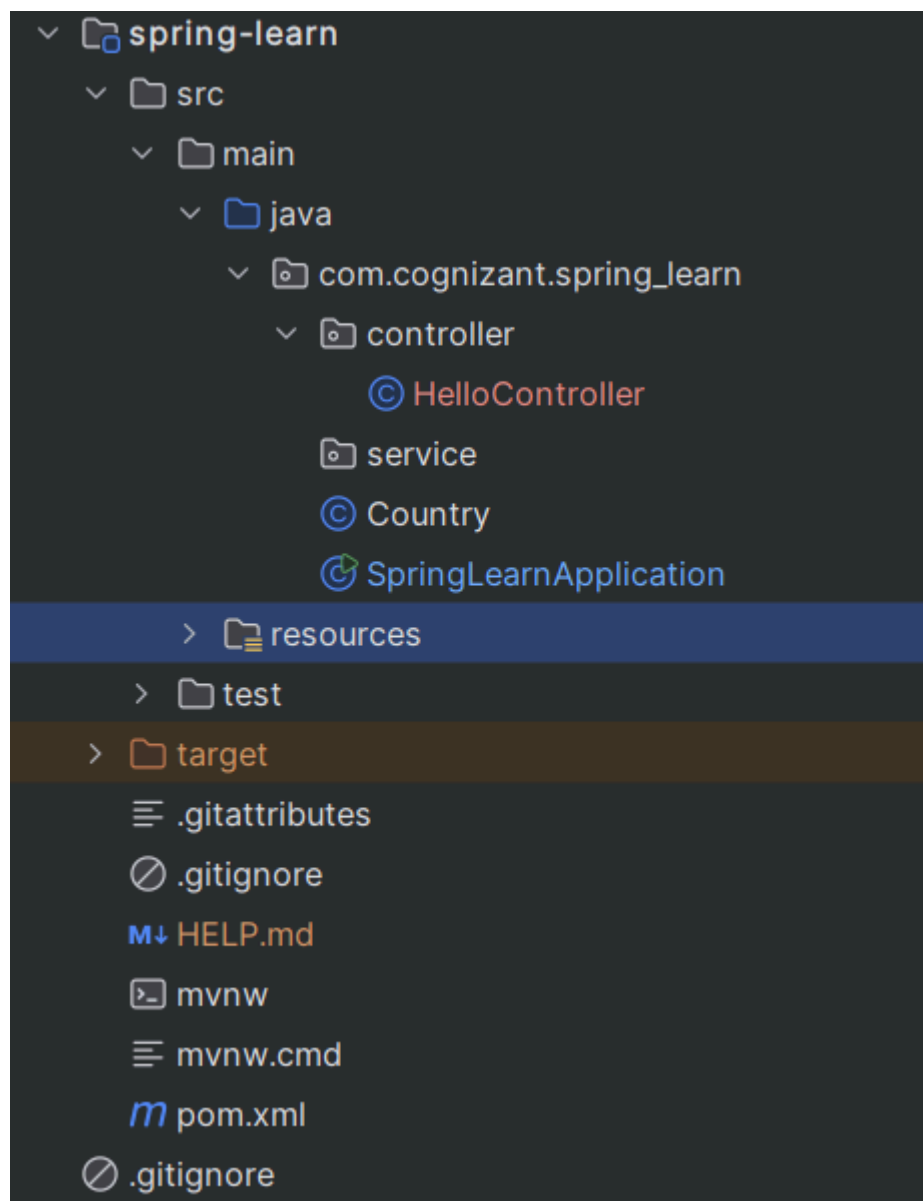


Spring REST Hands on

Source Code: [Here](#)

Hello World RESTful Web Service

Source Structure



Code

```
package com.cognizant.spring_learn.controller;  
  
import org.springframework.web.bind.annotation.GetMapping;
```

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

@RestController
@RequestMapping("/hello")
public class HelloController {
    @GetMapping
    public String sayHello(){
        return "Hello World!!";
    }
}
```

Output

Console logging messages using Logger

```
250710|12:36:47.505|http-nio-8083-exec-2|DEBUG|a.j.AuthConfigFactoryImpl| log>Loading persistent provider registrations from [C:\Users\hp\AppData\Local\Te
250710|12:36:47.505|http-nio-8083-exec-2| INFO|o.a.c.c.C.[.[/] | log|Initializing Spring DispatcherServlet 'dispatcherServlet'
250710|12:36:47.505|http-nio-8083-exec-2| INFO|o.s.w.s.DispatcherServlet| initServletBean|Initializing Servlet 'dispatcherServlet'
250710|12:36:47.505|http-nio-8083-exec-2| INFO|o.s.w.s.DispatcherServlet| initServletBean|Completed initialization in 0 ms
250710|12:36:47.536|http-nio-8083-exec-2| INFO|c.c.s.c.HelloController | sayHello|Start sayHello()
250710|12:36:47.536|http-nio-8083-exec-2| INFO|c.c.s.c.HelloController | sayHello|End sayHello()
```

Calling /hello route from Postman

GET Say Hello

Week-4 / Spring Rest / Say Hello

GET http://localhost:8083/hello

Overview Params Authorization Headers (8) Body Scripts Settings Cookies

Key	Value	Description
Cookie	refresh=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJlb...	
Cache-Control	no-cache	
Postman-Token	<calculated when request is sent>	
Host	<calculated when request is sent>	
User-Agent	PostmanRuntime/7.43.2	
Accept	/*/*	
Accept-Encoding	gzip, deflate, br	
Connection	keep-alive	

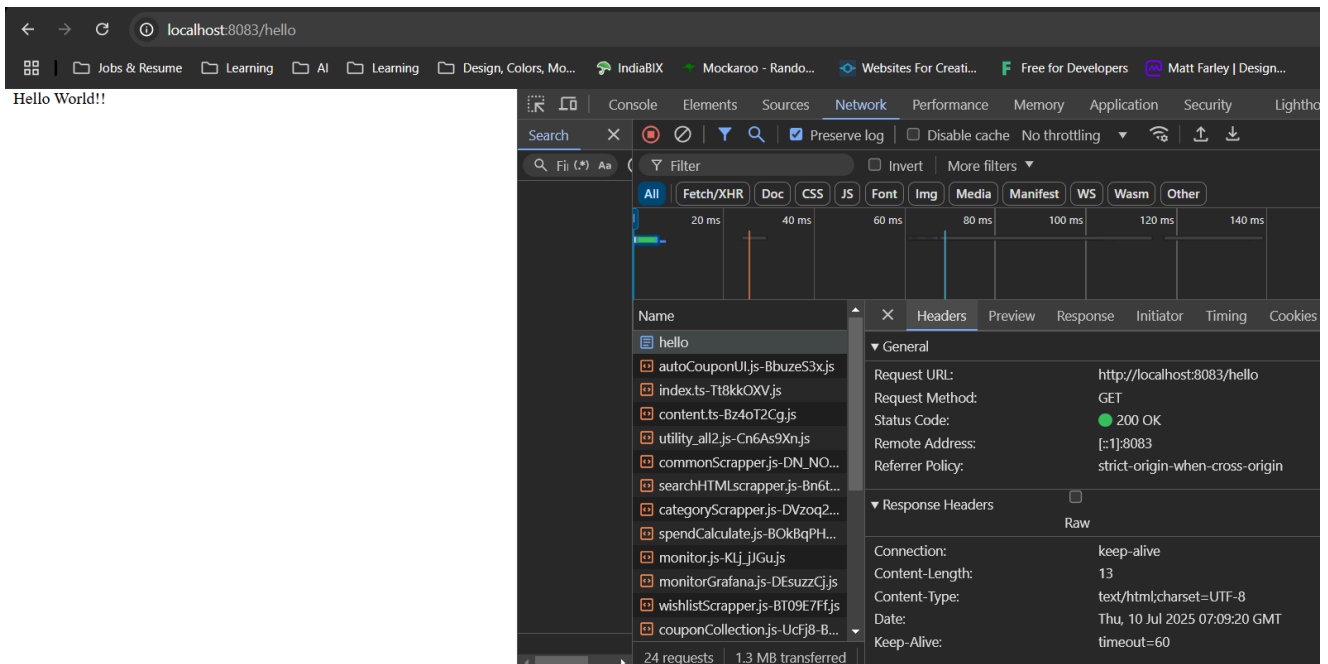
Body Cookies (1) Headers (5) Test Results

200 OK • 119 ms • 177 B • Save Response

Raw Preview Visualize

1 Hello World!!

Calling /hello route from Chrome



REST - Country Web Service

Code

```
package com.cognizant.spring_learn.controller;

import com.cognizant.spring_learn.Country;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

@RestController
@RequestMapping("/country")
public class CountryController {

    @GetMapping
    public Country getCountryIndia(){
        ApplicationContext context = new
ClassPathXmlApplicationContext("country.xml");
        return context.getBean("country", Country.class);
    }
}
```

Output

What happens in the controller method?

Here we have created an endpoint `/country` that returns information about a country (in this case, India). But instead of using the modern annotation-based approach, I have used **XML configuration file** (`country.xml`) to define the `Country` bean (As we have done in previous hands on). This controller reads this bean and returns it as a JSON response (By default).

Here we have used `@RestController` which is combination of `@Controller` and `@ResponseBody`.

`@ResponseBody` tells Spring **not to render HTML** but to **write the return value into the HTTP response body** (i.e., as JSON)

How the bean is converted into JSON response?

Spring Boot uses a library called **Jackson** to automatically convert Java objects (beans) into **JSON format** when returning them from a `@RestController`.

Jackson only recognizes **Non-static** fields, that have public **getters** or are accessible and which are not marked with `@JsonIgnore`.

Calling `/hello` route from Postman

The screenshot shows the Postman interface for a GET request to `http://localhost:8083/country`. The **Headers** tab is selected, showing a list of headers:

Key	Value	Description
Cookie	refresh=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJlb...	
Cache-Control	no-cache	
Postman-Token	<calculated when request is sent>	
Host	<calculated when request is sent>	
User-Agent	PostmanRuntime/7.43.2	
Accept	*/*	
Accept-Encoding	gzip, deflate, br	
Connection	keep-alive	

The **Body** tab is also visible, showing a JSON response:

```
1 {
2   "code": "IN",
3   "name": "India"
4 }
```

The status bar at the bottom indicates a **200 OK** response with a response time of 309 ms and a body size of 192 B.

Calling `/hello` route from Chrome

localhost:8083/country

Jobs & ResumeLearningAI LearningDesign, Colors, Mo...IndiaBIXMockaroo - Rando...Websites For Creati...Free for DevelopersMatt Farley | Design...

Pretty-print

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

ConsoleElementsSourcesNetworkPerformanceMemoryApplicationSecurityLighthouse

Search

Filter

AllFetch/XHRDocCSSJSFontImgMediaManifestWSWasmOther

Name

utility_all2-B1qfpIt.jsindex-C_MXz-c4.jsapp-DeFgW6qY.jsSpendCalculate-C-2kmK1i.jsen-CWaS8mPg.jscountryautoCouponUI.js-BbuzeS3x.jsindex.ts-T18kkOXV.jscontent.ts-Bz4oT2Cg.jsutility_all2.js-Cn6As9Xn.jscommonScrapper.js-DN_NO...searchHTMLScrapper.js-Bn6t...categoryScrapper.js-DVzoq2...

HeadersPreviewResponseInitiatorTimingCookies

General

Request URL:http://localhost:8083/countryRequest Method:GETStatus Code:200 OKRemote Address:[::1]:8083Referrer Policy:strict-origin-when-cross-origin

Response Headers

Raw

Connection:keep-aliveContent-Type:application/jsonDate:Thu, 10 Jul 2025 07:25:26 GMTKeep-Alive:timeout=60Transfer-Encoding:chunked

48 requests2.7 MB transferred