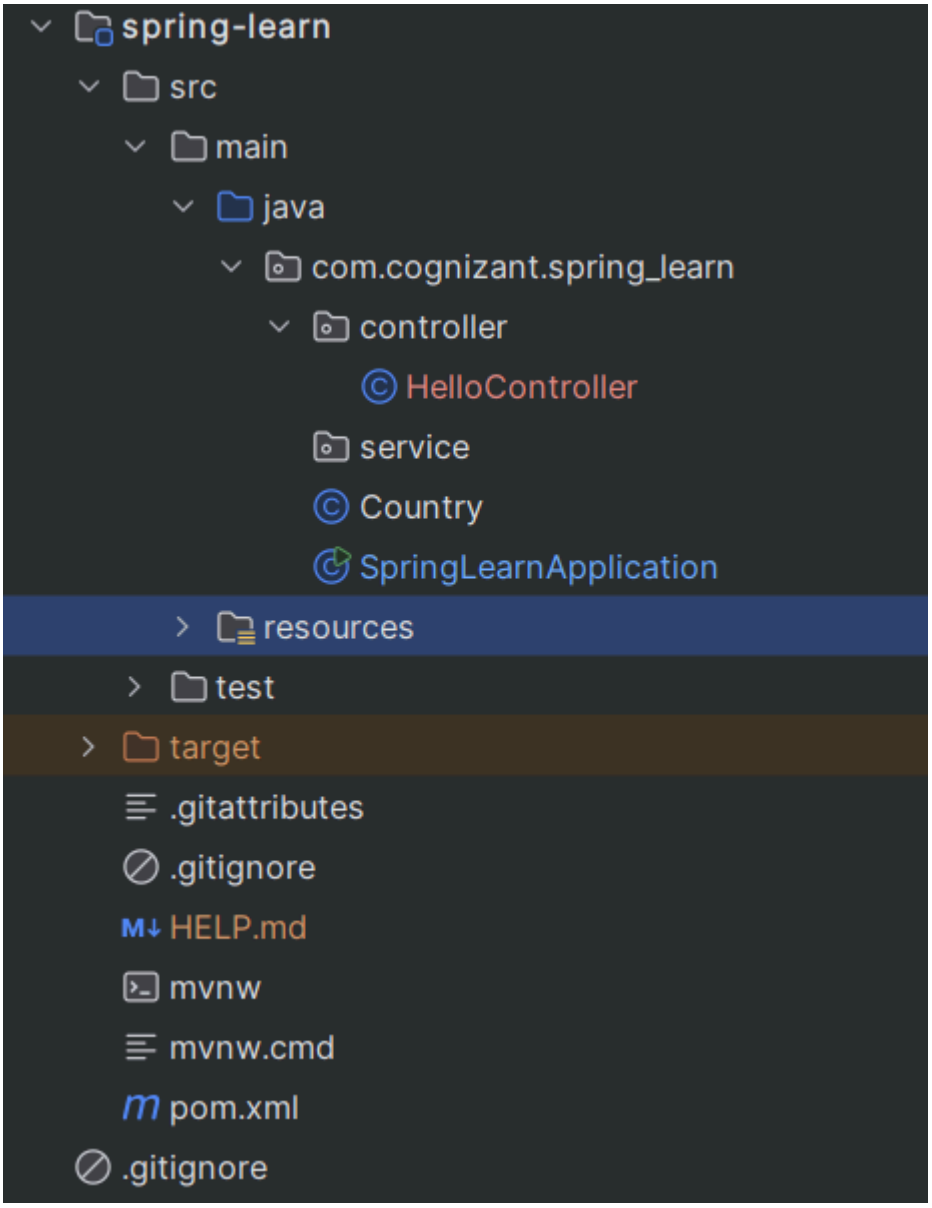
Spring REST Hands on

Source Code: Here

Hello World RESTful Web Service

Source Structure



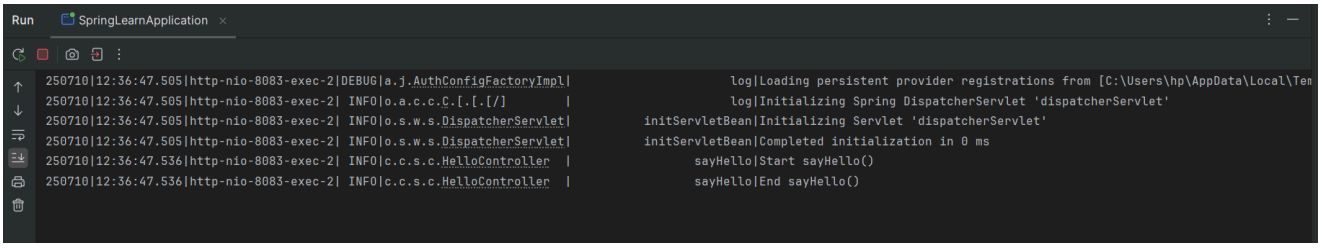
Code

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

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

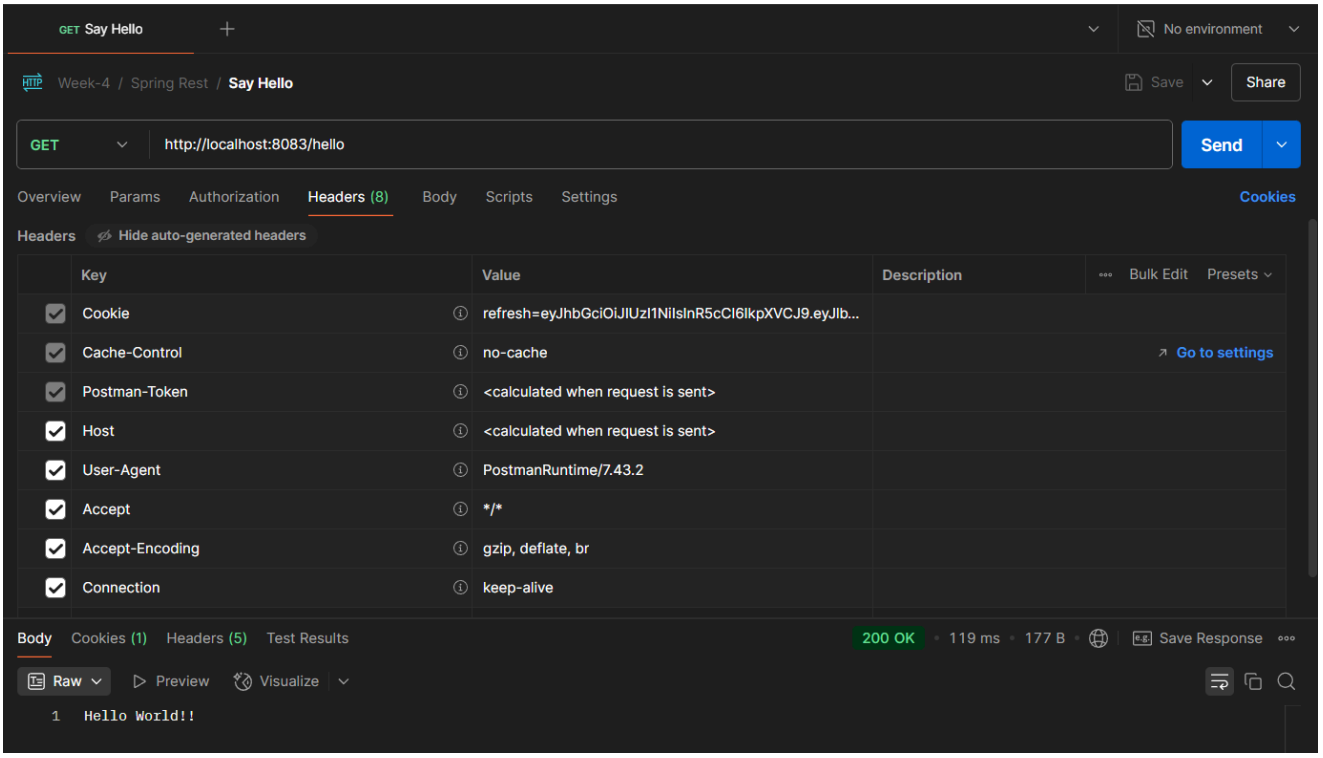
```java
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

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

Output

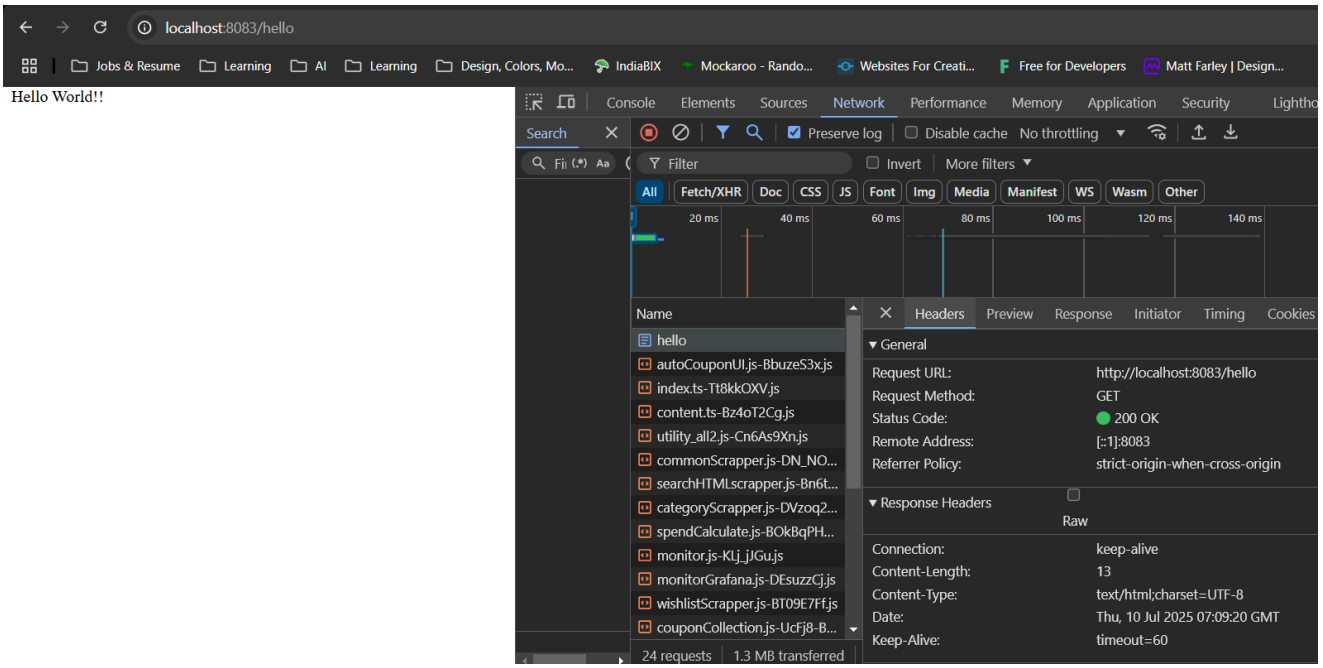
Console logging messages using Logger



Calling `/hello` route from Postman



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

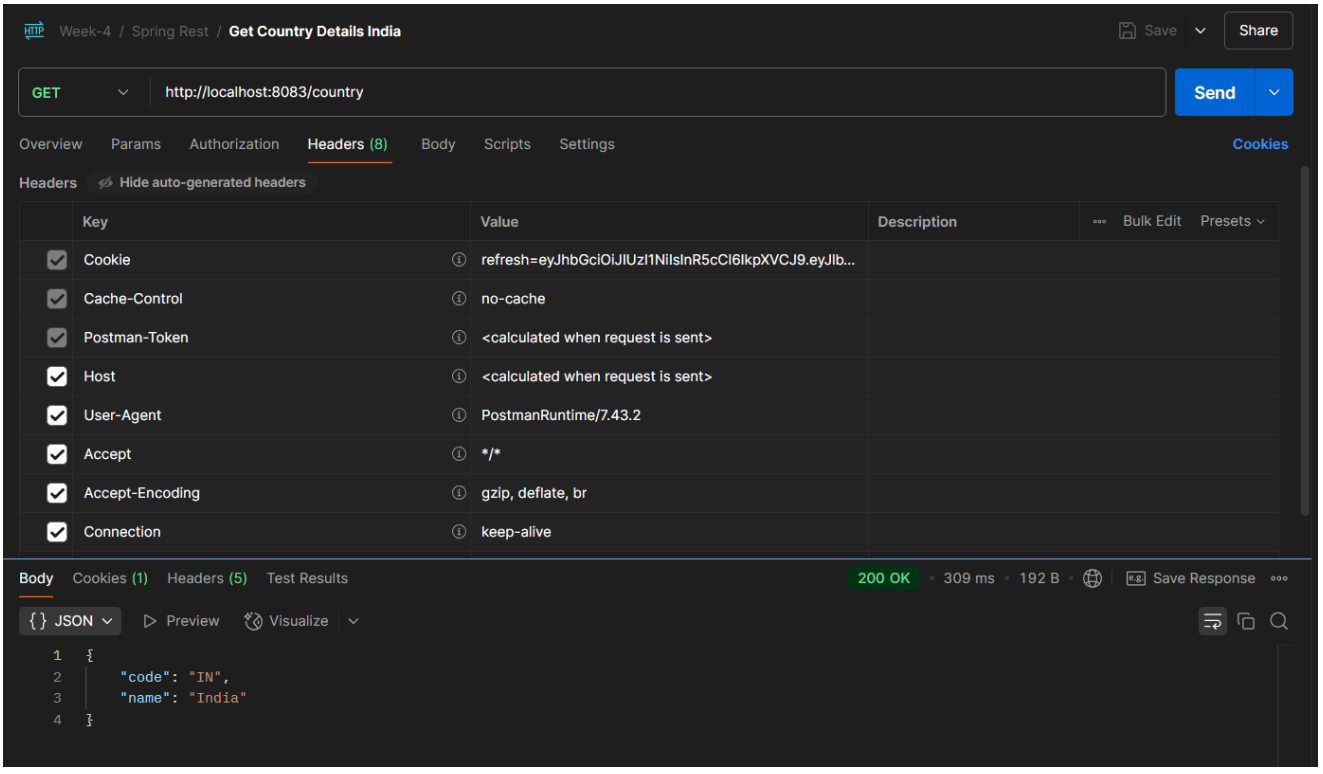
What happens in the controller method?

> 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)

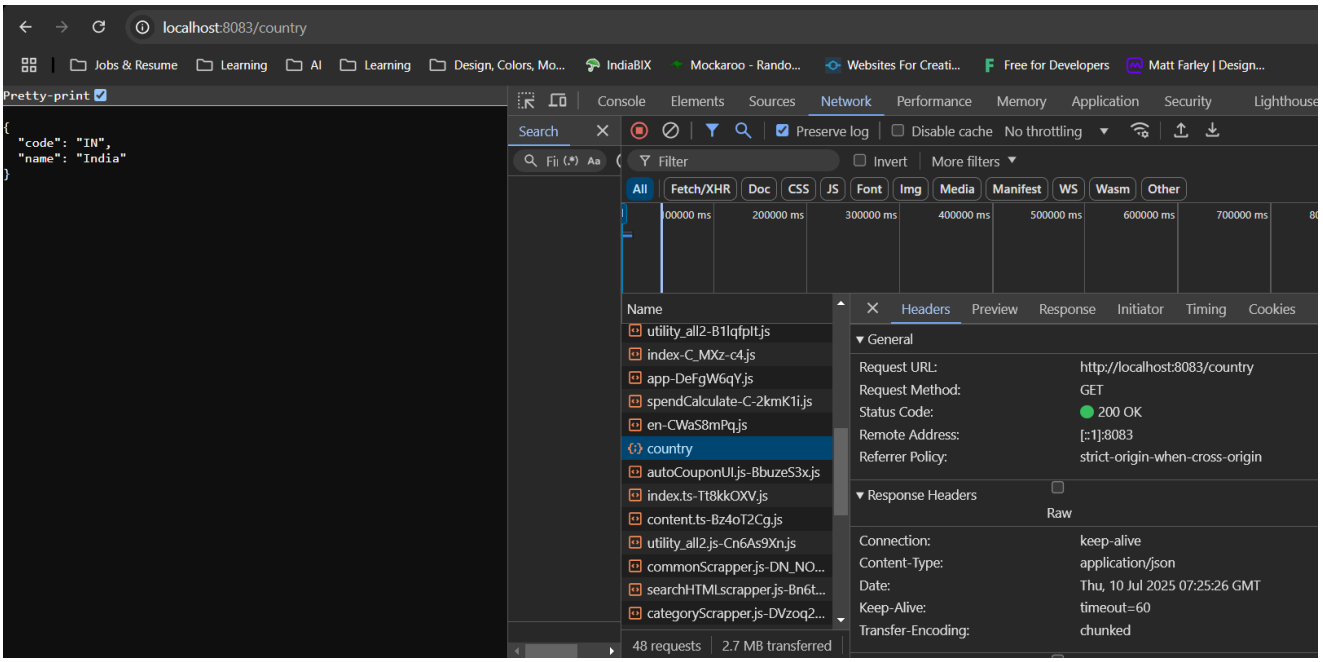
How the bean is converted into JSON reponse?

> 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



Calling `/hello` route from Chrome


localhost:8083/country
Jobs & Resume
Learning
AI
Learning
Design, Colors, Mo...
IndiaBIX
Mockaroo - Rando...
Websites For Creati...
Free for Developers
Matt Farley | Design...
Pretty-print
{
"code": "IN",
"name": "India"
}
Console
Elements
Sources
Network
Performance
Memory
Application
Security
Lighthouse
Search
Preserve log
Disable cache
No throttling
Filter
Invert
More filters
All
Fetch/XHR
Doc
CSS
JS
Font
Img
Media
Manifest
WS
Wasm
Other
00000 ms
200000 ms
300000 ms
400000 ms
500000 ms
600000 ms
700000 ms
Name
utility_all2-B1lqfpIt.js
index-C_MXz-c4.js
app-DeFgW6qY.js
spendCalculate-C-2kmK1i.js
en-CWaS8mPq.js
country
autoCouponUI.js-BbuzeS3x.js
index.ts-Tt8kkOXV.js
content.ts-Bz4oT2Cg.js
utility_all2.js-Cn6As9Xn.js
commonScrapper.js-DN_NO...
searchHTMLscrapper.js-Bn6t...
categoryScrapper.js-DVzoq2...
48 requests
2.7 MB transferred
Headers
Preview
Response
Initiator
Timing
Cookies
General
Request URL:
http://localhost:8083/country
Request Method:
GET
Status Code:
200 OK
Remote Address:
[::1]:8083
Referrer Policy:
strict-origin-when-cross-origin
Response Headers
Raw
Connection:
keep-alive
Content-Type:
application/json
Date:
Thu, 10 Jul 2025 07:25:26 GMT
Keep-Alive:
timeout=60
Transfer-Encoding:
chunked