Hibernate Introduction

Maven, Hibernate, Configuration, JPA, Annotations

SoftUni Team

Technical Trainers

Software University

SoftUni Foundation

Software University

http://softuni.bg

Table of Content

Questions

SoftUni Foundation

sli.do

#JavaDB

Maven

Project management and comprehension

Maven Overview

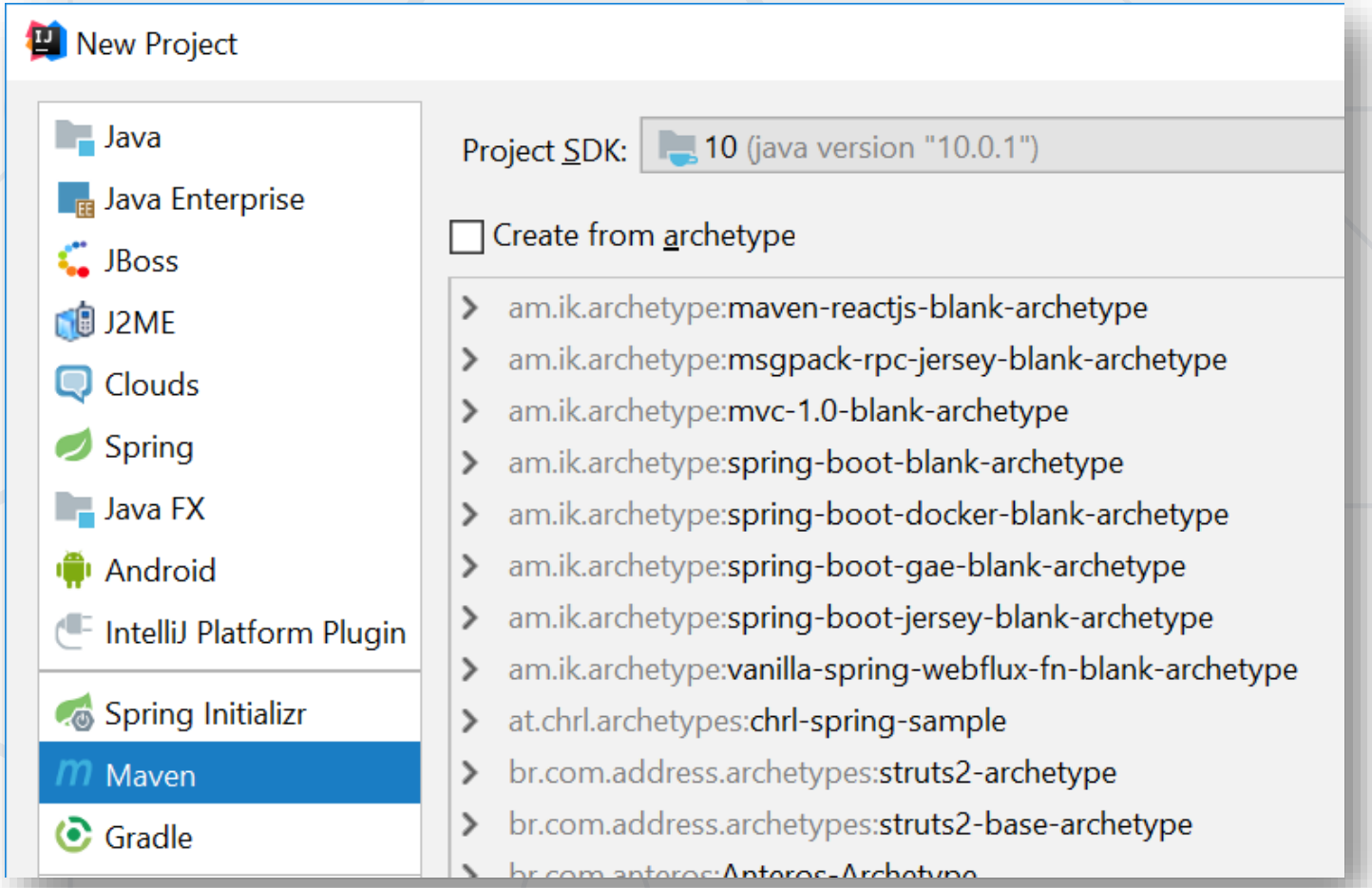
SoftUni Foundation

- Maven is a built automation tool.
  - Describes how software is built and it's dependencies
  - Uses XML files
- Dynamically downloads **Java libraries** and **Maven plug-ins**
  - Projects are configured using a **P**roject **O**bject **M**odel, which is stored in a **pom.xml** file

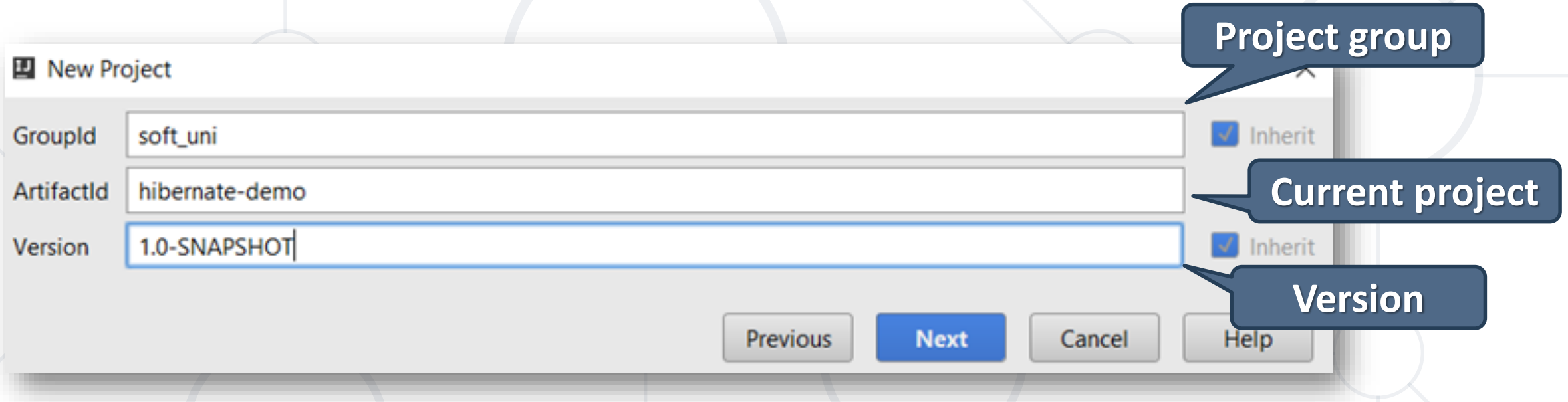
Setup – Creating a Maven project

SoftUni Foundation

- Select "Maven" project from the new project panel:

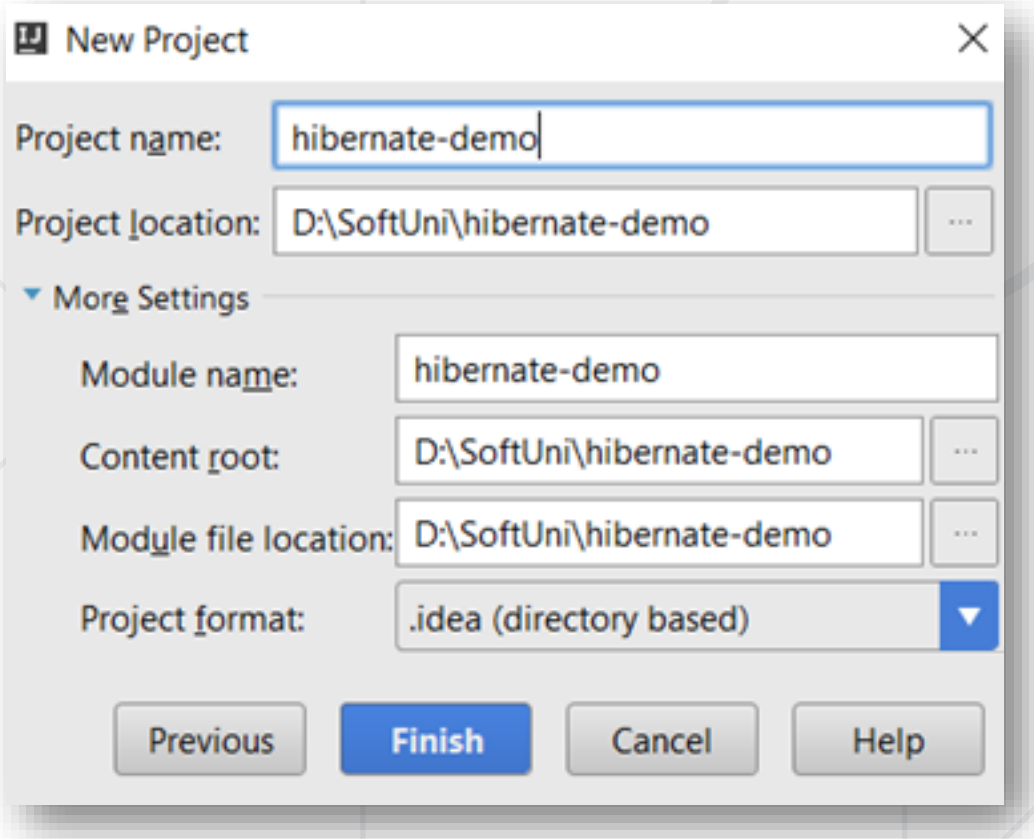


Setup (2)

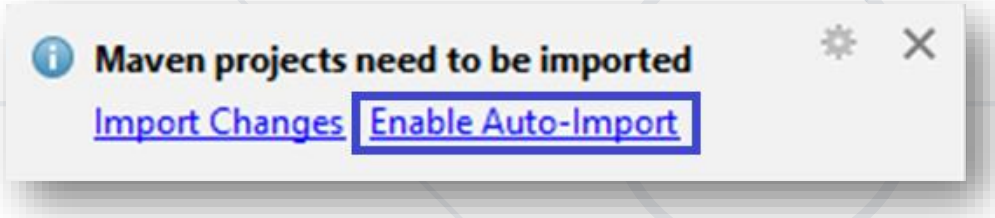

New Project
GroupId
soft_uni
Inherit
ArtifactId
hibernate-demo
Version
1.0-SNAPSHOT
Inherit
Previous
Next
Cancel
Help
Project group
Current project
Version

Setup (3)

- Set up **project name** and **location**

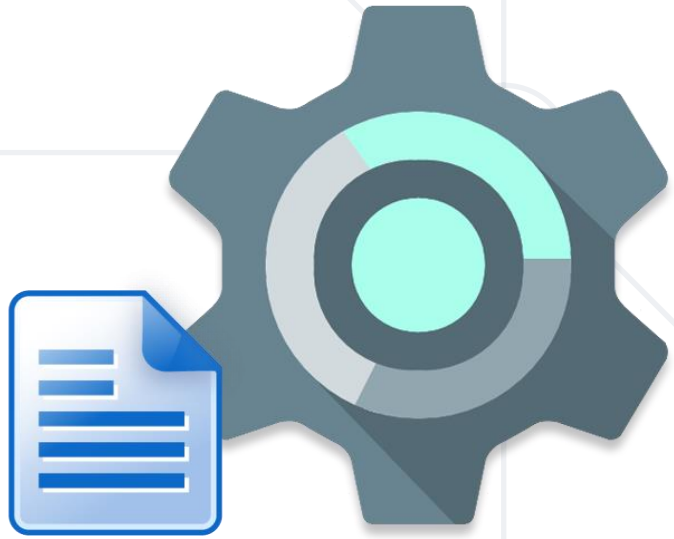


- Set up Maven **auto-import**



Maven Configurations

- A **P**roject **O**bject **M**odel(**POM**) is the fundamental unit of work in Maven
- **Configurations** are held in the **pom.xml** file
  - When executing a task or goal, Maven looks for the POM file in the current directory



POM model

SoftUni Foundation

pom.xml

```
<build>
    <plugins>
        <plugin>
            <groupId>org.apache.maven.plugins</groupId>
            <artifactId>maven-compiler-plugin</artifactId>
            <version>3.5.1</version>
            <configuration>
                <source>1.8</source>
                <target>1.8</target>
            </configuration>
        </plugin>
    </plugins>
</build>
```

Java compile version

Dependencies

- Dependencies are set with the **<dependency>** tag:

pom.xml

```
<dependencies>
        <dependency>
            <groupId>org.hibernate</groupId>
            <artifactId>hibernate-core</artifactId>
            <version>5.2.3.Final</version>
        </dependency>
        <dependency>
            <groupId>mysql</groupId>
            <artifactId>mysql-connector-java</artifactId>
            <version>6.0.4</version>
        </dependency>
</dependencies>
```

Dependency 1

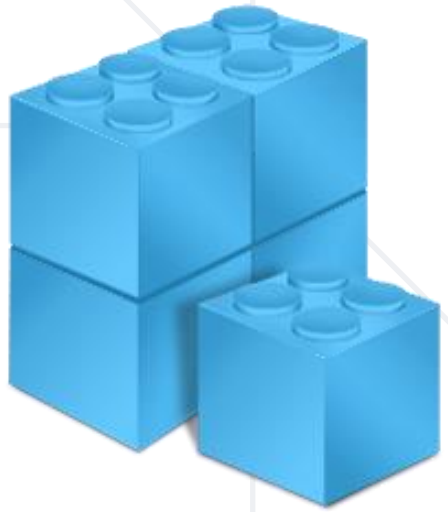
Dependency 2

Hibernate Framework

Mapping Java classes to database tables

Hibernate Framework

- Hibernate is a Java ORM framework
  - Mapping an object-oriented model to a relational database
    - It is implemented by the configuration of an **XML file** or by using **Java Annotations**
  - Maintain the database schema



Java ORM Approaches

SoftUni Foundation

- Different approaches to **Java ORM**:
  - POJO (Plain Old Java Objects) + XML mappings
    - A bit old-fashioned, but very powerful
    - Implemented in the "classical" Hibernate
  - Annotated Java classes (**POJO**) mapped to DB tables
    - Based on Java annotations and XML
    - Easier to implement and maintain
  - Code generation - tools

Hibernate configuration

- Add hibernate as a project dependency

pom.xml

```
<dependencies>
    <dependency>
        <groupId>org.hibernate</groupId>
        <artifactId>hibernate-core</artifactId>
        <version>5.2.3.Final</version>
    </dependency>
    <dependency>
        <groupId>mysql</groupId>
        <artifactId>mysql-connector-java</artifactId>
        <version>5.1.6</version>
    </dependency>
</dependencies>
```

Hibernate

MySQL connector

Hibernate configuration (2)

SoftUni Foundation

hibernate.cfg.xml

```
<?xml version='1.0' encoding='utf-8'?>
<!DOCTYPE hibernate-configuration
        PUBLIC "-//Hibernate/Hibernate Configuration DTD//EN"
        "http://www.hibernate.org/dtd/hibernate-configuration-
3.0.dtd">

<hibernate-configuration>
    <session-factory>
        <property name="hibernate.dialect">
            org.hibernate.dialect.MySQL5Dialect
        </property>
        <property name="hibernate.connection.driver_class">
            com.mysql.jdbc.Driver
        </property>
```

Configuration

SQL Dialect

Driver

Hibernate configuration (2)

SoftUni Foundation

hibernate.cfg.xml

```
<!-- Connection Settings -->
<property name="hibernate.connection.url">
    jdbc:mysql://localhost:3306/school
</property>
<property name="hibernate.connection.username">
    root
</property>
<property name="hibernate.connection.password">
    1234
</property>
<property name="hbm2ddl.auto">
    create
</property>
```

Connection string

User

Pass

Auto strategy

Hibernate configuration (3)

SoftUni Foundation

hibernate.cfg.xml

```
        …
            <!-- List of XML mapping files -->
            <mapping resource="student.hbm.xml"/>
    </session-factory>
</hibernate-configuration>
```

Mapping files

Hibernate Implementation Example

SoftUni Foundation

POJO (Plain Old Java Objects) + XML mappings

```
public class Student {
  private int id;
  private String name;
  private Date registeredOn;

  // Constructor, getters and setters
}
```

Hibernate mapping

SoftUni Foundation

student.hbm.xml

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE hibernate-mapping PUBLIC
        "-//Hibernate/Hibernate Mapping DTD//EN"
        "http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">

<hibernate-mapping>
    <class name="entities.Student" table="students">
        <id name="id" column="id">
            <generator class="identity" />
        </id>
      …
```

Mapping file

Class mapping

Field mapping

Hibernate mapping (2)

SoftUni Foundation

student.hbm.xml

```
…
        <property name="name" column="first_name" />
          <property name="registrationDate" column="registration_date"/>
    </class>
</hibernate-mapping>
```

Field mapping

Hibernate sessions

SoftUni Foundation

Main.java

```
public class Main {
    public static void main(String[] args) {
        Configuration cfg = new Configuration();
        cfg.configure();
        SessionFactory sessionFactory =
                cfg.buildSessionFactory();
        Session session = sessionFactory.openSession();
        session.beginTransaction();

        // Your Code Here
        session.getTransaction().commit();
        session.close();
    }
}
```

Service Registry

Session

Transaction commit

Hibernate save data

SoftUni Foundation

Main.java

```
public static void main(String[] args) {
        //…
        session.beginTransaction();

        Student example = new Student();
        session.save(example);

        session.getTransaction().commit();
                session.close();
    }
}
```

Save object

Hibernate retrieve data by Get

SoftUni Foundation

Main.java

```
public static void main(String[] args) {
      …
         session.beginTransaction();

      Student student = (Student) session.get(Student.class, 1);

         session.getTransaction().commit();
         session.close();
   }
}
```

Get object

Hibernate retrieve data by Query

SoftUni Foundation

Main.java

```
public static void main(String[] args) {
      // …
         session.beginTransaction();

      List<Student> studentList =
      session.createQuery("FROM Student ").list();
         for (Student student : studentList) {
             System.out.println(student.getId());
         }
         session.getTransaction().commit();
         session.close();
    }
}
```

Get list of objects

Hibernate Querying Language - HQL

SELECT

```
"FROM Student"
```

SELECT + WHERE

```
"FROM Student WHERE name = 'John'"
```

SELECT + JOIN

```
"FROM Student AS s
JOIN s.major AS major"
```

Hibernate retrieve data by Criteria

SoftUni Foundation

Main.java

```
public static void main(String[] args) {
      //…
        session.beginTransaction();

      List<Student> studentList =
      session.createCriteria(Student.class)
      .add(Restrictions.like("name",      "P%")).list();
        for (Student student : studentList) {
            System.out.println(student.getId());
        }

        session.getTransaction().commit();
        session.close();
    }
```

Get list of objects by criteria

Java Persistence API

ORM Fundamentals

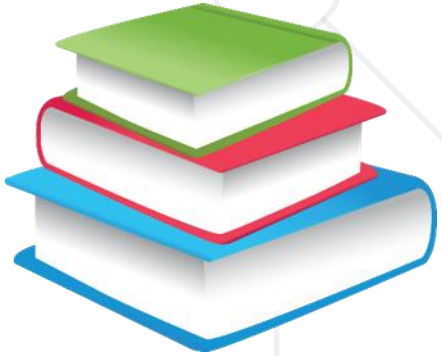
About JPA

SoftUni Foundation

- What is Java Persistence API (JPA)?
  - Database persistence technology for Java (**official standard**)
    - Object-relational mapping (ORM) technology
    - Operates with POJO entities with annotations or XML mappings
    - Implemented by many **ORM engines**: **Hibernate**, **EclipseLink**, …

About JPA (2)

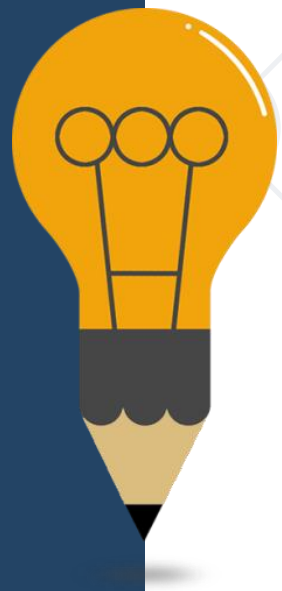
- JPA maps Java classes to database tables
  - Maps relationships between tables as associations between classes
- Provides **CRUD** functionality and queries
  - Create, read, update, delete + queries



Entities in JPA

- A JPA entity is just a POJO class
  - Abstract or concrete **top level** Java class
  - Non-final fields/properties, no-arguments constructor
  - No required interfaces
  - Direct field or property-based access
- Getter/setter can contain logic (e.g. validation)

Entity Class: Student



Student.java

```
@Entity @Table(name = "students")
public class Student {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Column(name = "id")
    private int id;
    @Column(name = "name", length = 50)
    private String name;
    @Column(name = “birth_date")
    private Date birthDate;
    // Getters and setters
}
```

Set table name

Primary key

Identity

Column name

Column name and length

Column name

Annotations

- **@Entity** - Declares the class as an entity or a table
- **@Table** - Declares table name
- **@Basic** - Specifies non-constraint fields explicitly
- **@Transient** - Specifies the property that is not persis tent, i.e.,
  the value is never stored in the database

Annotations (2)

- **@Id** - Specifies the property, use for identity (primary key of a table) of the class
  - **@GeneratedValue** - specifies how the identity attribute can be initialized
    - Automatic, manual, or value taken from a sequence table
- **@Column** -Specifies the column attribute for the per sistence property

JPA Configuration

SoftUni Foundation

pom.xml

```
<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/maven-v4_0_0.xsd">
    <modelVersion>4.0.0</modelVersion>
    <groupId>com.javawebtutor</groupId>
    <artifactId>JPAMavenExample</artifactId>
    <packaging>jar</packaging>
    <version>1.0-SNAPSHOT</version>
    <name>JPAMavenExample</name>
    <url>http://maven.apache.org</url>
     …
```

JPA Configuration (2)

SoftUni Foundation

pom.xml

```
…
<dependencies>
        <dependency>
            <groupId>org.eclipse.persistence</groupId>
            <artifactId>javax.persistence</artifactId>
            <version>2.1.0</version>
        </dependency>
        <dependency>
            <groupId>org.hibernate</groupId>
            <artifactId>hibernate-core</artifactId>
            <version>5.2.3.Final</version>
        </dependency>
```

JPA Configuration (3)

SoftUni Foundation

pom.xml

```
…
<groupId>mysql</groupId>
            <artifactId>mysql-connector-java</artifactId>
            <version>5.1.27</version>
        </dependency>
    </dependencies>
</project>
```

JPA Configuration

SoftUni Foundation

persistence.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<persistence xmlns="http://java.sun.com/xml/ns/persistence"
version="2.0">
    <persistence-unit name="school">
        <properties>
            <property name = "hibernate.connection.url"
value="jdbc:mysql://localhost:3306/school"/>
            <property name =
"hibernate.connection.driver_class"
value="com.mysql.jdbc.Driver"/>
            …
```

JPA Configuration (2)

SoftUni Foundation

persistence.xml

```
            …
        <property name = "hibernate.connection.username" value="root"/>
        <property name = "hibernate.connection.password" value="1234"/>
        <property name = "hibernate.dialect"
value="org.hibernate.dialect.MySQL5Dialect"/>
        <property name = "hibernate.hbm2ddl.auto" value="create"/>
        <property name = "hibernate.show_sql" value = "true" />
      </properties>
   </persistence-unit>
</persistence>
```

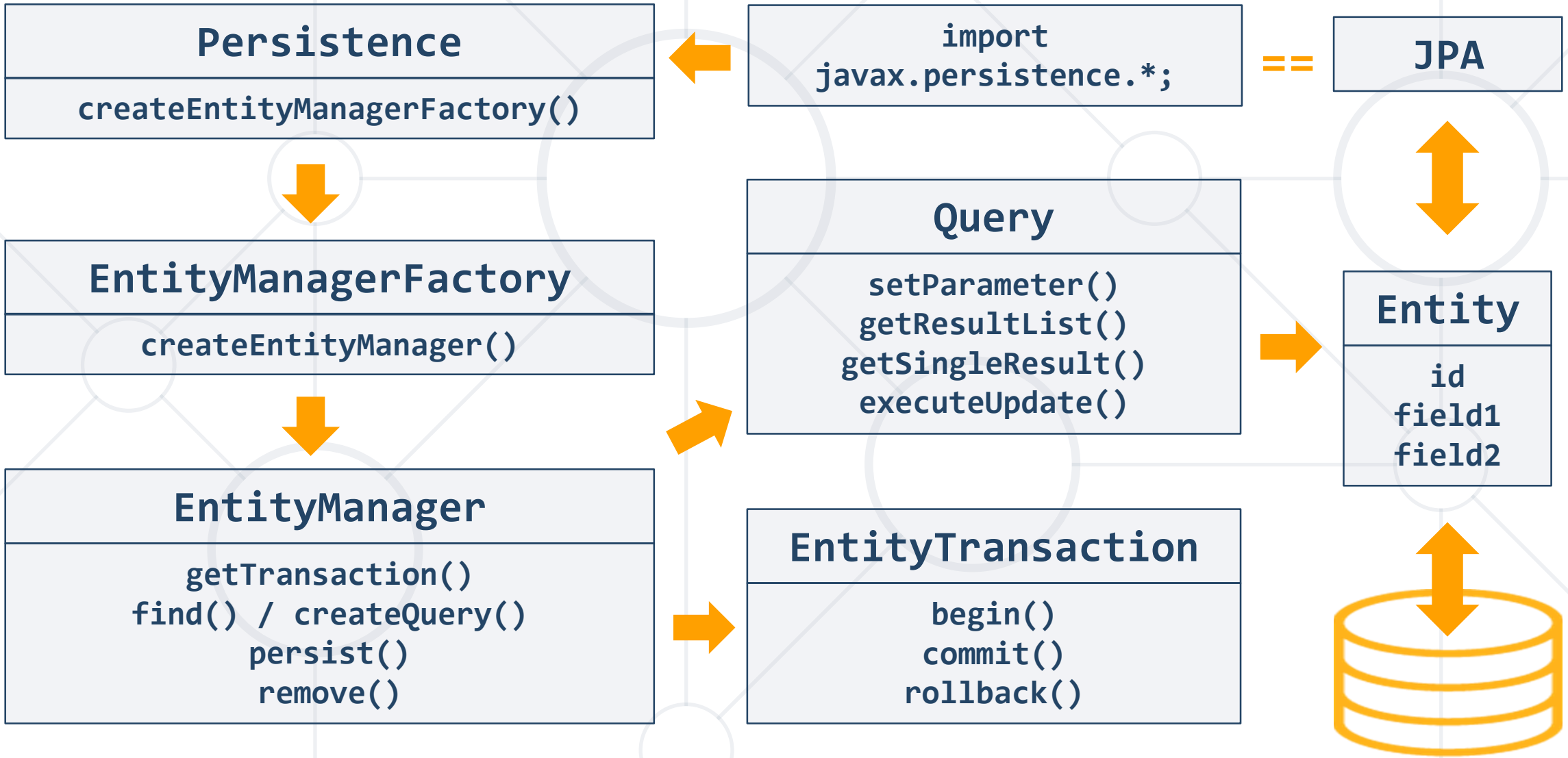
JPA save objects

SoftUni Foundation

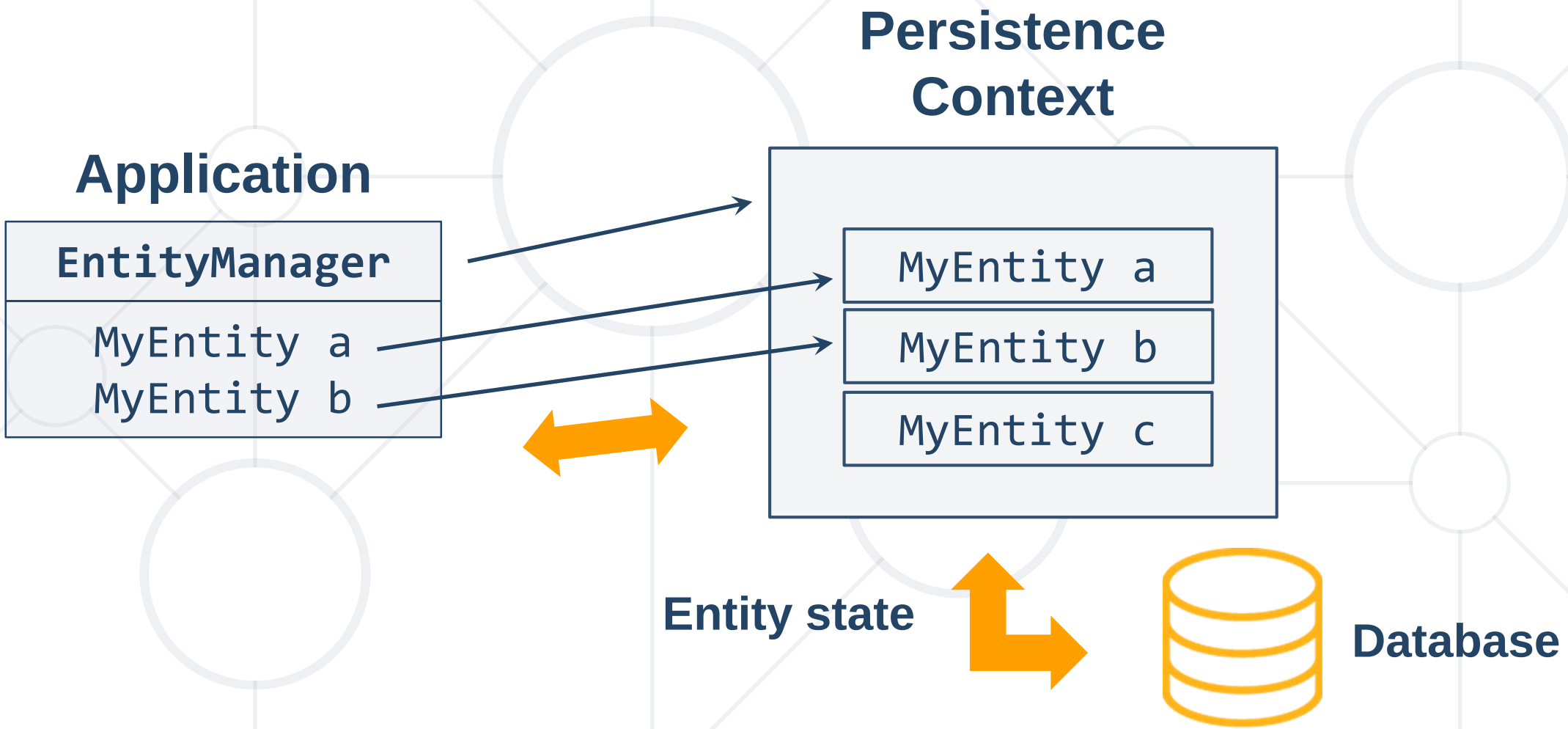
Main.java

```
public static void main(String[] args) {
        EntityManagerFactory emf =
Persistence.createEntityManagerFactory("school");
        EntityManager em = emf.createEntityManager();
        em.getTransaction().begin();
        Student student = new Student("Teo", new Date());
        em.persist(student);
        em.getTransaction().commit();
    }
}
```

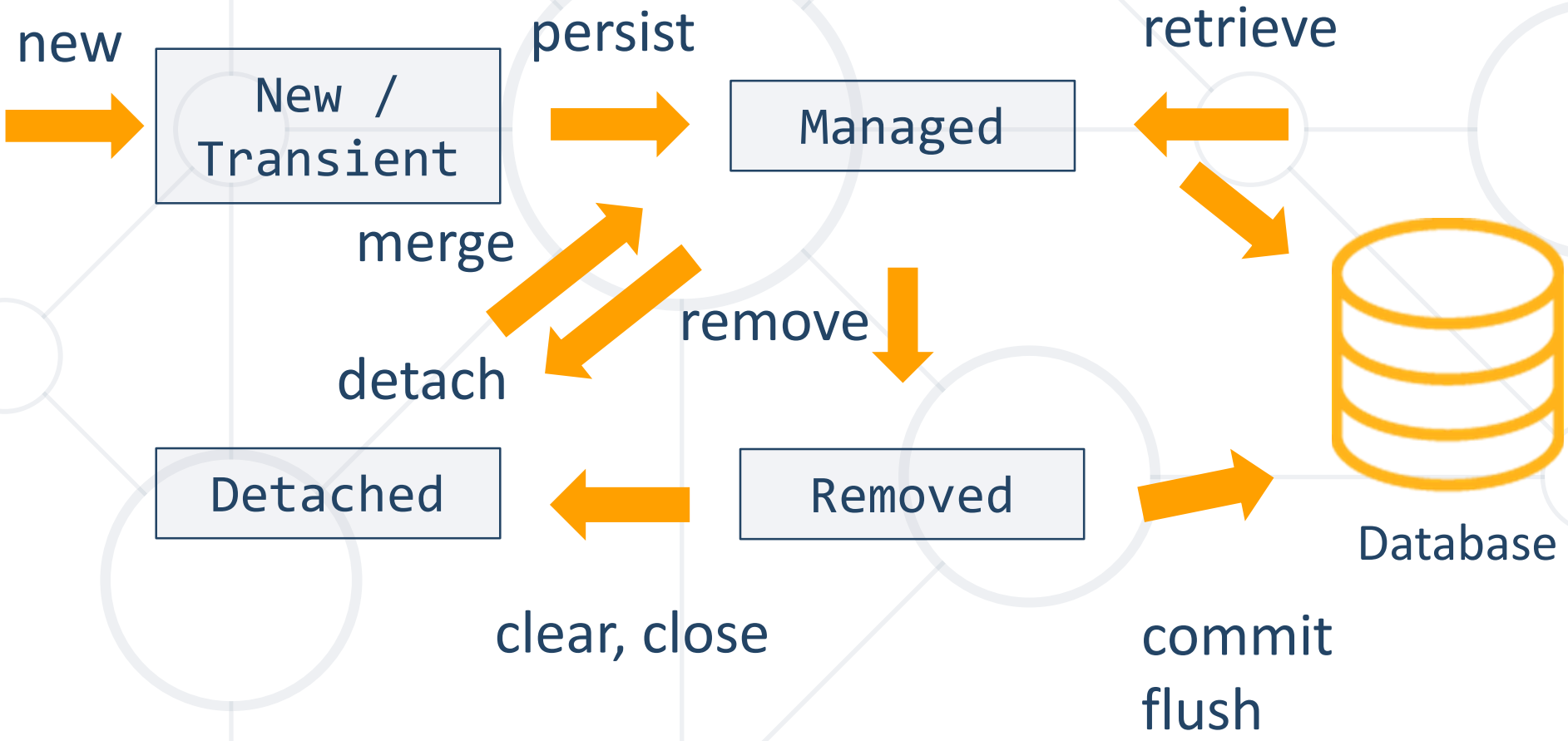
JPA – Java Persistence API



Persistence Context (PC) and Entities



Entity object life cycle



JPA write data methods

- **persist()** – persists given entity object into the DB (SQL INSERT)
- **remove()** – deletes given entity into the DB (SQL DELETE by primary key)
- **refresh()** – reloads given entity from the DB (SQL SELECT by primary key)

JPA write data methods (2)

- **detach()** – removes the object from the persistence context(PC)
- **merge()** – synchronize the state of detached entity with the PC
- **contains()** - determine if given entity is managed by the PC
- **flush()** – writes the changes from PC in the database

JPA read data methods

- **find()** - execute a simple Select query by primary key

Main.java

```
public static void main(String[] args) {
        EntityManagerFactory emf =
Persistence.createEntityManagerFactory("school");

        EntityManager em = emf.createEntityManager();
        em.getTransaction().begin();
        em.find(Student.class,1)
        em.getTransaction().commit();
    }
```

Get object

JPA delete objects

SoftUni Foundation

Main.java

```
public static void main(String[] args) {
        EntityManagerFactory emf =
Persistence.createEntityManagerFactory("school");
        EntityManager em = emf.createEntityManager();
        em.getTransaction().begin();
        Student student = em.find(Student.class,1);
        em.remove(student);
        em.getTransaction().commit();
    }
}
```

Remove object

JPA merge objects

- Merges the state of **detached** entity into a **managed copy** of th e detached entity.
  - Returned entity has a different Java identity than the detached one

```
public Student storeUpdatedStudent(Student student) {
  return entityManager.merge(student);
}
```

- May invoke SQL SELECT

Summary

- Maven helps us build our project easily
  - Easy dependency import by XMLs
- Java Persistence API (JPA) is an official standard for Java ORMs
- Hibernate is a widely used Java ORM
  - Implements JPA

Questions?

SoftUni

Software University

SoftUni Svetlina

SoftUni Creative

SoftUni Digital

SoftUni Foundation

SoftUni Kids

SoftUni Diamond Partners

SoftUni Foundation

XSsoftware

SBTech

telenor

SoftwareGroup
doing it right

NETPEAK

SmartIT

Postbank
Решения за твоето утре

SUPER HOSTING .BG

INDEAVR
Serving the high achievers

INFRAGISTICS®

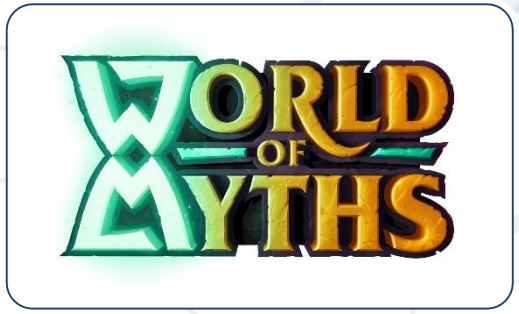
LIEBHERR

æternity

codeXio

SoftUni Organizational Partners

SoftUni Foundation



Trainings @ Software University (SoftUni)

SoftUni Foundation

- Software University – High-Quality Education and Employment Opportunities
  - softuni.bg
- Software University Foundation
  - http://softuni.foundation/
- Software University @ Facebook
  - facebook.com/SoftwareUniversity
- Software University Forums
  - forum.softuni.bg

License

SoftUni Foundation

- This course (slides, examples, demos, videos, homework, etc.) is licensed under the "Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International" license