**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**
7. **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**
8. **Import the project in Eclipse "File > Import > Maven > Existing Maven Projects > Click Browse and select extracted folder > Finish"**
9. **Include logs to verify if main() method of SpringLearnApplication.**
10. **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**
   1. **Walkthrough all the configuration defined in XML file**
   2. **Open 'Dependency Hierarchy' and show the dependency tree.**

**pom.xml**

<!-- File: 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/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<parent>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-parent</artifactId>

<version>3.2.4</version>

<relativePath/>

</parent>

<groupId>com.cognizant</groupId>

<artifactId>spring-learn</artifactId>

<version>0.0.1-SNAPSHOT</version>

<name>spring-learn</name>

<description>Demo project for Spring Boot</description>

<packaging>jar</packaging>

<properties>

<java.version>17</java.version>

</properties>

<dependencies>

<!-- Spring Web -->

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-web</artifactId>

</dependency>

<!-- Spring DevTools -->

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-devtools</artifactId>

<scope>runtime</scope>

<optional>true</optional>

</dependency>

<!-- Testing -->

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-test</artifactId>

<scope>test</scope>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-maven-plugin</artifactId>

</plugin>

</plugins>

</build>

</project>

**SpringLearnApplication.java**

// File: src/main/java/com/cognizant/SpringLearnApplication.java

package com.cognizant;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class SpringLearnApplication {

public static void main(String[] args) {

System.out.println("Starting Spring Boot Application...");

SpringApplication.run(SpringLearnApplication.class, args);

System.out.println("Application Started Successfully!");

}

}

**application.properties**

# File: src/main/resources/application.properties

# Empty for now — configurations go here later.

**SpringLearnApplicationTests.java**

package com.cognizant;

import org.junit.jupiter.api.Test;

import org.springframework.boot.test.context.SpringBootTest;

@SpringBootTest

class SpringLearnApplicationTests {

@Test

void contextLoads() {

}

}

**HelloController.java**

// File: src/main/java/com/cognizant/controller/HelloController.java

package com.cognizant.controller;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.RestController;

@RestController

public class HelloController {

@GetMapping("/")

public String sayHello() {

return "Hello from Spring Boot!";

}

}

**OUTPUT:**

**A screen shot of a computer

AI-generated content may be incorrect.**

**Hands on 4**

**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.

**Country.java**

package com.cognizant.springlearn;

public class Country {

private String code;

private String name;

public Country() {

System.out.println("Inside Country Constructor.");

}

public String getCode() {

System.out.println("Inside getCode()");

return code;

}

public void setCode(String code) {

System.out.println("Inside setCode()");

this.code = code;

}

public String getName() {

System.out.println("Inside getName()");

return name;

}

public void setName(String name) {

System.out.println("Inside setName()");

this.name = name;

}

@Override

public String toString() {

return "Country [code=" + code + ", name=" + name + "]";

}

}

**SpringLearnApplication.java**

package com.cognizant.springlearn;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

public class SpringLearnApplication {

public static void main(String[] args) {

displayCountry();

}

public static void displayCountry() {

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

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

System.out.println("Country: " + country.toString());

}

}

**country.xml**

<?xml version="1.0" encoding="UTF-8"?>

<beans xmlns="http://www.springframework.org/schema/beans"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://www.springframework.org/schema/beans

https://www.springframework.org/schema/beans/spring-beans.xsd">

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

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

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

</bean>

</beans>

**OUTPUT:**

**A screenshot of a computer program

AI-generated content may be incorrect.**