WEB TECHNOLOGIES USING JAVA

Laboratory 2 - 16.10.2020

AGENDA

- Introduction to Maven
- The Spring Context: Defining beans
  - Adding and retrieving beans from the context
- Homework

Introduction to Maven (I)

- Tool for describing, building, and managing Java software projects using a central piece of information: the Project Object Model (POM)
- Key features:
  - Simple project setup
  - Dependency management: it includes automatic updating, downloading and validating the compatibility
  - Central repository system: local file system or public repositories, such as Maven Central

Introduction to Maven (II)

- Project Identifiers:
  - groupId: a unique base name of the company or group that created the project
  - artifactId: a unique name of the project
  - version: a version of the project
  - packaging: a packaging method (e.g. WAR/JAR/ZIP)
- More information:
  - https://www.baeldung.com/maven
  - https://www.baeldung.com/maven-dependency-scopes

The Spring Context: Defining beans

- **Purpose**: We need to add beans into the Spring context in order for Spring to manage them
- **We can add beans in the context by:**
  - Using the @Bean annotation
  - Using stereotype annotations
  - Programatically

The Spring Context: Defining beans - Prerequisites

- Create a maven project
- Add the spring context dependency from https://mvnrepository.com/

```xml
<groupId>org.example</groupId>
<artifactId>Lab 2</artifactId>
<version>1.0-SNAPSHOT</version>

<properties>
    <maven.compiler.source>11</maven.compiler.source>
    <maven.compiler.target>11</maven.compiler.target>
</properties>

<dependencies>
    <dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-context</artifactId>
        <version>5.2.6.RELEASE</version>
    </dependency>
</dependencies>
```

The Spring Context: Defining beans - **Using the @Bean annotation (I)**

- **Steps:**
  - Define your configuration class - by annotating with @Configuration
  - Define a method that returns the object you need to add on the context
  - Annotate the method with @Bean
  - Make Spring use the @Configuration class

The Spring Context: Defining beans - **Using the @Bean annotation (II)**

```
@Configuration // Define a @Configuration class
public class ApplicationConfiguration {

   @Bean()
   public String greeting() {
       return "Hello Lab2!";
   }
}
```

```
public class Application {
   public static void main(String[] args) {
       // make Spring use the @Configuration class
       AnnotationConfigApplicationContext context = new
AnnotationConfigApplicationContext(ApplicationConfiguration.class);
   }
}
```

The Spring Context: Defining beans - **Using the @Bean annotation (III)**

➢ **Check the object is part of context:**

```
public class Application {
   public static void main(String[] args) {
       AnnotationConfigApplicationContext context = new
AnnotationConfigApplicationContext(ApplicationConfiguration.class);

       String greeting = context.getBean("greeting", String.class);
       System.out.println(greeting);
   }
}
```

The Spring Context: Defining beans - **Using the @Bean annotation (IV)**

- **More examples:**
    - Creating a User object, defining a bean and adding it to the context;
    - Create two more beans of type User. What happens?
    - Defining a bean as “primary” => Spring will select by default this bean; => @Primary

The Spring Context: Defining beans - **Using stereotype annotations (I)**

- **Cannot be used for classes that we can’t change.**
- Steps:
    - Define the object you want to make a bean;
    - Mark the class of the bean with a stereotype annotation (for example, @Component)
    - Add @ComponentScan to the @Configuration class, to instruct Spring where to look for stereotype annotations
    - Make Spring use the @Configuration class

The Spring Context: Defining beans - **Using stereotype annotations (II)**

```
@Configuration()
@ComponentScan(basePackages = "webprogramming.ex5.domain")
public class ApplicationConfiguration {
}
```

```
@Component
public class Cat {
```

```
public static void main(String[] args) {
   AnnotationConfigApplicationContext context = new
AnnotationConfigApplicationContext(ApplicationConfiguration.class);

   Cat catBean = context.getBean(Cat.class);
   System.out.println(catBean.getFur());
}
```

The Spring Context: Defining beans - **Programatically (I)**

- Steps:
  - Use registerBean() from context to register your bean.

```
public static void main(String[] args) {
   AnnotationConfigApplicationContext context = new
AnnotationConfigApplicationContext();

   Cat cat = new Cat("white");
   context.registerBean("myCat", Cat.class, () -> cat);
   context.refresh();

   Cat catBean = context.getBean(Cat.class);
   System.out.println(catBean.getFur());
}
```

Homework

- ➢ Choose a project and define 10 business requirements for the chosen business domain.
- ➢ Prepare a document based on the 10 business requirements containing a description of 4 main features your project should contain for the MVP (minimum viable product) phase.
- ➢ Choose a name for your project and add it in this document: https://docs.google.com/spreadsheets/d/1y7o3pa_3eRckTrA4lHF1AbwcDCz-Qo1G2r_-y3WnKBg/edit?usp=sharing
- ➢ Create a repository for your project and commit the document for review.

THANK YOU

Silvia Butan