**Week 4 (1)**

**Ex-1**

**Create a Spring Web Project using Maven**

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.

**Work:**

**SpringLearnApplication.java:**

package com.cognizant.springlearn;

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

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

@RestController

public class HelloController {

@GetMapping("/")

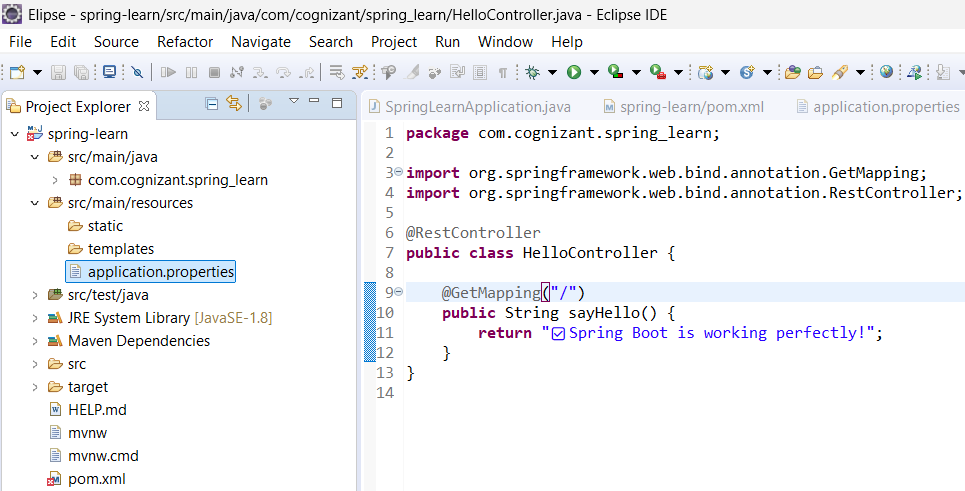
public String sayHello() {

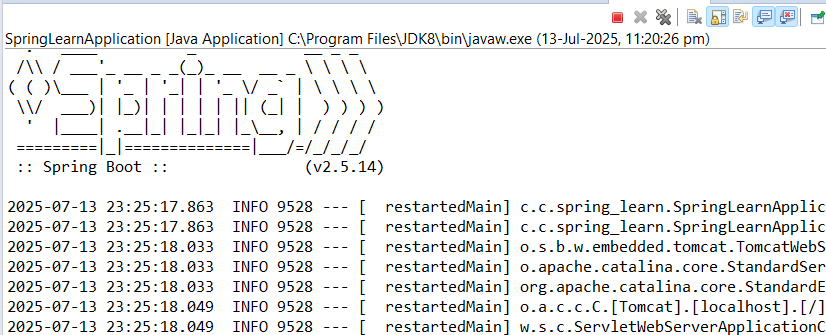
return "✅ Spring Boot is working perfectly!";

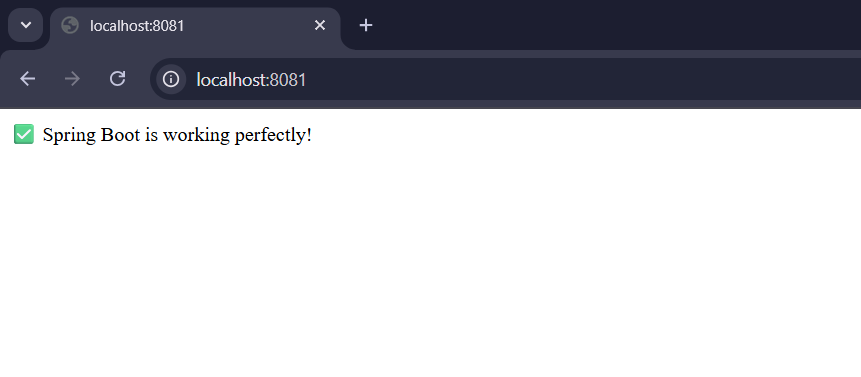
}

}

**Output:**

****

****

****

**EX 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.**

**Country.java**

package com.cognizant.spring\_learn;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

public class Country {

private static final Logger LOGGER = LoggerFactory.getLogger(Country.class);

private String code;

private String name;

public Country() {

LOGGER.debug("Inside Country Constructor.");

}

public String getCode() {

LOGGER.debug("Getting code");

return code;

}

public void setCode(String code) {

LOGGER.debug("Setting code");

this.code = code;}

public String getName() {

LOGGER.debug("Getting name");

return name;

}

public void setName(String name) {

LOGGER.debug("Setting name");

this.name = name;

}

public String toString() {

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

} }

**SpringLearnApplication.java**

package com.cognizant.spring\_learn;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

public class SpringLearnApplication {

private static final Logger LOGGER = LoggerFactory.getLogger(SpringLearnApplication.class);

public static void main(String[] args) {

LOGGER.info("START");

displayCountry();

LOGGER.info("END");

}

public static void displayCountry() {

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

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

LOGGER.debug("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

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

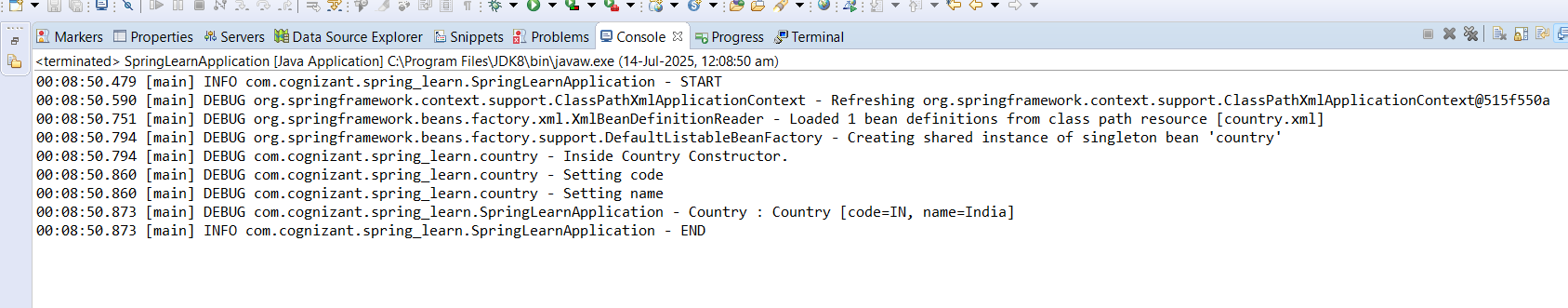
<bean id="country" class="com.cognizant.spring\_learn.Country">

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

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

</bean>

</beans>



| **Element** | **Description** |
| --- | --- |
| <bean> | Used to define a Spring-managed object (a bean). It tells Spring to create and manage the lifecycle of the specified class. |
| id | A unique identifier for the bean. Used to refer to the bean later using getBean("id"). Example: id="country" |
| class | Fully qualified class name that Spring should instantiate. Example: class="com.cognizant.spring\_learn.Country" |
| <property> | Used to inject values into the bean’s fields via setter methods. |
| name | The name of the property (field) in the Java class. It must match the setter name (e.g., setCode() → name="code"). |
| value | The actual value to be injected. Can be a string, number, or reference to another bean. Example: value="IN" |

1. Spring searches for a <bean> with id="country" in the loaded XML.
2. It creates an instance of com.cognizant.spring\_learn.Country.
3. Spring calls the default (no-arg) constructor.
4. It then injects property values (code, name) via setter methods.
5. The fully initialized bean is returned to you.

This is called **Dependency Injection** and is a core feature of the Spring Framework.