**Need for Spring Boot**

Traditional Spring applications require:

* A lot of boilerplate code
* Manual configuration (XML/Java config)
* Manual deployment setup (WAR packaging, server setup)
* Managing multiple dependencies for web, data, security, etc.

These steps make development slow, error-prone, and complex, especially for beginners or rapid prototyping.

To overcome these issues Spring Boot was introduced.

Benefits of Spring Boot

**1. Auto-Configuration**

* Automatically configures Spring components based on classpath and properties.

**2. No XML Configuration**

* Most setup is done using annotations and properties (application.properties or application.yml).

**3. Embedded Web Server**

* Comes with built-in servers like **Tomcat**, **Jetty**, or **Undertow**.

**4. Easy Testing**

* Supports unit, integration, and web tests with Spring Boot Test and MockMVC

**Configuring Spring Web Project**

Using spring initializer all the dependencies are added these include Spring WEB and Spring Developer tools essential for spring web application.

**Load Country from Spring Configuration XML**

Steps

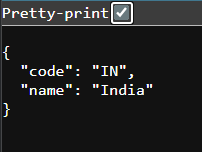
1. Create the Bean Class named Country.java

2.Create the Spring Configuration File

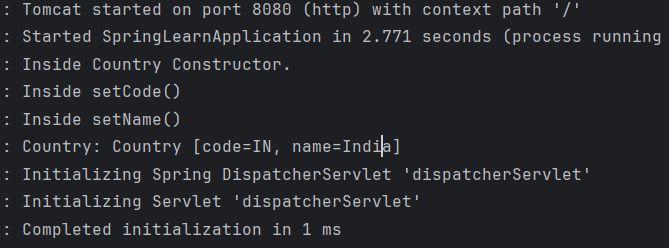
3.Create a REST controller called CountryController  
4.Update the SpringLearnApplication class

5.Run and visit <http://localhost:8080/country> to view the output

**OUTPUT (With REST)**



**OUTPUT(Without REST)**

****

**1.What Happens in the Controller Method?**

* The URL /country triggers the method due to @RequestMapping.
* Inside the method:
  + ClassPathXmlApplicationContext loads the country.xml file.
  + The bean with id="country" is fetched and returned.
* Because of @RestController, the returned Country object is automatically serialized to JSON and sent back as the HTTP response.

**2. How the Bean is Converted into JSON Response?**

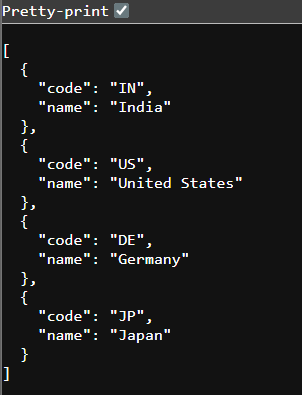
* Spring Boot includes **Jackson** (a JSON processing library) by default.
* The object returned from your controller is passed to **HttpMessageConverter.**
* Spring Boot uses MappingJackson2HttpMessageConverter to:
  + Convert the Java object to JSON
  + Set the content type to json in the response header

**Get all countries**

To get all countries :

1. Update country.xml and add a list of all the countries.
2. Updte the CountryController class and use @GetMapping(“/countries”) to map to a URL

**OUTPUT**



**Get country by code**

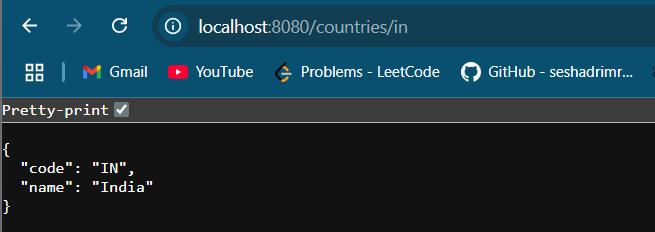
Create a new class called CountryService to

* Loads the XML
* Retrieves the list
* Finds the matching country using a Java stream and case-insensitive comparison

What happens in the controller?

* @RestController tells Spring to return objects as JSON by default.
* @GetMapping("/countries/{code}") maps GET requests to /countries/in or /countries/us.
* @PathVariable extracts the {code} from the URL.
* It calls CountryService.getCountry(code) to fetch the result.

**OUTPUT(code = in)**

****