**SpringCore Load Country From Spring Configuration XML**

**1. <bean> Tag and Attributes**

The <bean> tag in Spring XML is used to define a Java object (the bean) managed by the Spring container.

Attributes and tags under <bean>:

id: A unique name given to the bean for later reference and retrieval.

class: The fully qualified name of the Java class that is instantiated.

<property>: Setter injection to set values or inject dependencies to the bean.

name: The actual name of the property in the Java class (must have a setter method).

value: The actual value to be assigned to the property.

Example:

<bean id="in" class="com.cognizant.spring-learn.model.Country">

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

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

</bean>

This defines a Country bean, assigning "IN" to its property code and "India" to its property name.

**2. ApplicationContext and ClassPathXmlApplicationContext**

ApplicationContext: Main interface to access Spring beans and provide configuration. It is responsible for the entire lifecycle and dependencies of all beans.

ClassPathXmlApplicationContext: Concrete implementation of ApplicationContext that reads Spring configuration from the classpath (generally, from src/main/resources).

Example:

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

Country country = (Country) context.getBean("in").

This loads XML file country.xml, creates the Country bean and returns it using "in" as identification.

**3. What happens when context.getBean() is called?**

When context.getBean("in") is called:

Spring looks for a bean with id "in" in the XML configuration.

If the bean has not been created yet:

It creates an instance of the class defined (com.cognizant.spring-learn.model.Country).

It sets the properties using the <property> values (setCode("IN"), setName("India")).

The fully initialized bean is returned.

If the bean already exists (singleton by default), it simply returns the same instance.