## **1. <bean> tag and its attributes:**

In country.xml, Spring beans are defined using the <bean> tag.

**Example Code:**

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

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

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

</bean>

**Explanation:**

**<bean>**: Declares a Spring-managed component (i.e., object/instance).

**id attribute**: The name used to refer to this bean (e.g., "in").

**class attribute**: Fully qualified class name (FQCN) of the Java object (e.g., com.cognizant.spring\_learn.model.Country).

**<property> tag**: Sets values for the bean’s fields via setter methods.

**name**: The name of the JavaBean property (must match the setter, e.g., setName(String)).

**value**: The actual value assigned to that property (e.g., "India").

## **2. ApplicationContext and ClassPathXmlApplicationContext:**

* **ApplicationContext**: Spring's central interface for accessing the container.
  + Manages bean lifecycle, dependency injection, and configuration.
  + It's more powerful than the older BeanFactory.
* **ClassPathXmlApplicationContext**:
  + A concrete implementation of ApplicationContext.
  + Loads bean definitions from an XML file in the **classpath**.

**Example Code**:

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

* This loads country.xml from src/main/resources (or any directory on classpath).
* It parses the XML, creates all beans declared in it, and stores them in memory.

## **3. What happens when context.getBean("in") is invoked?**

**When you call:**

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

Here's what happening step-by-step:

1. **Lookup**: Spring searches the container for a bean with ID "in".
2. **Type Matching**: It matches the type (Country) and ensures the class exists.
3. **Instantiation (if not already done)**:
   * If the bean is **singleton** (default scope), it's already created at startup.
   * If it’s **prototype**, a **new instance** is created.
4. **Dependency Injection**: Spring injects any dependencies or properties defined via <property>.
5. **Returns the object**: The fully constructed and initialized bean is returned to you.