maven.compiler.source>
18         <maven.compiler.target>17</maven.compiler.target>
19     </properties>
```

- Ορίζουμε ως JDK version το 17



Dependencies

```
21 <dependencies>
22   <!-- https://mvnrepository.com/artifact/com.mysql/mysql-connector-j -->
23   <dependency>
24     <groupId>com.mysql</groupId>
25     <artifactId>mysql-connector-j</artifactId>
26     <version>8.3.0</version>
27   </dependency>
28
29   <!-- https://mvnrepository.com/artifact/org.apache.commons/commons-dbcp2 -->
30   <dependency>
31     <groupId>org.apache.commons</groupId>
32     <artifactId>commons-dbcp2</artifactId>
33     <version>2.12.0</version>
34   </dependency>
35 </dependencies>
```

- Έχουμε δύο dependencies, ένα για το mysql-connector-j και ένα για το connection pool (dbcp2) του Apache Commons



Model

Full Stack, SOA & Maven

- Έχοντας πρώτα φτιάξει τη ΒΔ, στη συνέχεια φτιάχνουμε το model που αποτελεί ένα abstraction των πινάκων της ΒΔ
- Στο παράδειγμα θα αναπτύξουμε την κλάση Teacher και όλες οι άλλες κλάσεις αναπτύσσονται παρόμοια



Κλάση Teacher

Full Stack, SOA & Maven

```
1  package gr.aueb.cf.schoolapp.model;
2
3  public class Teacher {
4      private int id;
5      private String firstname;
6      private String lastname;
7
8      public Teacher() {}
9
10     public Teacher(int id, String firstname, String lastname) {...}
11
12
13
14
15
16     public int getId() { return id; }
17
18     public void setId(int id) { this.id = id; }
19
20
21     public String getFirstname() { return firstname; }
22
23     public void setFirstname(String firstname) { this.firstname = firstname; }
24
25     public String getLastname() { return lastname; }
26
27     public void setLastname(String lastname) { this.lastname = lastname; }
28
29
30
31
32
33
34
35     @Override
36     public String toString() {
37         return id + ", " + firstname + ", " + lastname;
38     }
39 }
```



TeacherDAOException

Full Stack, SOA & Maven

```
1 package gr.aueb.cf.schoolapp.dao.exceptions;
2
3 import java.io.Serial;
4
5 public class TeacherDAOException extends Exception {
6     @java.io.Serial
7     private static final long serialVersionUID = 1L;
8
9     public TeacherDAOException(String s) {
10         super(s);
11     }
12 }
```

- Θα λειτουργήσει ως wrapper της SQLException



DAO – Interface / Public API

Full Stack, SOA & Maven

```
1 package gr.aueb.cf.schoolapp.dao;
2
3 import gr.aueb.cf.schoolapp.dao.exceptions.TeacherDAOException;
4 import gr.aueb.cf.schoolapp.model.Teacher;
5
6 import java.util.List;
7
8 public interface ITeacherDAO {
9     Teacher insert(Teacher teacher) throws TeacherDAOException;
10    Teacher update(Teacher teacher) throws TeacherDAOException;
11    void delete(int id) throws TeacherDAOException;
12    List<Teacher> getByLastname(String lastname) throws TeacherDAOException;
13    Teacher getById(int id) throws TeacherDAOException;
14 }
```



Insert

```
13 public class TeacherDAOImpl implements ITeacherDAO {
14     @Override
15     public Teacher insert(Teacher teacher) throws TeacherDAOException {
16         String sql = "INSERT INTO TEACHERS (FIRSTNAME, LASTNAME) VALUES (?, ?)";
17
18         try (Connection conn = DBUtil.getConnection();
19             PreparedStatement p = conn.prepareStatement(sql)) {
20
21             String firstname = teacher.getFirstname();
22             String lastname = teacher.getLastname();
23
24             if (firstname.equals("") || lastname.equals("")) {
25                 return null;
26             }
27
28             p.setString(1, firstname);
29             p.setString(2, lastname);
30             p.executeUpdate();
31             return teacher;
32         } catch (SQLException e) {
33             // e.printStackTrace(); // logging
34             throw new TeacherDAOException("SQL Error in Teacher " + teacher + " insertion");
35         }
36     }
```



Update

```
38      @Override
39      public Teacher update(Teacher teacher) throws TeacherDAOException {
40          String sql = "UPDATE TEACHERS SET FIRSTNAME = ?, LASTNAME = ? WHERE ID = ?";
41
42          try (Connection conn = DBUtil.getConnection();
43              PreparedStatement p = conn.prepareStatement(sql)) {
44
45              int id = teacher.getId();
46              String firstname = teacher.getFirstname();
47              String lastname = teacher.getLastname();
48
49              if (firstname.equals("") || lastname.equals("")) {
50                  return null;
51              }
52
53              p.setString(1, firstname);
54              p.setString(2, lastname);
55              p.setInt(3, id);
56              p.executeUpdate();
57              return teacher;
58          } catch (SQLException e) {
59              //e.printStackTrace();
60              throw new TeacherDAOException("SQL Error in Teacher " + teacher.getLastname() + " update");
61          }
62      }
```



Delete

```
64      @Override
65      public void delete(int id) throws TeacherDAOException {
66          String sql = "DELETE FROM TEACHERS WHERE ID = ?";
67
68          try (Connection conn = DBUtil.getConnection();
69              PreparedStatement p = conn.prepareStatement(sql)) {
70              p.setInt(1, id);
71              p.executeUpdate();
72          } catch (SQLException e) {
73              //e.printStackTrace();
74              throw new TeacherDAOException("SQL Error in Teacher with id = " + id + " deleted");
75          }
76      }
```

- Τα `printStackTrace` τα αφήνουμε κατά τη φάση του Development, δεν τα χρειαζόμαστε κατά τη φάση του production



Search με Lastname

```
78 @Override
79 public List<Teacher> getByLastname(String lastname) throws TeacherDAOException {
80     String sql = "SELECT ID, FIRSTNAME, LASTNAME FROM TEACHERS WHERE LASTNAME LIKE ?";
81     ResultSet rs;
82     List<Teacher> teachers = new ArrayList<>();
83
84     try (Connection conn = DBUtil.getConnection();
85         PreparedStatement p = conn.prepareStatement(sql)) {
86
87         p.setString(1, lastname + '%');
88         rs = p.executeQuery();
89
90         while (rs.next()) {
91             Teacher teacher = new Teacher(
92                 rs.getInt("ID"),
93                 rs.getString("FIRSTNAME"),
94                 rs.getString("LASTNAME")
95             );
96             teachers.add(teacher);
97         }
98
99         return teachers;
100     } catch (SQLException e) {
101         // e.printStackTrace();
102         throw new TeacherDAOException("SQL Error in Teacher with lastname = " + lastname);
```



Search με id

```
106      @Override
107      public Teacher getById(int id) throws TeacherDAOException {
108          Teacher teacher = null;
109          ResultSet rs;
110          String sql = "SELECT ID, FIRSTNAME, LASTNAME FROM TEACHERS WHERE ID = ?";
111
112          try (Connection conn = DBUtil.getConnection();
113               PreparedStatement p = conn.prepareStatement(sql)) {
114
115              p.setInt(1, id);
116              rs = p.executeQuery();
117
118              if (rs.next()) {
119                  teacher = new Teacher(
120                      rs.getInt("ID"),
121                      rs.getString("FIRSTNAME"),
122                      rs.getString("LASTNAME")
123                  );
124              }
125              return teacher;
126          } catch (SQLException e) {
127              // e.printStackTrace();
128              throw new TeacherDAOException("SQL Error in Teacher with id = " + id);
129          }
130      }
```




Service Layer

- Στη συνέχεια θα ορίσουμε το Service Layer
- Χρειαζόμαστε το DTO Layer που λειτουργεί ως user model και αναπαριστά τις φόρμες που βλέπει ο χρήστης και τα δεδομένα που εισάγει
- Αυτά μετατρέπονται σε DTOs και αποστέλλονται στο Service Layer, ως παράμετρος στις μεθόδους του Service Layer
- Θα πρέπει επίσης να ορίσουμε το public API του Service Layer τυπικά μέσα σε ένα Interface και μετά να περάσουμε στην υλοποίησή του



DTO (Data Transfer Object)

Full Stack, SOA & Maven

```
1 package gr.aueb.cf.schoolapp.dto;
2
3 public class TeacherDTO {
4     private int id;
5     private String firstname;
6     private String lastname;
7
8     public TeacherDTO() {}
9
10    public TeacherDTO(int id, String firstname, String lastname) {...}
11
12
13
14
15
16    public int getId() { return id; }
17
18
19    public void setId(int id) { this.id = id; }
20
21
22    public String getFirstname() { return firstname; }
23
24
25    public void setFirstname(String firstname) { this.firstname = firstname; }
26
27
28    public String getLastName() { return lastname; }
29
30
31    public void setLastName(String lastname) { this.lastname = lastname; }
32
33
34 }
```



ITeacherService - insertTeacher

Full Stack, SOA & Maven

```
1  package gr.aueb.cf.schoolapp.service;
2
3  import gr.aueb.cf.schoolapp.dao.exceptions.TeacherDAOException;
4  import gr.aueb.cf.schoolapp.dto.TeacherDTO;
5  import gr.aueb.cf.schoolapp.model.Teacher;
6  import gr.aueb.cf.schoolapp.service.exceptions.TeacherNotFoundException;
7
8  import java.util.List;
9
10 public interface ITeacherService {
11
12     /**
13      * Inserts a {@link Teacher} based on the data carried by the
14      * {@link TeacherDTO}.
15      *
16      * @param teacherToInsert
17      *           DTO object that contains the data.
18      * @return
19      *           The inserted teacher instance.
20      * @throws TeacherDAOException
21      *           if any DAO exception happens.
22      */
23     Teacher insertTeacher(TeacherDTO teacherToInsert) throws TeacherDAOException;
```



updateTeacher

```
25  /**
26      * Updates a {@link Teacher} based on the data carried by the
27      * {@link TeacherDTO}.
28      *
29      * @param teacherToUpdate
30      *         DTO object that contains the data
31      *         of the new {@link Teacher}.
32      * @return
33      *         the update instance of the {@link Teacher}.
34      * @throws TeacherNotFoundException
35      *         if any Teacher identified by their id
36      *         was not found.
37      * @throws TeacherDAOException
38      *         if any DAO exception happens.
39      */
40  Teacher updateTeacher(TeacherDTO teacherToUpdate)
41      throws TeacherDAOException, TeacherNotFoundException;
42
```



deleteTeacher

```
43  /**
44      * Deletes a {@link Teacher} based on the data carried by the
45      * {@link TeacherDTO}.
46      *
47      * @param id
48      *         the id of the teacher to be deleted
49      * @throws TeacherNotFoundException
50      *         if any Teacher needed to be deleted
51      *         has not found.
52      * @throws TeacherDAOException
53      *         if any error happens between the driver
54      *         and the server at the DAO Level.
55      */
56  void deleteTeacher(int id) throws TeacherDAOException, TeacherNotFoundException;
```



getTeachersByLastname

Full Stack, SOA & Maven

```
59      * Searches and gets back to the caller a list
60      * of the {@link Teacher} objects identified
61      * by their lastname or lastname's initial letters.
62      *
63      * @param lastname
64      *         a String object that contains the
65      *         surname or the letters that the
66      *         surname starts with, of the {@link Teacher}
67      *         objects we are looking for.
68      * @return
69      *         a List that contains the results of
70      *         the search, that is a List of {@link Teacher}
71      *         objects.
72      * @throws TeacherDAOException
73      *         if any error happens between the driver
74      *         and the server.
75      */
76      List<Teacher> getTeachersByLastname(String lastname)
77      |               throws TeacherDAOException;
78      }
```



TeacherServiceImpl - insert

Full Stack, SOA & Maven

```
1  package gr.aueb.cf.schoolapp.service;
2
3  import ...
4
11
12  public class TeacherServiceImpl implements ITeacherService {
13      private final ITeacherDAO teacherDAO;
14
15      public TeacherServiceImpl(ITeacherDAO teacherDAO) { this.teacherDAO = teacherDAO; }
16
17
18
19      @Override
20      public Teacher insertTeacher(TeacherDTO teacherToInsert) throws TeacherDAOException {
21          if (teacherToInsert == null) return null;
22
23          try {
24              Teacher teacher = mapTeacher(teacherToInsert);
25              return teacherDAO.insert(teacher);
26
27          } catch (TeacherDAOException e) {
28              // e.printStackTrace();
29              throw e;
30          }
31      }
```



updateTeacher

```
33  @Override
34  public Teacher updateTeacher(TeacherDTO teacherToUpdate)
35      throws TeacherDAOException, TeacherNotFoundException {
36
37      if (teacherToUpdate == null) return null;
38
39      try {
40          if (teacherDAO.getById(teacherToUpdate.getId()) == null) {
41              throw new TeacherNotFoundException("Teacher with id " + teacherToUpdate.getId() + " was not found");
42          }
43
44          Teacher teacher = mapTeacher(teacherToUpdate);
45          return teacherDAO.update(teacher);
46      } catch (TeacherDAOException | TeacherNotFoundException e) {
47          // e.printStackTrace();
48          throw e;
49      }
50  }
```




deleteTeacher

```
52      @Override
53      public void deleteTeacher(int id) throws TeacherDAOException, TeacherNotFoundException {
54          try {
55              if (teacherDAO.getById(id) == null) {
56                  throw new TeacherNotFoundException("Teacher with id " + id + " not found");
57              }
58              teacherDAO.delete(id);
59          } catch (TeacherDAOException | TeacherNotFoundException e) {
60              e.printStackTrace();
61              throw e;
62          }
63      }
```



getTeacherByLastname

```
65  @Override
66  public List<Teacher> getTeachersByLastname(String lastname)
67      throws TeacherDAOException {
68      List<Teacher> teachers = new ArrayList<>();
69      if (lastname == null) return teachers;
70
71      try {
72          teachers = teacherDAO.getByLastname(lastname);
73          if (teachers.size() == 0) throw new TeacherNotFoundException("Not teacher found with lastname starting with "
74              + lastname);
75          return teachers;
76      } catch (TeacherDAOException e) {
77          // e.printStackTrace();
78          throw e;
79      }
80  }
81
82  @ private Teacher mapTeacher(TeacherDTO dto) {
83      return new Teacher(dto.getId(), dto.getFirstname(), dto.getLastname());
84  }
85  }
```

- Επιστρέφει κενή List αν δεν βρεθεί κανένας teacher. Εναλλακτικά θα μπορούσαμε να έχουμε TeacherNotFoundException.



TeacherNotFoundException

Full Stack, SOA & Maven

```
1 package gr.aueb.cf.schoolapp.service.exceptions;
2
3 import gr.aueb.cf.schoolapp.model.Teacher;
4
5 public class TeacherNotFoundException extends Exception {
6     private final static long serialVersionUID = 1L;
7
8     @Override
9     public TeacherNotFoundException(Teacher teacher) {
10         super("Teacher with id " + teacher.getId() + " does not exist");
11     }
12
13     public TeacherNotFoundException(String s) { super(s); }
14
15 }
```



DBUtil (1)

- Ορίζουμε το BasicDataSource για τη σύνδεση
- Το password για λόγους ασφαλείας το παίρνουμε από την system variable που έχουμε ορίσει (TS_USER_PASSWORD)

```
3 import java.sql.Connection;
4 import java.sql.SQLException;
5
6 import org.apache.commons.dbcp2.BasicDataSource;
7
8 public class DBUtil {
9
10     private final static BasicDataSource ds = new BasicDataSource();
11     private static Connection conn;
12
13     /**
14      * No instances of this class should be available
15      */
16     private DBUtil(){ }
17
18     static {
19         ds.setUrl("jdbc:mysql://localhost:3306/tsdb?serverTimezone=UTC");
20         ds.setUsername("thanos3");
21         ds.setPassword(System.getenv("TS_USER_PASSWORD"));
22         ds.setInitialSize(8);
23         ds.setMaxTotal(32);
24         ds.setMinIdle(8);
25         ds.setMaxIdle(10);
26
27     }
```



DBUtil (2)

```
29     public static Connection getConnection() throws SQLException {
30         conn = ds.getConnection();
31         //return ds.getConnection();
32         return conn;
33     }
34
35     public static void closeConnection() {
36         try {
37             if (conn != null) conn.close();
38         } catch (SQLException e) {
39             e.printStackTrace();
40         }
41     }
42 }
```



Back End

Full Stack, SOA & Maven

- Στο σημείο αυτό έχει ολοκληρωθεί το μεγάλο μέρος του Backend, που είναι το Service Layer και το DAO
- Μένουν οι Controllers που στο Swing είναι μαζί με το View



Front-End

Full Stack, SOA & Maven

- Σε αυτή τη φάση είμαστε πολύ ευέλικτοι γιατί οποιοδήποτε front-end αναπτύξουμε μπορεί πλέον να καλέσει το back-end και να πάρει υπηρεσίες
- Για παράδειγμα στο Swing μπορούμε απλά να προσαρμόσουμε αυτά που έχουμε κάνει μέχρι τώρα και οι Listeners απλά να καλούνε το Service Layer



Insert Listener - Swing

Full Stack, SOA & Maven

```
btnInsert.addActionListener(new ActionListener() {  
    public void actionPerformed(ActionEvent e) {  
  
        String inputLastname = frmSname.getText().trim();  
        String inputFirstname = frmFname.getText().trim();  
  
        if (inputLastname.equals("") || inputFirstname.equals("")) {  
            JOptionPane.showMessageDialog(null, "Not valid input", "INSERT ERROR", JOptionPane.ERROR_MESSAGE);  
        }  
  
        try {  
            TeacherDTO teacherDTO = new TeacherDTO();  
            teacherDTO.setFirstname(inputFirstname);  
            teacherDTO.setLastname(inputLastname);  
  
            Teacher teacher = teacherService.insertTeacher(teacherDTO);  
  
            JOptionPane.showMessageDialog(null, "Teacher" + teacher.getLastname()  
                + " was inserted", "INSERT", JOptionPane.PLAIN_MESSAGE);  
        } catch (TeacherDAOException e1) {  
            String message = e1.getMessage();  
            JOptionPane.showMessageDialog(null, message, "Error", JOptionPane.ERROR_MESSAGE);  
        }  
    }  
});
```

21 12

- Ο insert Listener κάνει κάποια validations του input (ελέγχει αν το όνομα και το επώνυμο δεν είναι κενά) και στη συνέχεια συνθέτει το DTO και καλεί το service layer



Select on Form Activate

Full Stack, SOA & Maven

```
addWindowListener((WindowAdapter) windowActivated(e) → {  
    try {  
        teachers = teacherService.getTeachersByLastname(Main.getSearchForm().getInputLastname());  
  
        listPosition = 0;  
        listSize = teachers.size();  
  
        if (listSize == 0) {  
            udfrm_id.setText("");  
            udfrm_fname.setText("");  
            udfrm_sname.setText("");  
            return;  
        }  
  
        udfrm_id.setText(Integer.toString(teachers.get(listPosition).getId()));  
        udfrm_fname.setText(teachers.get(listPosition).getFirstname());  
        udfrm_sname.setText(teachers.get(listPosition).getLastname());  
    } catch (TeacherDAOException e1) {  
        String message = e1.getMessage();  
        JOptionPane.showMessageDialog(null, message, "Error in getting teachers", JOptionPane.ERROR_MESSAGE);  
    }  
});
```



Update Listener

```
btnUpdate.addActionListener(new ActionListener() {  
    public void actionPerformed(ActionEvent e) {  
        String id = udfm_id.getText().trim();  
        String inputLastname = udfm_sname.getText().trim();  
        String inputFirstname = udfm_fname.getText().trim();  
  
        if (inputLastname.equals("") || inputFirstname.equals("") || id.equals("")) {  
            JOptionPane.showMessageDialog(null, "Not valid input", "UPDATE ERROR", JOptionPane.ERROR_MESSAGE);  
            return;  
        }  
  
        try {  
            TeacherDTO teacherDTO = new TeacherDTO();  
            teacherDTO.setId(Integer.parseInt(id));  
            teacherDTO.setFirstname(inputFirstname);  
            teacherDTO.setLastname(inputLastname);  
            Teacher teacher = teacherService.updateTeacher(teacherDTO);  
            JOptionPane.showMessageDialog(null, "Teacher "  
                + " was updated", "UPDATE", JOptionPane.PLAIN_MESSAGE);  
        } catch (TeacherDAOException | TeacherNotFoundException e1) {  
            String message = e1.getMessage();  
            JOptionPane.showMessageDialog(null, message, "Error", JOptionPane.ERROR_MESSAGE);  
        }  
    }  
});
```



Delete Listener

```
btnDelete.addActionListener(new ActionListener() {  
    public void actionPerformed(ActionEvent e) {  
        try {  
            int response;  
            String idStr = udfrm_id.getText();  
            int id;  
  
            if (idStr.equals("")) return;  
  
            id = Integer.parseInt(udfrm_id.getText());  
  
            response = JOptionPane.showConfirmDialog (null, "Είστε σίγουρος;",  
                "Warning", JOptionPane.YES_NO_OPTION);  
  
            if (response == JOptionPane.YES_OPTION){  
                teacherService.deleteTeacher(id);  
                JOptionPane.showMessageDialog (null, "Teacher was deleted successfully",  
                    "DELETE", JOptionPane.INFORMATION_MESSAGE);  
            }  
        } catch (TeacherDAOException | TeacherNotFoundException e1) {  
            String message = e1.getMessage();  
            JOptionPane.showMessageDialog (null, message, "DELETE", JOptionPane.ERROR_MESSAGE);  
        }  
    }  
});
```



First – Previous Buttons

```
btnFirst.addActionListener(new ActionListener() {  
    public void actionPerformed(ActionEvent e) {  
        if (listSize > 0) {  
            listPosition = 0;  
            udfrm_id.setText(String.format("%s", teachers.get(listPosition).getId()));  
            udfrm_sname.setText(teachers.get(listPosition).getLastname());  
            udfrm_fname.setText(teachers.get(listPosition).getFirstname());  
        }  
    }  
});  
  
contentPane.add(btnFirst);  
JButton btnPrev = new JButton("");  
btnPrev.setIcon(new ImageIcon(Objects.requireNonNull(Thread.currentThread().getContextClassLoader().getResourceAsStream("img/prev.png"))));  
btnPrev.addActionListener(new ActionListener() {  
    public void actionPerformed(ActionEvent e) {  
        if (listPosition > 0) {  
            listPosition--;  
            udfrm_id.setText(String.format("%s", teachers.get(listPosition).getId()));  
            udfrm_sname.setText(teachers.get(listPosition).getLastname());  
            udfrm_fname.setText(teachers.get(listPosition).getFirstname());  
        }  
    }  
});
```



Next – Last Buttons

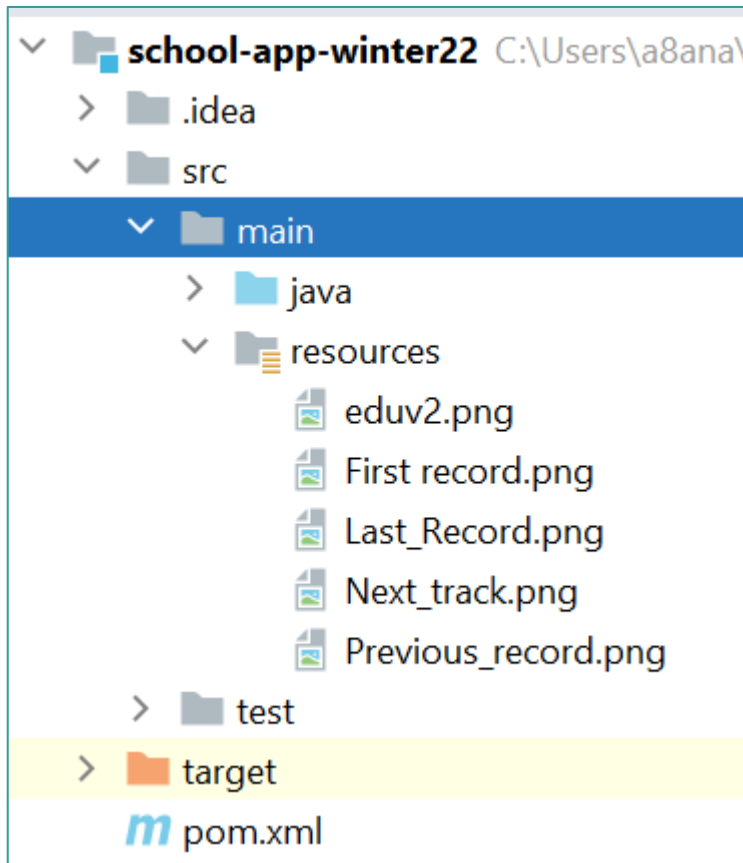
```
btnNext.addActionListener(new ActionListener() {  
    public void actionPerformed(ActionEvent e) {  
        if (listPosition <= listSize - 2) {  
            listPosition++;  
            udfrm_id.setText(String.format("%s", teachers.get(listPosition).getId()));  
            udfrm_sname.setText(teachers.get(listPosition).getLastname());  
            udfrm_fname.setText(teachers.get(listPosition).getFirstname());  
        }  
    }  
});
```

```
JButton btnLast = new JButton("");  
btnLast.setIcon(new ImageIcon(Objects.requireNonNull(Thread.currentThread().getContextClassLoader().getResourceAsStream("img/last.png"))));  
btnLast.addActionListener(new ActionListener() {  
    public void actionPerformed(ActionEvent e) {  
        if (listSize > 0) {  
            listPosition = listSize - 1;  
            udfrm_id.setText(String.format("%s", teachers.get(listPosition).getId()));  
            udfrm_sname.setText(teachers.get(listPosition).getLastname());  
            udfrm_fname.setText(teachers.get(listPosition).getFirstname());  
        }  
    }  
});
```



Resources Folder

Full Stack, SOA & Maven



- Εισάγουμε κάτω από τον main folder τον folder/directory resources και στο IntelliJ τον κάνουμε mark Directory as Resources root (δεξί κλικ πάνω στον resources)
- Εισάγουμε τα αρχεία εικόνων



Maven

```
108     </pluginManagement>
109
110     <plugins>
111         <plugin>
112             <groupId>org.codehaus.mojo</groupId>
113             <artifactId>exec-maven-plugin</artifactId>
114             <version>3.3.0</version>
115             <configuration>
116                 <mainClass>gr.aueb.cf.schoolapp.Main</mainClass>
117             </configuration>
118         </plugin>
119     </plugins>
```

- Για να εκτελέσουμε ένα app με Maven χρειαζόμαστε, το `exec-maven-plugin` το οποίο κάνει config το Maven να μπορεί να τρέχει java programs ορίζοντας επίσης την `mainClass` path
- Εκτελούμε με **`mvn exec:java`** στο root folder του App



Εκτέλεση plugins

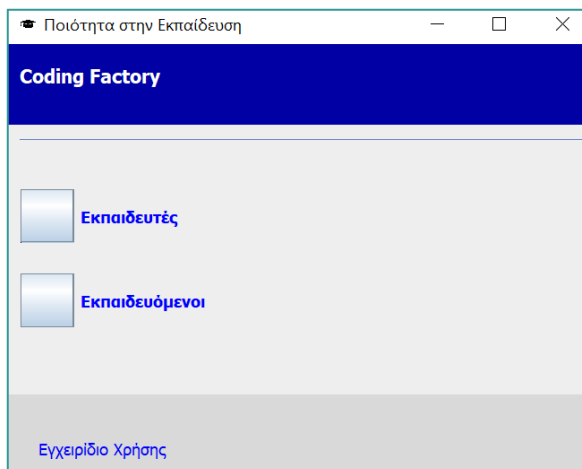
Full Stack, SOA & Maven

- Τα plugins που είναι μέσα στο pluginManagement είναι τα core plugins του Maven που εκτελούνται στα lifecycle phases του Maven. Δεν χρειάζεται να εισαχθούν στο <plugins> section
- Τα εξωτερικά plugins πρέπει να εισάγονται στο <plugins> section



Mvn exec:java

```
Windows PowerShell
PS C:\Users\A8ana\IdeaProjects\idea1023\school6-test> mvn exec:java
[INFO] Scanning for projects...
[INFO]
[INFO] -----< gr.aueb.cf:school6-test >-----
[INFO] Building school6-test 1.0-SNAPSHOT
[INFO] from pom.xml
[INFO] -----[ jar ]-----
[INFO]
[INFO] >>> exec:1.2.1:java (default-cli) > validate @ school6-test >>>
[INFO]
[INFO] <<< exec:1.2.1:java (default-cli) < validate @ school6-test <<<
[INFO]
[INFO] --- exec:1.2.1:java (default-cli) @ school6-test ---
PS C:\Users\A8ana\IdeaProjects\idea1023\school6-test>
```



- Με `mvn exec:java` εκτελείται η `main` όπως έχει προσδιοριστεί στο `plugin`



Release Distribution

Full Stack, SOA & Maven

- Για τη διανομή ενός project σε τρίτους υπάρχουν δύο τύποι jar files
 - Τα jar files που προορίζονται να λειτουργήσουν ως **dependencies** σε άλλα projects
 - Τα **executable jars** που περιέχουν και ένα manifest file (τα manifest files γενικά περιέχουν metadata) με το **FQN (Fully Qualified Name)** της **main class**
- Για να δημιουργήσουμε executable jars που περιέχουν και dependencies μπορούμε να χρησιμοποιήσουμε το **maven-shade-plugin** ή το **maven-assembly-plugin**, ώστε το τελικό jar να περιέχονται και τα dependencies



Maven-shade-plugin

```
94 <plugin>
95   <groupId>org.apache.maven.plugins</groupId>
96   <artifactId>maven-shade-plugin</artifactId>
97   <version>3.3.0</version>
98   <executions>
99     <execution>
100       <phase>package</phase>
101       <goals>
102         <goal>shade</goal>
103       </goals>
104       <configuration>
105         <minimizeJar>>false</minimizeJar>
106         <transformers>
107           <transformer
108             implementation="org.apache.maven.plugins.shade.resource.ServicesResourceTransformer" />
109           <transformer
110             implementation="org.apache.maven.plugins.shade.resource.ManifestResourceTransformer">
111             <mainClass>gr.aueb.cf.schoolapp.Main</mainClass>
112           </transformer>
113         </transformers>
114         <createDependencyReducedPom>>false</createDependencyReducedPom>
115         <finalName>${project.artifactId}</finalName>
116       </configuration>
117     </execution>
118   </executions>
119 </plugin>
120
121 </plugins>
122
123 </build>
124 </project>
```

Οι transformers κάνουν bundle τα διάφορα jars και resources σε ένα fat jar που περιέχει όλη την πληροφορία για να τρέξει το τελικό jar

Ο 2^{ος} transformer εισάγει την Main class στο META-INF/MANIFEST.MF που περιέχει metadata για το project

- Η φάση που θα εκτελεστεί είναι η maven package



Build, Test, Package

Full Stack, SOA & Maven

- Κατά τη φάση package του maven, το project γίνεται compile, test και package δηλαδή μεταγλωττίζεται, τρέχει τυχόν tests και δημιουργεί το τελικό jar (εδώ executable jar)
- Εισάγουμε το JUnit dependency που χρησιμοποιείται κατά τη φάση του testing και διαγράφουμε τυχόν Test κλάσεις που υπάρχουν από το setup του project (το IntelliJ δημιουργεί αυτόματα μία τέτοιο κλάση και τη διαγράφουμε)



Mvn package

```
a8ana@thanassis-pc MINGW64 ~/IdeaProjects/idea1023/school6-test (main)
$ mvn package
[INFO] Scanning for projects...
[INFO]
[INFO] -----< gr.aueb.cf:school6-test >-----
[INFO] Building school6-test 1.0-SNAPSHOT
[INFO] from pom.xml
[INFO] -----[ jar ]-----
[INFO]
[INFO] --- resources:3.0.2:resources (default-resources) @ school6-test ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] Copying 1 resource
[INFO]
[INFO] --- compiler:3.8.0:compile (default-compile) @ school6-test ---
[INFO] Nothing to compile - all classes are up to date
[INFO]
[INFO] --- resources:3.0.2:testResources (default-testResources) @ school6-test
```

```
a8ana@thanassis-pc MINGW64 ~/IdeaProjects/idea1023/school6-test (main)
$ cd target/

a8ana@thanassis-pc MINGW64 ~/IdeaProjects/idea1023/school6-test/target (main)
$ ls
classes/                  maven-archiver/          school6-test.jar
generated-sources/        maven-status/            test-classes/
generated-test-sources/   school6-test-1.0-SNAPSHOT.jar
```



Εκτέλεση jar (java -jar)

Full Stack, SOA & Maven

```
a8ana@thanassis-pc MINGW64 ~/IdeaProje  
$ java -Xmx512m -jar school6-test.jar
```

- Με `-Xmx` ορίζουμε το max size του Heap (είναι optional, το default είναι συνήθως το $\frac{1}{4}$ της physical memory)
- Με `java -jar <όνομα-jar>.jar` εκτελούμε το executable jar



Maven-assembly-plugin

Full Stack, SOA & Maven

```
122 <plugin>
123   <groupId>org.apache.maven.plugins</groupId>
124   <artifactId>maven-assembly-plugin</artifactId>
125   <version>3.3.0</version>
126   <configuration>
127     <archive>
128       <manifest>
129         <mainClass>gr.aueb.cf.schoolapp.Main</mainClass>
130       </manifest>
131     </archive>
132     <descriptorRefs>
133       <descriptorRef>jar-with-dependencies</descriptorRef>
134     </descriptorRefs>
135   </configuration>
136   <executions>
137     <execution>
138       <id>make-assembly</id>
139       <phase>package</phase>
140       <goals>
141         <goal>single</goal>
142       </goals>
143     </execution>
144   </executions>
145 </plugin>
```

- Και με το maven assembly plugin μπορούμε να δημιουργήσουμε executable jars



mvn clean package

Full Stack, SOA & Maven

```
a8ana@thanassis-pc MINGW64 ~/IdeaProjects/idea1023/school6-test (main)
$ mvn clean package
[INFO] Scanning for projects...
[INFO] -----< gr.aueb.cf:school6-test >-----
[INFO] Building school6-test 1.0-SNAPSHOT
[INFO] from pom.xml
[INFO] -----[ jar ]-----
[INFO] --- clean:3.1.0:clean (default-clean) @ school6-test ---
[INFO] Deleting C:\Users\a8ana\IdeaProjects\idea1023\school6-test\target
[INFO] --- resources:3.0.2:resources (default-resources) @ school6-test ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] Copying 1 resource
```




cd target / mvn -jar

```
a8ana@thanassis-pc MINGW64 ~/IdeaProjects/idea1023/school6-test (main)
$ cd target/

a8ana@thanassis-pc MINGW64 ~/IdeaProjects/idea1023/school6-test/target (main)
$ ls -la
total 4792
drwxr-xr-x 1 a8ana 197609      0 Jul 24 13:46 ./
drwxr-xr-x 1 a8ana 197609      0 Jul 24 13:46 ../
drwxr-xr-x 1 a8ana 197609      0 Jul 24 13:46 archive-tmp/
drwxr-xr-x 1 a8ana 197609      0 Jul 24 13:46 classes/
drwxr-xr-x 1 a8ana 197609      0 Jul 24 13:46 generated-sources/
drwxr-xr-x 1 a8ana 197609      0 Jul 24 13:46 maven-archiver/
drwxr-xr-x 1 a8ana 197609      0 Jul 24 13:46 maven-status/
-rw-r--r-- 1 a8ana 197609 4847988 Jul 24 13:46 school6-test-1.0-SNAPSHOT-jar-with-dependencies.jar
-rw-r--r-- 1 a8ana 197609  48844 Jul 24 13:46 school6-test-1.0-SNAPSHOT.jar

a8ana@thanassis-pc MINGW64 ~/IdeaProjects/idea1023/school6-test/target (main)
$ java -Xmx512m -jar school6-test-1.0-SNAPSHOT-jar-with-dependencies.jar
```

- Όπως πριν, εκτελούμε με `java -jar` στον `target`, το fat jar



Εργασία

Full Stack, SOA & Maven

- Αναπτύξτε κατά τον ίδιο τρόπο μία εφαρμογή με domain της επιλογής σας
- Ή αναπτύξτε περαιτέρω την παρούσα εφαρμογή ώστε να περιλαμβάνει και Students