Hands-on 4: Spring Core – Load Country from Spring Configuration XML

# 1. Objective

The goal of this hands-on exercise is to create a Spring application that loads a country bean from an XML configuration file. This bean will be used to display the country code and name using the Spring Framework's ApplicationContext.

# 2. Requirements

Store country data using Spring XML configuration and display it via a Java application.

# 3. Steps to Implement

• Create an XML configuration file named country.xml in src/main/resources.

• Define a <bean> with id='country' and class='com.cognizant.springlearn.Country'.

• Set <property> values: code='IN', name='India'.

• Create a Country class with instance variables, constructor, getters, setters, and toString().

• Add logging in each method to observe Spring behavior.

• Use ClassPathXmlApplicationContext to load the configuration file and fetch the bean.

• Display the bean values using LOGGER.debug().

• Invoke the method from the main() method of SpringLearnApplication.

# 4. Spring Configuration - country.xml

<bean id="country" class="com.cognizant.springlearn.Country">  
 <property name="code" value="IN"/>  
 <property name="name" value="India"/>  
</bean>

# 5. Country.java

• Has two fields: code and name.

• Contains a no-argument constructor with debug log: 'Inside Country Constructor.'

• Getters and setters include logging to trace method calls.

• Overrides toString() for better logging output.

# 6. SpringLearnApplication.java

• Includes a static method displayCountry() that:

- Creates ApplicationContext using ClassPathXmlApplicationContext.

- Reads the 'country' bean and logs it using LOGGER.debug().

• displayCountry() is called from the main() method.

# 7. Key Spring Concepts

• <bean>: Defines a Spring-managed Java object (bean).

• id: Unique identifier for the bean.

• class: Fully qualified class name of the bean.

• <property>: Used to inject values into fields of a bean.

• name: Refers to the property in the bean class.

• value: The value to be injected.

• ApplicationContext: Central interface for Spring configuration.

• ClassPathXmlApplicationContext: Loads configuration from classpath.

• context.getBean(): Fetches bean instance using bean id and type.

# 8. Execution and Logs

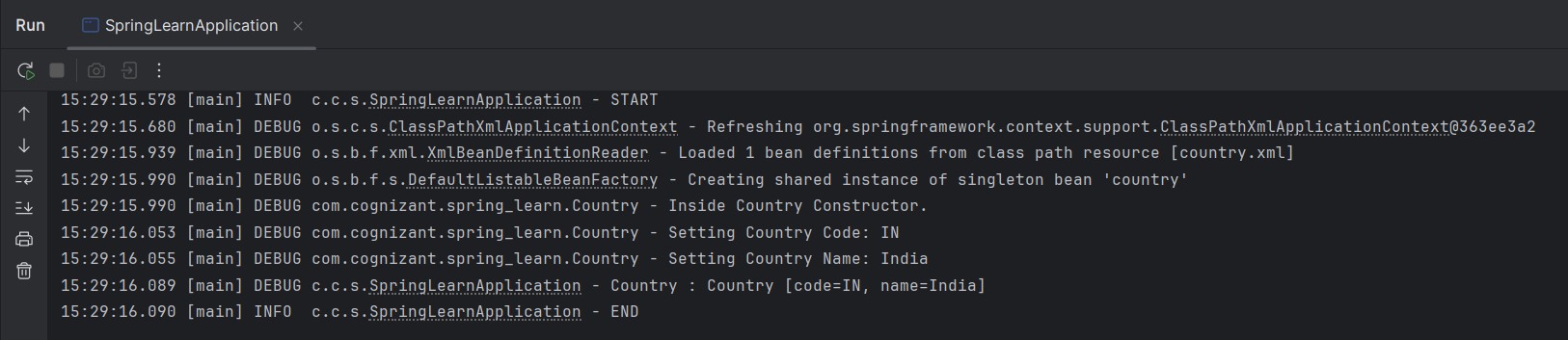
On running the application, logs are displayed showing:

• Country constructor invoked.

• Setters invoked with values 'IN' and 'India'.

• Output of the Country object via toString().

**9. Output**



# 9. Conclusion

This hands-on demonstrates how to use Spring XML configuration to manage and initialize beans. It also shows how to access and display the bean data using the Spring container.