**Spring Core – Load List of Countries from Spring Configuration XML**

This assignment demonstrates how to use Spring Core to load a list of countries using XML configuration. Individual country beans were defined in country.xml and grouped into a list using <list> and <ref>. The displayCountries() method in SpringLearnApplication.java retrieves and logs this list, showcasing Spring's dependency injection for collections.

1. **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">

<!-- Individual Country Beans -->

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

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

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

</bean>

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

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

<property name="name" value="United States"/>

</bean>

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

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

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

</bean>

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

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

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

</bean>

<!-- List of Countries -->

<bean id="countryList" class="java.util.ArrayList">

<constructor-arg>

<list>

<ref bean="in"/>

<ref bean="us"/>

<ref bean="de"/>

<ref bean="jp"/>

</list>

</constructor-arg>

</bean>

</beans>

**2. 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("Inside getCode");

return code;

}

public void setCode(String code) {

***LOGGER***.debug("Inside setCode");

this.code = code;

}

public String getName() {

***LOGGER***.debug("Inside getName");

return name;

}

public void setName(String name) {

***LOGGER***.debug("Inside setName");

this.name = name;

}

*@Override*

public String toString() {

return "Country{" +

"code='" + code + '\'' +

", name='" + name + '\'' +

'}';

}

}

**3. SpringLearnApplication.java**

package com.cognizant.spring\_learn;

import java.util.List;

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) {

*displayCountries*();

}

public static void displayCountries() {

***LOGGER***.info("START - displayCountries");

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

*@SuppressWarnings*("unchecked")

List<Country> countryList = (List<Country>) context.getBean("countryList");

for (Country country : countryList) {

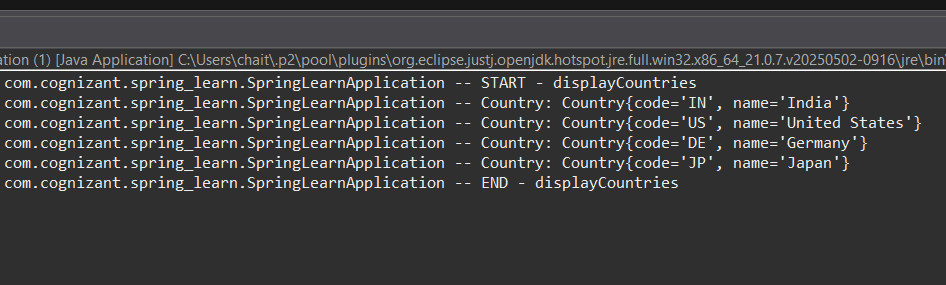
***LOGGER***.info("Country: {}", country);

}

***LOGGER***.info("END - displayCountries");

}

}

**4. Output Screenshot (Observed Output)**

**5. XML Tag Explanations**

* **<list>**:  
  The <list> tag is used in Spring XML configuration to define a java.util.List collection. It allows us to group multiple elements, such as values or references to other beans, in a specific order. In this assignment, <list> was used to create a list of Country beans, which were then injected into the countryList bean. This enables centralized collection management directly from the configuration file without needing manual instantiation in code.
* **<ref>**:  
  The <ref> tag is used to reference another bean that is defined elsewhere in the same or different Spring XML configuration file. Instead of duplicating bean definitions, Spring allows you to reuse existing beans using <ref>. For example, <ref bean="in"/> refers to the country bean with ID "in" (India). This promotes reuse and reduces redundancy in configuration.
* **bean attribute**:  
  The bean attribute is used in different contexts:
  + Inside the <bean> tag, it is used to **define a bean** by assigning it a unique id and specifying the class it represents.
  + Inside the <ref> tag, it is used to **refer to an existing bean** by its id.  
    For example:
  + <bean id="in" class="com.cognizant.spring\_learn.Country"/> defines a new bean.
  + <ref bean="in"/> refers to the previously defined in bean.