Spring Core Workshop
Lab 3

1 LAB SETUP 1

2 JAVA-BASED CONFIGURATION 1

2.1 BASIC CONFIGURATION SETUP WITH @CONFIGURATION AND @COMPONENTSCAN 2
2.2 DEFINING BEANS WITH @BEAN AND RETRIEVING THEM 2
2.3 USING @PRIMARY TO GIVE HIGHER PREFERENCE TO A @BEAN FACTORY METHOD 2
2.4 INJECTING VALUES USING @PROPERTYSOURCE AND @VALUE 3
2.5 INJECTING COLLECTIONS 3
2.6 USING @COMPONENT AND @AUTOWIRED WITH @BEAN 3

1 Lab setup

Make sure you have the following items installed

- Latest version of JDK 8 / 11 (note: labs are tested with JDK 8 but should work on JDK 11 with no or minimal changes)
- Eclipse Enterprise Edition for Java (or a suitable alternative IDE for Enterprise Java)
- Latest version of Maven
- A suitable text editor (Notepad ++)
- A utility to extract zip files

In each of the main lab folders, there are two subfolders: `changes` and `final`. The `changes` subfolder just holds the source code files for the lab, while the `final` subfolder holds the complete Eclipse project starting from its project root folder. We will use the code from the `changes` subfolder to build up our applications from scratch and you can always fall back on the complete Eclipse project if you encounter any errors while building up the application.

2 Java-based configuration

The source code for this lab is found in `Java-Config-DI/changes` folder.

We can create a Maven project from scratch, or we can make a copy from any of the existing Maven projects.
Choose any previous Maven lab project to make a copy from, for e.g.: `XMLConfigConstructorDI`

In the Project Explorer, right click on `XMLConfigConstructorDI`, select Copy and then right click in any empty space in the Explorer and select Paste.

For the new project name, type: `JavaConfigDI`

Replace the contents of the `pom.xml` in the project with `pom.xml` from `changes`. Right click on the project, select Maven -> Update Project, and click OK.

Delete all the packages and files in `src/main/java` and `src/main/resources`. We will start populating the project from scratch.

2.1 Basic configuration setup with @Configuration and @ComponentScan

Create the following classes in `com.workshop.javaconfig`

```
SwimmingExercise.java
Exercise.java
MainConfig.java
JavaConfigDIMainApp.java
```

Open and right click on `JavaConfigDIMainApp` and select Run As -> Java Application. Verify that the correct bean has been created and its output logged to the console correctly.

2.2 Defining beans with @Bean and retrieving them

Make the following changes:

```
JavaConfigDIMainApp-v2.java
MainConfig-v2.java
SwimmingExercise-v2.java

JoggingExercise.java
Student.java
CollegeStudent.java
```

Open and right click on `JavaConfigDIMainApp` and select Run As -> Java Application. Verify that the correct bean has been created and its output logged to the console correctly.

2.3 Using @Primary to give higher preference to a @Bean factory method

Make the following changes:

```
CyclingExercise.java
JavaConfigDIMainApp-v3.java
MainConfig-v3.java
```

Open and right click on `JavaConfigDIMainApp` and select Run As -> Java Application. Verify that the correct bean has been created and its output logged to the console correctly.

Remove the `@Primary` on `public Exercise cyclingExercise()` in `MainConfig` and run the application again. Notice this time an exception is thrown as Spring is unable to determine which implementation for the dependency to use.

2.4 Injecting values using @PropertySource and @Value

Make the following changes:

Place `highschool.properties` in `src/main/resources`

```
HighSchoolStudent.java
MainConfig-v4.java
JavaConfigDIMainApp-v4.java
```

Open and right click on `JavaConfigDIMainApp` and select Run As -> Java Application. Notice that the correct values are read for HighSchoolStudent's fields.

2.5 Injecting collections

Make the following changes:

```
CollegeStudent-v2.java
MainConfig-v5.java
JavaConfigDIMainApp-v5.java
```

Open and right click on `JavaConfigDIMainApp` and select Run As -> Java Application. Notice that the correct values are read for CollegeStudent's collection fields.

2.6 Using @Component and @Autowired with @Bean

Make the following changes:

```
MainConfig-v6.java
JavaConfigDIMainApp-v6.java
HighSchoolStudent-v2.java
CollegeStudent-v3.java
```

Open and right click on `JavaConfigDIMainApp` and select Run As -> Java Application.