Spring Boot Labs

Lab 1 – Basic Spring

- In this Lab, you will finish the wiring up of a Spring application. You will use both XML and annotation based configuration. The end goal is to make a suite of Junit tests run successfully.
- Instructions start on the next page

Lab 1 – Basic Spring

1. Do your work in **Labs/BasicSpringLab.** You may need to import it into your workspace.
2. You may need to set up or configure some Libraries. If you are unsure about how to do this, ask your Instructor.
3. Examine the code. Source code is in **src/main/java**, configuration resources are in **src/main/resources**, and Junit tests are in **src/test/java**.
4. Run any of the **service** tests in **src/test/java** (right click and choose **Run As → Junit Test**)
5. You will find see a whole bunch of errors in the Junit console.
6. Your job is to fix the errors for all the service tests.

Lab 1 – Basic Spring

8. You will **NOT** need to make any changes to the code itself. All your changes will be to Spring configuration elements.
9. You will need to make changes to the Spring configuration. The config class you should use is **ttl.larku.jconfig.LarkUConfig.**
10. You will also need to make annotation based changes to some of the Junit test cases.
11. There are some **TODO** comments in various source files that provide hints about what needs to be done.
12. You will probably need to iterate through a sequence of changes, fixing errors one at a time. In some cases, one fix will cause a bunch of errors to go away.
13. Your goal is to see that lovely green bar indicating a successful Junit test run.
14. A good strategy would be to proceed a test at a time.

Lab 2 – Spring Boot

1) Expose an existing application as a Spring Boot REST application.
2) You have a working Spring Boot application in **Labs/SpringBootStarter.**
3) The application is very simple music playlist manager. The only classes you have right now are **Track, TrackDAO,** and **TrackService**. These are used to manage tracks in a playlist.
4) There is an "application" in **ttl.larku.app.Playlist.java,** and some unit tests. Examine and run the app and the tests so you know how it works

Lab 2 – Spring Boot contd.

2) The Track class implements a Builder pattern so you can create Track objects like this:
   Track.title("Sunrise").artist("Bill Taylor").build();
3) Your job is to write a REST application which will allow for the following:
   a) Getting all Tracks
   b) Getting a Track by Id
   c) Getting a Track by name – may require changes to the business layer.
   d) Creating a new Track
   e) Deleting a Track by Id
   f) Updating a Track

Lab 2 – Spring Boot contd.

3) The easiest way to do this would be to create a new Spring Boot application using the Spring Initializer. You will need the **web** starter at a minumum.
4) You can then copy the code from the **SpringBootStarter** project to your new project.
5) For now, test your controller using a web based REST Client. We will cover Unit/Integration testing and Mock objects later in the course.

The End