**Mandatory Hands On**

**Hello World RESTful Web Service**

package com.cognizant.springlearn;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

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

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

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

@SpringBootApplication

public class SpringLearnApplication {

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

public static void main(String[] args) {

LOGGER.info("START");

SpringApplication.run(SpringLearnApplication.class, args);

LOGGER.info("END");

}

@RestController

public static class HelloController {

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

@GetMapping("/hello")

public String sayHello() {

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

String message = "Hello World!!";

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

return message;

}

}

}

**Output**

When you access: http://localhost:8083/hello

Browser / Postman Response:

Hello World!!

**Console Output**

INFO com.cognizant.springlearn.SpringLearnApplication - START

INFO com.cognizant.springlearn.SpringLearnApplication - END

INFO com.cognizant.springlearn.SpringLearnApplication$HelloController - START - sayHello()

INFO com.cognizant.springlearn.SpringLearnApplication$HelloController - END - sayHello()

**SME Explanation:**

**HTTP Headers**

**In Chrome Developer Tools:**

1. Open browser and visit: http://localhost:8083/hello
2. Press F12 to open Developer Tools
3. Go to Network tab
4. Click on the /hello request
5. Under **Headers** tab, you’ll see:

**Request Headers:**

GET /hello HTTP/1.1

Host: localhost:8083

User-Agent: Mozilla/5.0 ...

Accept: text/html,application/xhtml+xml,...

**Response Headers:**

HTTP/1.1 200

Content-Type: text/plain;charset=UTF-8

Content-Length: 13

Date: Thu, 11 Jul 2025 14:00:00 GMT

**In Postman:**

1. Open Postman → New Request
2. Method: GET
3. URL: http://localhost:8083/hello
4. Click Send
5. Go to Headers tab

**Response Headers:**

| **Key** | **Value** |
| --- | --- |
| Content-Type | text/plain;charset=UTF-8 |
| Content-Length | 13 |
| Date | Thu, 11 Jul 2025 14:00:00 GMT |
|  |  |

**REST - Country Web Service**

package com.cognizant.springlearn;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

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

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

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

@SpringBootApplication

public class SpringLearnApplication {

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

public static void main(String[] args) {

LOGGER.info("START");

SpringApplication.run(SpringLearnApplication.class, args);

LOGGER.info("END");

}

@RestController

public static class CountryController {

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

@RequestMapping("/country")

public Country getCountryIndia() {

LOGGER.info("START - getCountryIndia()");

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

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

LOGGER.info("END - getCountryIndia()");

return country;

}

}

public static class Country {

private String code;

private String name;

public Country() {

LoggerFactory.getLogger(Country.class).debug("Inside Country Constructor.");

}

public String getCode() {

LoggerFactory.getLogger(Country.class).debug("Inside getCode");

return code;

}

public void setCode(String code) {

LoggerFactory.getLogger(Country.class).debug("Inside setCode");

this.code = code;

}

public String getName() {

LoggerFactory.getLogger(Country.class).debug("Inside getName");

return name;

}

public void setName(String name) {

LoggerFactory.getLogger(Country.class).debug("Inside setName");

this.name = name;

}

@Override

public String toString() {

return "Country [code=" + code + ", name=" + name + "]";

}

}

}

**Spring XML Configuration: country.xml**

Path: src/main/resources/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 http://www.springframework.org/schema/beans/spring-beans.xsd">

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

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

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

</bean>

</beans>

**Sample Output**

**URL:**

bash

Copy code

http://localhost:8083/country

**Browser / Postman Response:**

json

Copy code

{

"code": "IN",

"name": "India"

}

**Output**

INFO com.cognizant.springlearn.SpringLearnApplication - START

INFO com.cognizant.springlearn.SpringLearnApplication - END

DEBUG com.cognizant.springlearn.SpringLearnApplication$Country - Inside Country Constructor.

DEBUG com.cognizant.springlearn.SpringLearnApplication$Country - Inside setCode

DEBUG com.cognizant.springlearn.SpringLearnApplication$Country - Inside setName

INFO com.cognizant.springlearn.SpringLearnApplication$CountryController - START - getCountryIndia()

DEBUG com.cognizant.springlearn.SpringLearnApplication$Country - Inside getCode

DEBUG com.cognizant.springlearn.SpringLearnApplication$Country - Inside getName

INFO com.cognizant.springlearn.SpringLearnApplication$CountryController - END - getCountryIndia()

**SME EXPLANATION**

**What Happens in the Controller?**

* @RequestMapping("/country") handles GET requests.
* ApplicationContext loads country.xml.
* context.getBean("country", Country.class) retrieves the configured Country object.
* Spring uses Jackson (default) to convert the Country object into JSON.

**How the Bean is Converted to JSON?**

* Spring Boot includes **Jackson JSON processor** by default.
* Any object returned from a @RestController method is **automatically converted to JSON**.
* Getters (getCode, getName) are used to create the JSON structure.

**In Chrome DevTools (F12 > Network > /country):**

**Request Headers:**

GET /country HTTP/1.1

Host: localhost:8083

Accept: application/json

User-Agent: Mozilla/5.0

**Response Headers:**

HTTP/1.1 200

Content-Type: application/json

Content-Length: 34

**In Postman → Headers tab:**

**Response Headers Table:**

| **Key** | **Value** |
| --- | --- |
| Content-Type | application/json |
| Content-Length | 34 |
| Date | Thu, 11 Jul 2025 15:10:00 GMT |

**REST - Get country based on country code** 

package com.cognizant.springlearn;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

import org.springframework.web.bind.annotation.\*;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import java.util.List;

@SpringBootApplication

public class SpringLearnApplication {

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

public static void main(String[] args) {

LOGGER.info("START");

SpringApplication.run(SpringLearnApplication.class, args);

LOGGER.info("END");

}

@RestController

@RequestMapping("/countries")

public static class CountryController {

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

private final CountryService countryService = new CountryService();

@GetMapping("/{code}")

public Country getCountry(@PathVariable String code) {

LOGGER.info("START - getCountry({})", code);

Country country = countryService.getCountry(code);

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

return country;

}

}

public static class CountryService {

public Country getCountry(String code) {

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

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

return countryList.stream()

.filter(c -> c.getCode().equalsIgnoreCase(code))

.findFirst()

.orElse(null);

}

}

public static class Country {

private String code;

private String name;

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

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 + "]";

}

}

}

**Spring XML Config: 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 http://www.springframework.org/schema/beans/spring-beans.xsd">

<bean id="country1" class="com.cognizant.springlearn.SpringLearnApplication$Country">

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

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

</bean>

<bean id="country2" class="com.cognizant.springlearn.SpringLearnApplication$Country">

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

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

</bean>

<bean id="country3" class="com.cognizant.springlearn.SpringLearnApplication$Country">

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

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

</bean>

<bean id="country4" class="com.cognizant.springlearn.SpringLearnApplication$Country">

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

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

</bean>

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

<constructor-arg>

<list>

<ref bean="country1" />

<ref bean="country2" />

<ref bean="country3" />

<ref bean="country4" />

</list>

</constructor-arg>

</bean>

</beans>

**Port Configuration**

Path: src/main/resources/application.properties

server.port=8083

**Sample Request**

**URL:**

http://localhost:8083/countries/in

**Sample Response**

{

"code": "IN",

"name": "India"

}

**Output**

INFO com.cognizant.springlearn.SpringLearnApplication - START

INFO com.cognizant.springlearn.SpringLearnApplication - END

DEBUG com.cognizant.springlearn.SpringLearnApplication$Country - Inside Country Constructor.

DEBUG com.cognizant.springlearn.SpringLearnApplication$Country - Inside setCode

DEBUG com.cognizant.springlearn.SpringLearnApplication$Country - Inside setName

... (repeated for each country)

INFO com.cognizant.springlearn.SpringLearnApplication$CountryController - START - getCountry(in)

DEBUG com.cognizant.springlearn.SpringLearnApplication$Country - Inside getCode

INFO com.cognizant.springlearn.SpringLearnApplication$CountryController - END - getCountry()

**SME Explanation**

**What happens in the controller?**

* URL /countries/in hits getCountry(String code)
* code = "in" (from @PathVariable)
* It calls countryService.getCountry(code)
* Bean list from country.xml is loaded
* A stream filter is used to find the matching country
* JSON of that country is returned

**How the bean is converted to JSON?**

* Spring Boot uses **Jackson** internally
* The object returned from controller is **automatically serialized** into JSON
* Getters (getCode, getName) are used to construct JSON

**In Chrome DevTools → Network → /countries/in**

**Request Headers:**

GET /countries/in HTTP/1.1

Host: localhost:8083

Accept: application/json

**Response Headers:**

Content-Type: application/json

Content-Length: 34

**In Postman → Headers Tab**

| **Key** | **Value** |
| --- | --- |
| Content-Type | application/json |
| Content-Length | 34 |