Name:Duggirala Bhagya Vivekini

Superset ID: 5673888

Cognizant Digital-Nurture-4.0-JavaFSE- Handson WEEK-4

Create a Spring Web Project using Maven  
1. pom.xml :  
<?xml version="1.0" encoding="UTF-8"?>

<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<parent>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-parent</artifactId>

<version>2.7.18</version>

<relativePath/>

</parent>

<groupId>com.cognizant</groupId>

<artifactId>spring-learn</artifactId>

<version>0.0.1-SNAPSHOT</version>

<name>spring-learn</name>

<description>Demo project for Spring Boot</description>

<properties>

<java.version>1.8</java.version>

</properties>

<dependencies>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-web</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-devtools</artifactId>

<scope>runtime</scope>

<optional>true</optional>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-test</artifactId>

<scope>test</scope>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-maven-plugin</artifactId>

</plugin>

</plugins>

</build>

</project>  
  
2. SpringLearnApplication.java

package com.cognizant.springlearn;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class SpringLearnApplication {

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

public static void main(String[] args) {

LOGGER.info("Inside main");

SpringApplication.run(SpringLearnApplication.class, args);

LOGGER.info("End of main");

}

}

3. application.properties

# No specific properties required for this hands-on.

OUTPUT:

YYYY-MM-DD HH:MM:SS.SSS main INFO com.cognizant.springlearn.SpringLearnApplication main 29 Inside main ...

(Spring Boot startup logs will appear here) ...

YYYY-MM-DD HH:MM:SS.SSS main INFO com.cognizant.springlearn.SpringLearnApplication main 32 End of main

Spring Core – Load Country from Spring Configuration XML  
1. Country.java  
package com.cognizant.springlearn.model;

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 Country getter - getCode()");

return code;

}

public void setCode(String code) {

LOGGER.debug("Inside Country setter - setCode()");

this.code = code;

}

public String getName() {

LOGGER.debug("Inside Country getter - getName()");

return name;

}

public void setName(String name) {

LOGGER.debug("Inside Country setter - setName()");

this.name = name;

}

@Override

public String toString() {

return "Country{" +

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

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

'}';

}

}

2. 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">

<bean id="country" class="com.cognizant.springlearn.model.Country">

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

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

</bean>

</beans>

3. SpringLearnApplication.java

package com.cognizant.springlearn;

import com.cognizant.springlearn.model.Country;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

@SpringBootApplication

public class SpringLearnApplication {

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

public static void main(String[] args) {

LOGGER.info("Inside main");

SpringApplication.run(SpringLearnApplication.class, args);

LOGGER.info("End of main");

displayCountry();

}

public static void displayCountry() {

LOGGER.info("START displayCountry()");

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

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

LOGGER.debug("Country : {}", country.toString());

LOGGER.info("END displayCountry()");

}

}

4. application.properties

logging.level.org.springframework=info

logging.level.com.cognizant.springlearn=debug

logging.pattern.console=%d{yyMMdd}|%d{HH:mm:ss.SSS}|%-20.20thread|%5p|%-25.25logger{25}|%25M|%m%n

OUTPUT:

... (Spring Boot startup logs) ...

250711|17:00:00.123|main | INFO|c.c.s.SpringLearnApplication | main 30 Inside main

250711|17:00:00.456|main | INFO|c.c.s.SpringLearnApplication | displayCountry 60 START displayCountry()

250711|17:00:00.789|main |DEBUG|c.c.s.model.Country | <init> 20 Inside Country Constructor.

250711|17:00:01.012|main |DEBUG|c.c.s.model.Country | setCode 28 Inside Country setter - setCode()

250711|17:00:01.123|main |DEBUG|c.c.s.model.Country | setName 38 Inside Country setter - setName()

250711|17:00:01.234|main |DEBUG|c.c.s.model.Country | toString 49 Country : Country{code='IN', name='India'}

250711|17:00:01.345|main | INFO|c.c.s.SpringLearnApplication | displayCountry 63 END displayCountry()

250711|17:00:01.456|main | INFO|c.c.s.SpringLearnApplication | main 33 End of main

Hello World RESTful Web Service

HelloController.java  
package com.cognizant.springlearn.controller;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

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

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

@RestController

public class HelloController {

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

public HelloController() {

LOGGER.debug("Inside HelloController Constructor");

}

@GetMapping("/hello")

public String sayHello() {

LOGGER.info("START sayHello()");

LOGGER.info("END sayHello()");

return "Hello World!";

}

}

OUTPUT:

Hello World!

REST - Country Web Service

CountryController.java

package com.cognizant.springlearn.controller;

import com.cognizant.springlearn.model.Country;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

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

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

@RestController

public class CountryController {

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

public CountryController() {

LOGGER.debug("Inside CountryController Constructor");

}

@GetMapping("/country")

public Country getCountry() {

LOGGER.info("START getCountry()");

Country country = new Country();

country.setCode("IN");

country.setName("India");

LOGGER.info("END getCountry()");

return country;

}

}

OUTPUT:  
{"code":"IN","name":"India"}

REST - Get country based on country code

Country.java

package com.cognizant.springlearn.model;

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 Country getter - getCode()");

return code;

}

public void setCode(String code) {

LOGGER.debug("Inside Country setter - setCode()");

this.code = code;

}

public String getName() {

LOGGER.debug("Inside Country getter - getName()");

return name;

}

public void setName(String name) {

LOGGER.debug("Inside Country setter - setName()");

this.name = name;

}

@Override

public String toString() {

return "Country{" +

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

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

'}';

}

}

country-list.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">

<bean id="in" class="com.cognizant.springlearn.model.Country">

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

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

</bean>

<bean id="us" class="com.cognizant.springlearn.model.Country">

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

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

</bean>

<bean id="de" class="com.cognizant.springlearn.model.Country">

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

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

</bean>

<bean id="jp" class="com.cognizant.springlearn.model.Country">

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

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

</bean>

<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>

CountryService.java

package com.cognizant.springlearn.service;

import com.cognizant.springlearn.model.Country;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

import org.springframework.stereotype.Service; // New import

import java.util.List; // New import

@Service // Annotate as a Spring Service

public class CountryService {

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

private List<Country> countryList; // Declare a list to hold countries

public CountryService() {

LOGGER.debug("Inside CountryService Constructor");

// Load the country list when the service is created

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

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

}

public List<Country> getAllCountries() {

LOGGER.info("START getAllCountries()");

LOGGER.info("END getAllCountries()");

return countryList;

}

// New method to get country by code

public Country getCountryByCode(String code) {

LOGGER.info("START getCountryByCode()");

for (Country country : countryList) {

if (country.getCode().equalsIgnoreCase(code)) {

LOGGER.info("END getCountryByCode()");

return country;

}

}

LOGGER.info("END getCountryByCode() - Country not found");

return null; // Or throw an exception if country is not found

}

}

CountryController.java

package com.cognizant.springlearn.controller;

import com.cognizant.springlearn.model.Country;

import com.cognizant.springlearn.service.CountryService; // New import for CountryService

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.beans.factory.annotation.Autowired; // New import

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

import org.springframework.web.bind.annotation.PathVariable; // New import

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

import java.util.List; // New import if you plan to return all countries from controller

@RestController

public class CountryController {

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

@Autowired // Inject CountryService

private CountryService countryService; // Declare CountryService

public CountryController() {

LOGGER.debug("Inside CountryController Constructor");

}

// Existing endpoint from "REST - Country Web Service"

@GetMapping("/country")

public Country getCountry() {

LOGGER.info("START getCountry()");

// This method can now return a specific country from the service if needed,

// or remain hardcoded as per the original hands-on.

Country country = new Country();

country.setCode("IN");

country.setName("India");

LOGGER.info("END getCountry()");

return country;

}

// New endpoint to get all countries (often useful with a list)

@GetMapping("/countries")

public List<Country> getAllCountries() {

LOGGER.info("START getAllCountries()");

List<Country> countries = countryService.getAllCountries();

LOGGER.info("END getAllCountries()");

return countries;

}

// New endpoint for "REST - Get country based on country code"

@GetMapping("/country/{code}") // Map GET requests to /country/{code}

public Country getCountryByCode(@PathVariable("code") String code) {

LOGGER.info("START getCountryByCode() with code: {}", code);

Country country = countryService.getCountryByCode(code);

LOGGER.info("END getCountryByCode()");

return country;

}

}

OUTPUT:

Access <http://localhost:8080/country/in>  
{"code":"IN","name":"India"}

Access <http://localhost:8080/country/us>

{"code":"US","name":"United States"}

Access <http://localhost:8080/country/xyz>

(This would typically return an empty JSON response or a 500 error if no exception handling is implemented. In a real app, you'd return 404 Not Found.)