**Objectives**

* Demonstrate creation of Spring Boot Application
  + Spring initializr, https://start.spring.io, @SpringBootApplication, SpringApplication.run()
    - Ref - https://start.spring.io
* Explain the need and benefits of Spring Boot
  + Makes Java development easy, avoids tedious development steps, reduces development time, avoids writing boilerplate code, provides embedded tomcat server, avoid XML configuration
    - Ref - https://www.journaldev.com/7969/spring-boot-tutorial

**Hands on 1**

**Create a Spring Web Project using Maven**

Follow steps below to create a project:

1. Go to https://start.spring.io/
2. Change Group as “com.cognizant”
3. Change Artifact Id as “spring-learn”
4. Select Spring Boot DevTools and Spring Web
5. Create and download the project as zip
6. Extract the zip in root folder to Eclipse Workspace

* Build the project using ‘mvn clean package -Dhttp.proxyHost=proxy.cognizant.com -Dhttp.proxyPort=6050 -Dhttps.proxyHost=proxy.cognizant.com -Dhttps.proxyPort=6050 -Dhttp.proxyUser=123456’ command in command line
* Import the project in Eclipse "File > Import > Maven > Existing Maven Projects > Click Browse and select extracted folder > Finish"
* Include logs to verify if main() method of SpringLearnApplication.
* Run the SpringLearnApplication class.

SME to walk through the following aspects related to the project created:

1. src/main/java - Folder with application code
2. src/main/resources - Folder for application configuration
3. src/test/java - Folder with code for testing the application
4. SpringLearnApplication.java - Walkthrough the main() method.
5. Purpose of @SpringBootApplication annotation
6. pom.xml

* Walkthrough all the configuration defined in XML file
* Open 'Dependency Hierarchy' and show the dependency tree.

**Hands-on 2**

**Spring Core – Load Country from Spring Configuration XML**

An airlines website is going to support booking on four countries. There will be a drop down on the home page of this website to select the respective country. It is also important to store the two-character ISO code of each country.

|  |  |
| --- | --- |
| **Code** | **Name** |
| US | United States |
| DE | Germany |
| IN | India |
| JP | Japan |

Above data has to be stored in spring configuration file. Write a program to read this configuration file and display the details.

**Steps to implement**

* Pick any one of your choice country to configure in Spring XML configuration named country.xml.
* Create a bean tag in spring configuration for country and set the property and values

    <bean id="country" class="com.cognizant.springlearn.Country">

        <property name="code" value="IN" />

        <property name="name" value="India" />

    </bean>

* Create Country class with following aspects:
  + Instance variables for code and name
  + Implement empty parameter constructor with inclusion of debug log within the constructor with log message as “Inside Country Constructor.”
  + Generate getters and setters with inclusion of debug with relevant message within each setter and getter method.
  + Generate toString() method // To display object in string format it will gives you string

representation of object.

* Create a method displayCountry() in SpringLearnApplication.java, which will read the country bean from spring configuration file and display the country details.

ClassPathXmlApplicationContext, ApplicationContext and context.getBean(“beanId”, Country.class). Refer sample code for displayCountry() method below.

ApplicationContext context = new ClassPathXmlApplicationContext("country.xml");

Country country = (Country) context.getBean("country", Country.class);

LOGGER.debug("Country : {}", country.toString());

* Invoke displayCountry() method in main() method of SpringLearnApplication.java.
* Execute main() method and check the logs to find out which constructors and methods were invoked.

**SME to provide more detailing about the following aspects:**

* bean tag, id attribute, class attribute, property tag, name attribute, value attribute

<bean id=”e” class=”Employee/> Employee e=new Employee()

* ApplicationContext, ClassPathXmlApplicationContext
* What exactly happens when context.getBean() is invoked

**Troubleshooting Tips**

If the tomcat port has a conflict and the server is not starting include the below property in application.properties file in src/main/resources folder.

**Hands-on 3**

**Spring Core – Demonstration of Singleton Scope and Prototype Scope**

The Country bean done in the previous hands on will be used to demonstrate the scopes in Spring. Implement the steps below.

**Follow steps below to demonstrate Singleton Scope**

* Include a line in displayCountry() to get country bean reference one more time from the same application context. Only the third line of the below code snippet should be copied and pasted.

ApplicationContext context = new ClassPathXmlApplicationContext("country.xml");

Country country = context.getBean("country", Country.class);

Country anotherCountry = context.getBean("country", Country.class);

* The constructor will be called only once, which means that only one instance of Country bean is created

**Follow steps below to demonstrate Prototype Scope**

* Include scope="prototype" attribute in bean definition xml.

<bean id="country" class="com.cognizant.springlearn.Country" scope="prototype">

* Run the application
* Constructor will be called twice, which means that two instances of country is created.

**Hands-on 4**

**Spring Core – Load list of countries from Spring Configuration XML**

Our main objective was to retrieve the list of four countries for the airlines website. Refer steps below to get this incorporated.

* Create a separate bean for each of the four country in country.xml.
* Create an ArrayList of Country in country.xml. Refer code below.

    <bean id="countryList" class="java.util.ArrayList">

        <constructor-arg>

            <list>

                <ref bean="in"></ref>

                <ref bean="us"></ref>

                <ref bean="de"></ref>

                <ref bean="jp"></ref>

            </list>

        </constructor-arg>

    </bean>

* Include new method displayCountries() in SpringLearnApplication.java
* In displayCountries() read the country list created above
* Display the list of countries as debug log.

SME to provide detailing on below aspects:

* <list>
* <ref>
* bean attribute

**IMPORTANT NOTE**: Do not forget to include the start and end logs in this new method.