

2.spring-rest-handson

Hello World RESTful Web Service

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>3.5.3</version>

    <relativePath/> <!-- lookup parent from repository -->

  </parent>

  <groupId>com.cognizant</groupId>

  <artifactId>spring-learn</artifactId>

  <version>0.0.1-SNAPSHOT</version>

  <name>spring-learn</name>

  <description>Spring Web Project</description>

  <url/>

  <licenses>

    <license/>

  </licenses>

  <developers>

    <developer/>

  </developers>

  <scm>
```

```
<connection/>

<developerConnection/>

<tag/>

<url/>

</scm>

<properties>

    <java.version>17</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>
```

```
        </plugins>
    </build>
</project>
```

application.properties

```
spring.application.name=spring-learn
server.port=8083
```

HelloController.java

```
package com.cognizant.spring_learn;

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

        String message = "Hello World!!";

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

        return message;

    }

}
```

SpringLearnApplication.java

```
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 {

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

    public static void main(String[] args) {

        SpringApplication.run(SpringLearnApplication.class, args);

        LOGGER.info("START");

    }

}

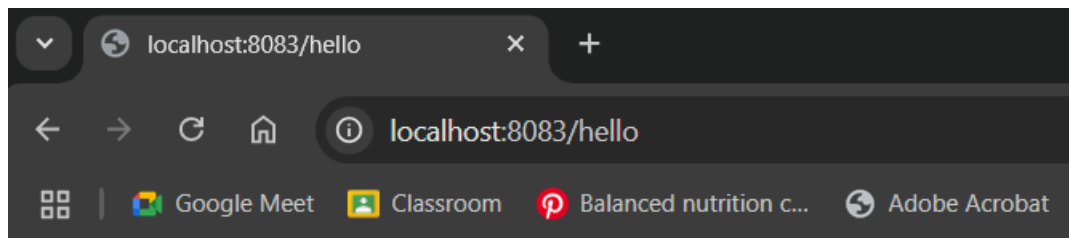
```

Output :

```

2025-07-13T20:49:18.513+05:30 INFO 13268 --- [spring-learn] [ restartedMain] c.c.spring_learn.SpringLearnApplication : Starting SpringLearnApplication using Java 21.0.
2025-07-13T20:49:18.513+05:30 INFO 13268 --- [spring-learn] [ restartedMain] c.c.spring_learn.SpringLearnApplication : No active profile set, falling back to 1 default
2025-07-13T20:49:18.719+05:30 INFO 13268 --- [spring-learn] [ restartedMain] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat initialized with port 8083 (http)
2025-07-13T20:49:18.719+05:30 INFO 13268 --- [spring-learn] [ restartedMain] o.apache.catalina.core.StandardService : Starting service [Tomcat]
2025-07-13T20:49:18.719+05:30 INFO 13268 --- [spring-learn] [ restartedMain] o.apache.catalina.core.StandardEngine : Starting Servlet engine: [Apache Tomcat/10.1.42]
2025-07-13T20:49:18.743+05:30 INFO 13268 --- [spring-learn] [ restartedMain] o.a.c.c.C.[Tomcat-18].[localhost].[/] : Initializing Spring embedded WebApplicationConte
2025-07-13T20:49:18.743+05:30 INFO 13268 --- [spring-learn] [ restartedMain] w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: initialization compl
2025-07-13T20:49:18.818+05:30 INFO 13268 --- [spring-learn] [ restartedMain] o.s.b.d.a.OptionalLiveReloadServer : LiveReload server is running on port 35729
2025-07-13T20:49:18.830+05:30 INFO 13268 --- [spring-learn] [ restartedMain] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port 8083 (http) with context
2025-07-13T20:49:18.834+05:30 INFO 13268 --- [spring-learn] [ restartedMain] c.c.spring_learn.SpringLearnApplication : Started SpringLearnApplication in 0.345 seconds
2025-07-13T20:49:18.834+05:30 INFO 13268 --- [spring-learn] [ restartedMain] .ConditionEvaluationDeltaLoggingListener : Condition evaluation unchanged
2025-07-13T20:49:18.834+05:30 INFO 13268 --- [spring-learn] [ restartedMain] c.c.spring_learn.SpringLearnApplication : START
2025-07-13T20:49:19.123+05:30 INFO 13268 --- [spring-learn] [ restartedMain] .e.DevToolsPropertyDefaultsPostProcessor : Devtools property defaults active! Set 'spring.d
2025-07-13T20:49:19.123+05:30 INFO 13268 --- [spring-learn] [ restartedMain] .e.DevToolsPropertyDefaultsPostProcessor : For additional web related logging consider sett

```



Hello World!!

Headers Explanation (Browser vs Postman)

In Chrome Browser:

1. Open Developer Tools (F12 or Right Click → Inspect → Network tab).
 2. Reload the page.
 3. Click on the /hello request.
 4. Go to the Headers tab:
 - **Request Headers** (sent by browser)
 - Host: localhost:8083
 - User-Agent: Chrome version
 - Accept: text/html,...
 - **Response Headers** (received from server)
 - Content-Type: text/plain;charset=UTF-8
 - Content-Length: 13
 - Date: [server timestamp]
-

In Postman:

1. Open Postman and send a GET request to `http://localhost:8083/hello`.
2. Click on the **Headers** tab in the response section:
 - You will see response headers like:
 - Content-Type: text/plain;charset=UTF-8
 - Content-Length: 13
 - Date: [server timestamp]

REST - Country Web Service

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>3.5.3</version>

        <relativePath/>

    </parent>

    <groupId>com.cognizant</groupId>

    <artifactId>spring-learn</artifactId>

    <version>0.0.1-SNAPSHOT</version>

    <name>spring-learn</name>

    <description>Spring Web Project</description>

    <url/>

    <licenses>

        <license/>

    </licenses>

    <developers>

        <developer/>

    </developers>

    <scm>

        <connection/>

        <developerConnection/>

        <tag/>
```

```
        <url/>
    </scm>
    <properties>
        <java.version>17</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>
```

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="in" class="com.cognizant.spring_learn.Country">

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

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

    </bean>

</beans>
```

application.properties

```
spring.application.name=spring-learn
server.port=8083
```

Country.java

```
package com.cognizant.spring_learn;

public class Country {

    private String code;

    private String name;

    public Country() {}

    public Country(String code, String name) {

        this.code = code;

        this.name = name;

    }

    public String getCode() {

        return code;

    }

    public void setCode(String code) {

        this.code = code;

    }

}
```



```

    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }
}

```

CountryController.java

```

package com.cognizant.spring_learn;

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.info("START: getCountryIndia()");

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

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

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

        return country;
    }
}

```

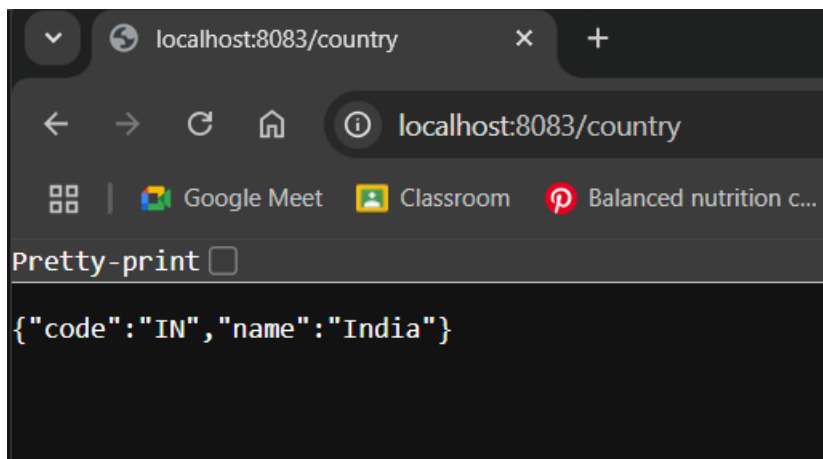
Output :

```

  ____  __  __
 / ___/  / /  /  ___  /
/ /   /  /  /  / _ /
/ /___/  /  /  / ___/
/_____/___/_/___/_/

Spring Boot (v3.5.3)

2025-07-13T21:16:49.937+05:30 INFO 18332 --- [spring-learn] [ restartedMain] c.c.spring_learn.SpringLearnApplication : Starting SpringLearnApplication using Java 21.0.3 with PID 18332 (C:\Users\sashm\
2025-07-13T21:16:49.937+05:30 INFO 18332 --- [spring-learn] [ restartedMain] c.c.spring_learn.SpringLearnApplication : No active profile set, falling back to 1 default profile: "default"
2025-07-13T21:16:50.335+05:30 INFO 18332 --- [spring-learn] [ restartedMain] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat initialized with port 8083 (http)
2025-07-13T21:16:50.342+05:30 INFO 18332 --- [spring-learn] [ restartedMain] o.apache.catalina.core.StandardService : Starting service [Tomcat]
2025-07-13T21:16:50.342+05:30 INFO 18332 --- [spring-learn] [ restartedMain] o.apache.catalina.core.StandardEngine : Starting servlet engine: [Apache Tomcat/10.1.42]
2025-07-13T21:16:50.363+05:30 INFO 18332 --- [spring-learn] [ restartedMain] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Spring embedded WebApplicationContext
2025-07-13T21:16:50.370+05:30 INFO 18332 --- [spring-learn] [ restartedMain] w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: initialization completed in 427 ms
2025-07-13T21:16:50.525+05:30 INFO 18332 --- [spring-learn] [ restartedMain] o.s.b.d.a.OptionalLiveReloadServer : LiveReload server is running on port 35729
2025-07-13T21:16:50.549+05:30 INFO 18332 --- [spring-learn] [ restartedMain] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port 8083 (http) with context path '/'
2025-07-13T21:16:50.555+05:30 INFO 18332 --- [spring-learn] [ restartedMain] c.c.spring_learn.SpringLearnApplication : Started SpringLearnApplication in 0.712 seconds (process running for 1241.262)
2025-07-13T21:16:50.558+05:30 INFO 18332 --- [spring-learn] [ restartedMain] .ConditionEvaluationDeltaLoggingListener : Condition evaluation unchanged
2025-07-13T21:16:50.558+05:30 INFO 18332 --- [spring-learn] [ restartedMain] c.c.spring_learn.SpringLearnApplication : START
```



SME Explanation

1. What Happens in the Controller Method?

- When a request is made to /country, Spring calls `getCountryIndia()`.
- It logs the start of the method.
- It loads the Spring context from `country.xml`.
- It retrieves the bean with ID "in" (which is the India country bean).
- Returns the bean as a response.
- Logs the end of the method.

2. How is the Bean Converted to JSON?

- Spring Boot uses **Jackson** (automatically included in `spring-boot-starter-web`) to convert Java objects into JSON.
- When you return a `Country` object from a `@RestController`, Spring uses Jackson to serialize it.
- Jackson inspects getters (`getCode()`, `getName()`) and creates the JSON:

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

3. View HTTP Headers in Browser Developer Tools

- Open Chrome → Dev Tools (F12) → **Network** tab.
 - Hit the URL: `http://localhost:8083/country`
 - Click on the `/country` request.
 - Go to **Headers** tab:
 - **Request Headers:**
 - Host: `localhost:8083`
 - User-Agent: `Chrome`
 - Accept: `application/json`
 - **Response Headers:**
 - Content-Type: `application/json`
 - Content-Length: `XX`
 - Date: `[Timestamp]`
-

4. View HTTP Headers in Postman

- Open Postman → Send GET `http://localhost:8083/country`
- Click **Headers** tab in the **response** section:
 - You will see:
 - Content-Type: `application/json`
 - Content-Length: `XX`
 - Date: `[Timestamp]`
 - Connection: `keep-alive`

REST - Get country based on country code

country.xml

```
<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="CN" />
          <property name="name" value="China" />
        </bean>
      </list>
    </constructor-arg>
  </bean>
</beans>
```

Country.java

```
package com.cognizant.spring_learn;

public class Country {
  private String code;
```

```

private String name;

public Country() {}

public Country(String code, String name) {
    this.code = code;
    this.name = name;
}

public String getCode() {
    return code;
}

public void setCode(String code) {
    this.code = code;
}

public String getName() {
    return name;
}

public void setName(String name) {
    this.name = name;
}
}

```

CountryController.java

```

package com.cognizant.spring_learn;

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("/countries/{code}")
public Country getCountry(@PathVariable String code) {
    LOGGER.info("START: getCountry()");
    Country country = countryService.getCountry(code);
    LOGGER.info("END: getCountry()");
    return country;
}
}

```

CountryService.java

```

package com.cognizant.spring_learn;

import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;
import org.springframework.stereotype.Service;
import java.util.List;

@Service
public 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);
    }
}

```

SpringLearnApplication.java

```

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 {  
    private static final Logger LOGGER = LoggerFactory.getLogger(SpringLearnApplication.class);  
    public static void main(String[] args) {  
        SpringApplication.run(SpringLearnApplication.class, args);  
        LOGGER.info("START");  
    }  
}
```

Output :

