**Hello World RESTful Web Service:**

**Solution:**

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) {

***LOGGER***.info("STARTING SPRING LEARN APPLICATION...");

SpringApplication.*run*(SpringLearnApplication.**class**, args);

***LOGGER***.info("SPRING LEARN APPLICATION STARTED SUCCESSFULLY.");

}

}

HelloController.java:

**package** com.cognizant.spring\_learn.controller;

**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;

}

}

When we test our REST service in the **Network tab of Chrome Developer Tools**, we can view HTTP header details by refreshing the /hello URL while the Network tab is open and clicking on the request entry. The **Request Headers** show what the browser sends to our Spring application, including the GET method, Host: localhost:8080, User-Agent, and Accept headers indicating acceptable response formats. The **Response Headers** show what our Spring application returns, including:

1. Content-Type: text/plain;charset=UTF-8, indicating the response type.
2. Content-Length: 13, showing the size of “Hello World!!”.
3. Date, showing when the response was generated.
4. Connection: keep-alive, indicating the connection remains open for further requests.

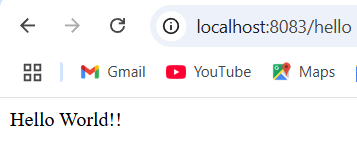
Similarly, when we test in **Postman** by sending a GET request to http://localhost:8080/hello, we can click on the **Headers tab in the response section** to view the received HTTP header details. We will see:

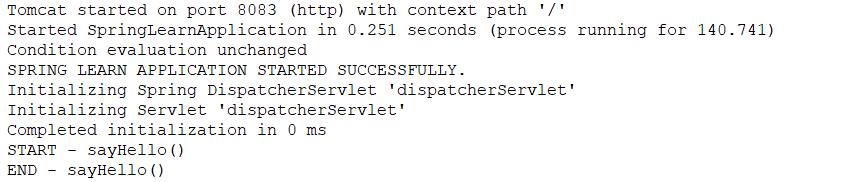
1. content-type: text/plain;charset=UTF-8
2. content-length: 13
3. date: [current date and time]
4. connection: keep-alive

These headers confirm that our REST API is working correctly, returning the expected plain text and content length, and managing the HTTP connection efficiently. By observing these headers, we understand how the browser and Postman communicate with our Spring REST application, demonstrating clear knowledge of RESTful communication during our lab discussions.

Top of Form

Bottom of Form

OUTPUT:



**REST - Country Web Service:**

**Solution:**

**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) {

***LOGGER***.info("STARTING Spring Learn Application...");

SpringApplication.*run*(SpringLearnApplication.**class**, args);

***LOGGER***.info("Spring Learn Application STARTED and ready on port 8083.");

}

}

**Country.java:**

**package** com.cognizant.spring\_learn.model;

**public** **class** Country {

**private** String code;

**private** String name;

// Getters and setters

**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.controller;

**import** com.cognizant.spring\_learn.model.Country;

**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");

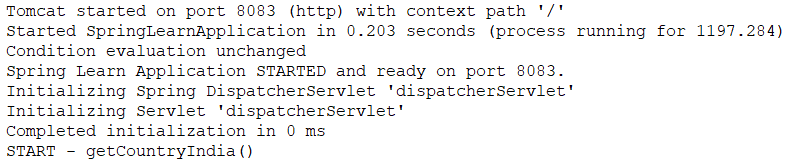
***LOGGER***.info("END getCountryIndia");

**return** country;

}

}

OUTPUT:



**REST - Get country based on country code:**

**Solution:**

**CountryController.java:**

**package** com.cognizant.spring\_learn.controller;

**import** com.cognizant.spring\_learn.model.Country;

**import** com.cognizant.spring\_learn.service.CountryService;

**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) **throws** Exception {

***LOGGER***.info("START getCountry with code: {}", code);

Country country = countryService.getCountry(code);

***LOGGER***.info("END getCountry with result: {}", country);

**return** country;

}

}

**CountryService.java:**

**package** com.cognizant.spring\_learn.service;

**import** com.cognizant.spring\_learn.model.Country;

**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) **throws** Exception {

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

List<Country> countryList = (List<Country>) context.getBean("countryList");

**return** countryList.stream()

.filter(country -> country.getCode().equalsIgnoreCase(code))

.findFirst()

.orElseThrow(() -> **new** Exception("Country not found with code: " + code));

}

}

**SpringLearnApplication.java:**

**package** com.cognizant.spring\_learn;

**import** org.springframework.boot.SpringApplication;

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

@SpringBootApplication

**public** **class** SpringLearnApplication {

**public** **static** **void** main(String[] args) {

SpringApplication.*run*(SpringLearnApplication.**class**, args);

}

}

OUTPUT:

