

# LAB 1

## Part 1: Getting it printed on console:

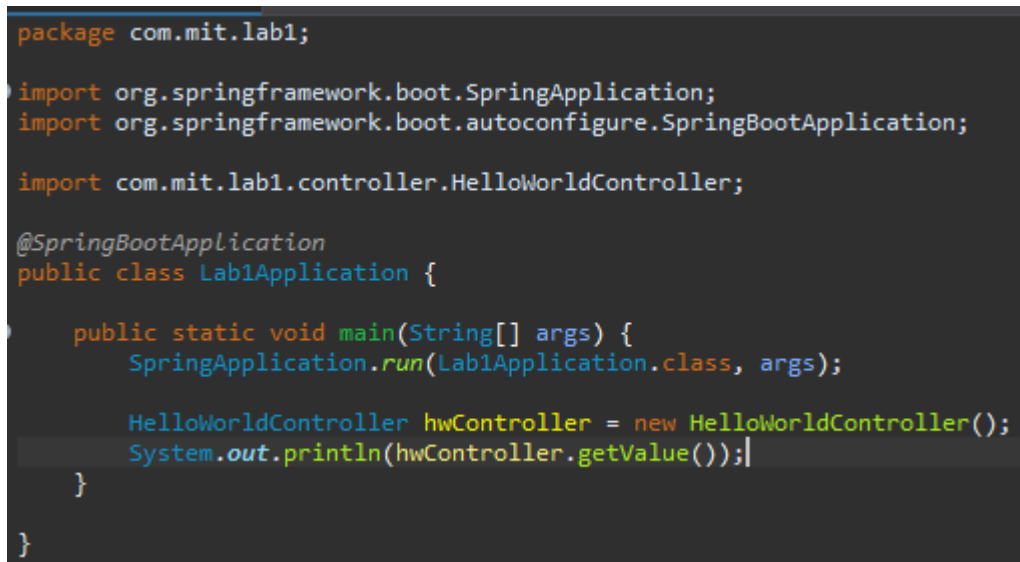
**Step 1:** I created a sub-package called “controller” in my main package. I then created a class called “HelloWorldController” in my newly created sub-package.

I then added a getter for my private String variable that I will use in the main program.

A screenshot of an IDE showing the code for HelloWorldController.java. The code is as follows:

```
1 package com.mit.lab1.controller;
2
3
4 public class HelloWorldController {
5     private String value = "Hello, world!";
6
7     public String getValue() {
8         return value;
9     }
10 }
11
```

**Step 2:** I imported my newly created class to my main program. I then created an object of this new class and used the getter method to access the string. I used this return value to print to the console.

A screenshot of an IDE showing the code for Lab1Application.java. The code is as follows:

```
package com.mit.lab1;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

import com.mit.lab1.controller.HelloWorldController;

@SpringBootApplication
public class Lab1Application {

    public static void main(String[] args) {
        SpringApplication.run(Lab1Application.class, args);

        HelloWorldController hwController = new HelloWorldController();
        System.out.println(hwController.getValue());
    }
}
```

Output:

```
2025-01-20T10:24:47.758+05:30
2025-01-20T10:24:47.763+05:30
2025-01-20T10:24:47.764+05:30
2025-01-20T10:24:47.765+05:30
Hello, world!
```

## Part 2: Setting up an API for this string

**Step 1:** I modified my existing class “HelloWorldController” that resides in the “controller” sub-package. Specifically, I added the “@RestController”, “@RequestMapping”, and “@GetMapping” annotations to inform the spring framework that this is a service class.

```
package com.mit.lab1.controller;

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

@RestController
@RequestMapping("/hello-world")
public class HelloWorldController {
    private String value = "Hello, world!";

    @GetMapping("/get")
    public String getValue() {
        return value;
    }
}
```

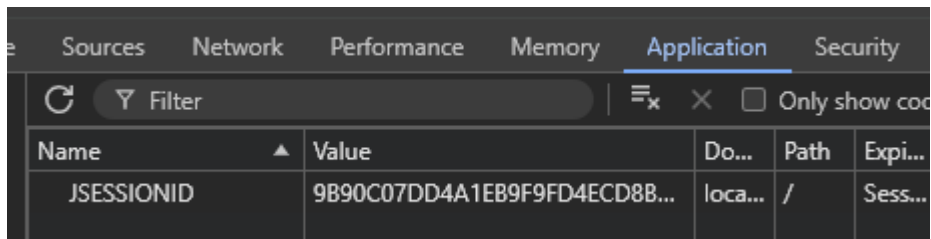
← → ↻ ⓘ localhost:8080/hello-world/get

Hello, world!

```
src/main/java
├── com.mit.lab1
│   ├── Lab1Application.java
│   └── com.mit.lab1.controller
│       └── HelloWorldController.java
```

## Part 3: Sending a Postman GET request

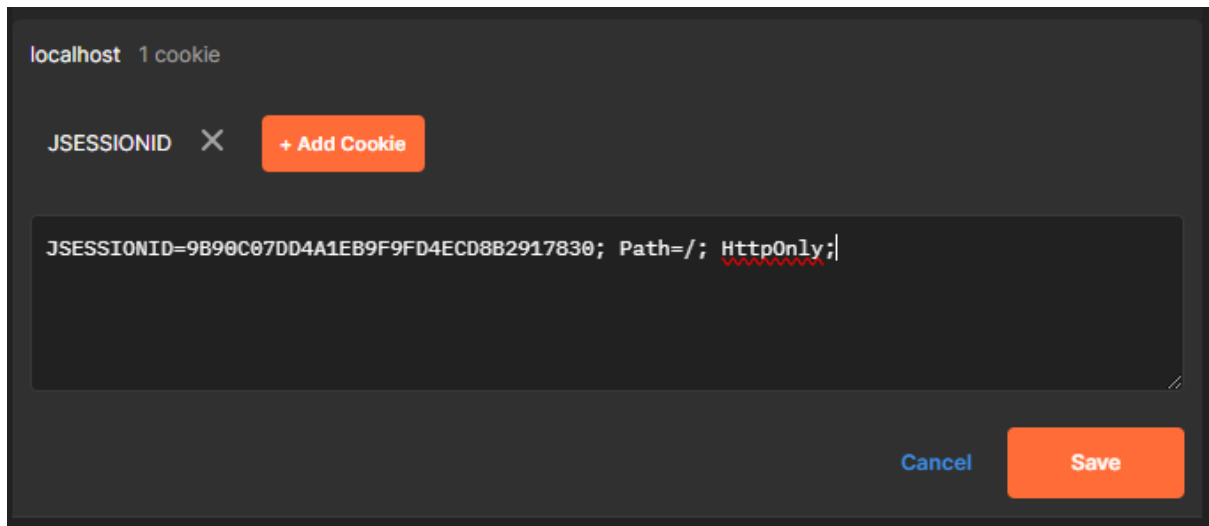
**Step 1:** I retrieved my “JSESSIONID” cookie from my browser developer tools.



The screenshot shows the Chrome DevTools Application tab with the 'Cookies' section expanded. A table lists the cookies for the current page.

Name	Value	Domain	Path	Expires
JSESSIONID	9B90C07DD4A1EB9F9FD4ECD8B...	localhost	/	Session

**Step 2:** I added this value in Postman cookie settings.



**Step 3:** I sent a GET request to <http://localhost:8080/hello-world/get>. Output:

