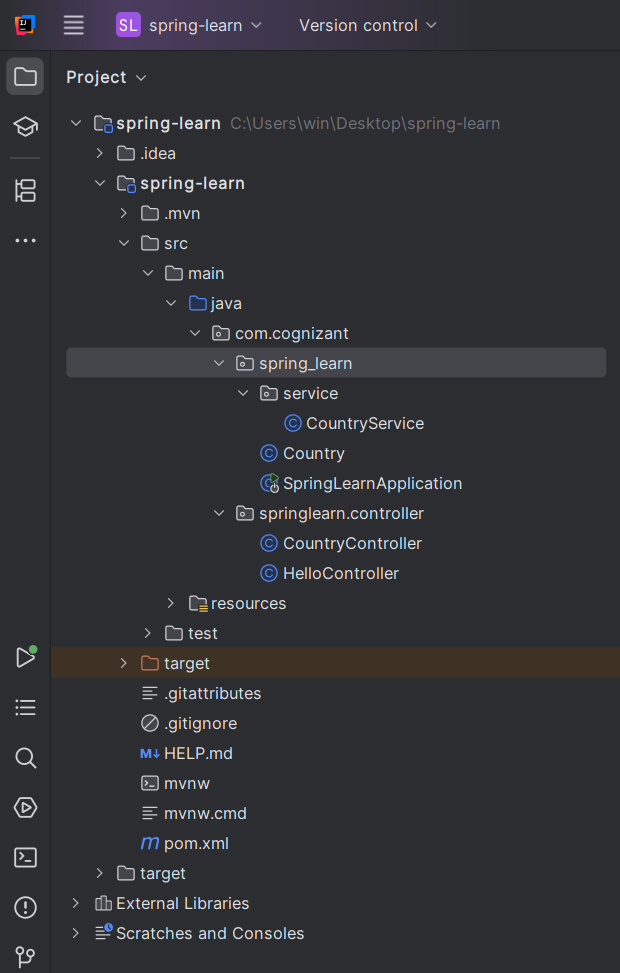
**Hello World RESTful Web Service**:

File Hierarchy



STEP 1: Add spring-boot-starter-web Dependency (if not present)

In pom.xml:

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  
 http://maven.apache.org/xsd/maven-4.0.0.xsd">  
  
 <modelVersion>4.0.0</modelVersion>  
 <groupId>com.cognizant</groupId>  
 <artifactId>spring-learn</artifactId>  
 <version>0.0.1-SNAPSHOT</version>  
 <packaging>jar</packaging>  
  
 <name>spring-learn</name>  
 <description>Spring Boot Web App</description>  
  
 <parent>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-parent</artifactId>  
 <version>3.2.2</version> <!-- Or whatever version you're using -->  
 </parent>  
  
 <dependencies>  
 <!-- Spring Boot Web -->  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-web</artifactId>  
 </dependency>  
  
 <!-- Logging with SLF4J -->  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-logging</artifactId>  
 </dependency>  
  
 <!-- DevTools (optional for hot reload) -->  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-devtools</artifactId>  
 <scope>runtime</scope>  
 </dependency>  
  
 <!-- Testing -->  
 <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>

## STEP 2: Set Server Port to 8083

In src/main/resources/application.properties, add:

server.port=8083

## STEP 3: Create the Controller

**Create Folder:** com.cognizant.springlearn.controller under src/main/java

**Create Class:** HelloController.java

package com.cognizant.spring\_learn.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);  
  
 @GetMapping("/hello")  
 public String sayHello() {  
 *LOGGER*.debug("START");  
 *LOGGER*.debug("END");  
 return "Hello World!!";  
 }  
}

## STEP 4: Run the Application

In SpringLearnApplication.java, no change is needed as @SpringBootApplication already includes @ComponentScan which will pick up your controller.

* Run it using Shift + F10 or the green icon.

## STEP 5: Test the Endpoint

### In Chrome:

* Open: http://localhost:8083/hello
* Output: Hello World!!

**BONUS: View HTTP Headers**

### In Chrome Developer Tools:

1. Right-click → Inspect → Network tab
2. Reload the /hello page
3. Click the /hello request
4. Under **Headers**, you'll see:
   * Request URL
   * Request Method: GET
   * Status Code: 200 OK
   * Content-Type: text/plain;charset=UTF-8

## SME EXPLANATION NOTES

### 1. @RestController

* Combines @Controller + @ResponseBody
* Every method returns response body (no JSP/view)

### 2. @GetMapping("/hello")

* Maps HTTP GET requests to /hello endpoint

### 3. LOGGER.debug("START") and LOGGER.debug("END")

* Helps trace when the method was entered and exited in logs

### 4. application.properties

* Used to customize server port, etc.

**REST - Country Web Service**

## OBJECTIVE

* Endpoint: GET /country
* Return: Country bean loaded from country.xml
* Output Format: JSON

### 1. Ensure Country.java Exists

**File:** src/main/java/com/cognizant/springlearn/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 + "]";  
 }  
}

### 2. Create country.xml (Spring config)

**File:** 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.spring\_learn.Country">  
 <property name="code" value="IN"/>  
 <property name="name" value="India"/>  
 </bean>  
  
</beans>

### 3. Create CountryController.java

**File:** src/main/java/com/cognizant/springlearn/controller/CountryController.java

package com.cognizant.spring\_learn.controller;  
  
import com.cognizant.spring\_learn.Country;  
import org.slf4j.Logger;  
import org.slf4j.LoggerFactory;  
import org.springframework.context.ApplicationContext;  
import org.springframework.context.support.ClassPathXmlApplicationContext;  
import org.springframework.web.bind.annotation.RequestMapping;  
import org.springframework.web.bind.annotation.RestController;  
  
@RestController  
public class CountryController {  
  
 private static final Logger *LOGGER* = LoggerFactory.*getLogger*(CountryController.class);  
  
 @RequestMapping("/country")  
 public Country getCountryIndia() {  
 *LOGGER*.debug("START");  
 ApplicationContext context = new ClassPathXmlApplicationContext("country.xml");  
 Country country = context.getBean("country", Country.class);  
 *LOGGER*.debug("Country: {}", country);  
 *LOGGER*.debug("END");  
 return country;  
 }  
}

### 4. Set Port

**File:** src/main/resources/application.properties

server.port=8083

### 5. Main Class

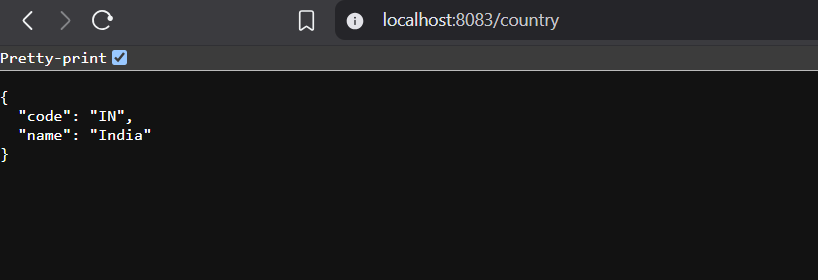
No change needed if you already have:

package com.cognizant.spring\_learn;  
  
import org.slf4j.Logger;  
import org.slf4j.LoggerFactory;  
import org.springframework.boot.SpringApplication;  
import org.springframework.boot.autoconfigure.SpringBootApplication;  
  
@SpringBootApplication  
  
public class SpringLearnApplication {  
 public static void main(String[] args) {  
 SpringApplication.*run*(SpringLearnApplication.class, args);  
 }  
}

### 6. RUN THE APP

* Run SpringLearnApplication.java
* Open browser
* Hit:

http://localhost:8083/country

****

## SME EXPLANATION

### What Happens in the Controller?

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

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

* The country.xml config file is loaded
* Spring instantiates the Country bean
* Properties (code, name) are set via setters
* Bean is returned from the controller

### How Is the Bean Converted into JSON?

* @RestController → implies @ResponseBody on all methods
* Spring Boot uses **Jackson** (via spring-boot-starter-web)
* Jackson auto-converts the Country object into JSON using getters

### Chrome Developer Tools → Network Tab

1. Open Chrome → Go to http://localhost:8083/country
2. Press F12 → Open **Developer Tools**
3. Go to **Network** tab
4. Refresh the page
5. Click the /country request → View **Headers** tab

You’ll see:

#### 📥 Request Headers:

GET /country HTTP/1.1

Host: localhost:8083

User-Agent: Mozilla/5.0 ...

Accept: application/json

#### 📤 Response Headers:

Content-Type: application/json

Content-Length: ...

Date: ...

### Postman → Headers Tab

1. Open Postman
2. GET http://localhost:8083/country
3. Click **Headers** in the Response section

You'll see similar headers like:

content-type: application/json

transfer-encoding: chunked

date: Sun, 13 Jul 2025 ...

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

## 1. country.xml (Spring Configuration File)

Update your src/main/resources/country.xml to hold a list of countries:

<?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="countryList" class="java.util.ArrayList">  
 <constructor-arg>  
 <list>  
 <bean class="com.cognizant.spring\_learn.Country">  
 <property name="code" value="IN"/>  
 <property name="name" value="India"/>  
 </bean>  
 <bean class="com.cognizant.spring\_learn.Country">  
 <property name="code" value="US"/>  
 <property name="name" value="United States"/>  
 </bean>  
 <bean class="com.cognizant.spring\_learn.Country">  
 <property name="code" value="DE"/>  
 <property name="name" value="Germany"/>  
 </bean>  
 <bean class="com.cognizant.spring\_learn.Country">  
 <property name="code" value="JP"/>  
 <property name="name" value="Japan"/>  
 </bean>  
 </list>  
 </constructor-arg>  
</bean>  
  
</beans>

## 2. Country.java (Model)

Located at: src/main/java/com/cognizant/springlearn/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. CountryService.java (Business Logic)

Located at: src/main/java/com/cognizant/springlearn/service/CountryService.java

package com.cognizant.spring\_learn.service;  
  
import com.cognizant.spring\_learn.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;  
  
import java.util.List;  
  
@Service  
public class CountryService {  
  
 private static final Logger *LOGGER* = LoggerFactory.*getLogger*(CountryService.class);  
  
 public Country getCountry(String code) {  
 *LOGGER*.debug("START");  
  
 ApplicationContext context = new ClassPathXmlApplicationContext("country.xml");  
 List<Country> countryList = (List<Country>) context.getBean("countryList");  
  
 return countryList.stream()  
 .filter(country -> country.getCode().equalsIgnoreCase(code))  
 .findFirst()  
 .orElse(null); // Optionally, throw exception  
 }  
}

## 4. CountryController.java

Located at: src/main/java/com/cognizant/springlearn/controller/CountryController.java

package com.cognizant.spring\_learn.controller;  
  
import com.cognizant.spring\_learn.Country;  
import com.cognizant.spring\_learn.service.CountryService;  
import org.slf4j.Logger;  
import org.slf4j.LoggerFactory;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.web.bind.annotation.\*;  
  
@RestController  
public class CountryController {  
  
 private static final Logger *LOGGER* = LoggerFactory.*getLogger*(CountryController.class);  
  
 @Autowired  
 private CountryService countryService;  
  
 @GetMapping("/country/{code}")  
 public Country getCountry(@PathVariable String code) {  
 *LOGGER*.debug("START: code = {}", code);  
 Country country = countryService.getCountry(code);  
 *LOGGER*.debug("END: country = {}", country);  
 return country;  
 }  
}

## 5. SpringLearnApplication.java (Main class)

Located at: src/main/java/com/cognizant/springlearn/SpringLearnApplication.java

package com.cognizant.spring\_learn;  
  
import org.springframework.boot.SpringApplication;  
import org.springframework.boot.autoconfigure.SpringBootApplication;  
  
@SpringBootApplication  
public class SpringLearnApplication {  
 public static void main(String[] args) {  
 SpringApplication.*run*(SpringLearnApplication.class, args);  
 }  
}

## 6. application.properties (Set Port)

Located at: src/main/resources/application.properties

server.port=8083

## 7. Test It

### In Browser:

GET <http://localhost:8083/country/in>

**Output:**

## 

## EXPLANATION

### What Happens in the Controller?

* @PathVariable fetches {code} from the URL
* Calls CountryService.getCountry(code)
* Returns matching Country object

### How It Converts to JSON?

* Spring Boot auto-configures **Jackson**
* The Country object is serialized into JSON because of @RestController

### View HTTP Headers

#### In Chrome DevTools:

1. Open DevTools (F12)
2. Go to **Network** tab → Select /country/in
3. View Response and Headers

Bottom of Form