1.Create the directory structure

mkdir -p src/main/java/hello

2.Create pom.xml to import the Spring dependency.

<?xml version="1.0" encoding="UTF-8"?>

<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>org.springframework</groupId>

<artifactId>sample-rest-service</artifactId>

<version>1.0</version>

<parent>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-parent</artifactId>

<version>1.5.9.RELEASE</version>

</parent>

<dependencies>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-web</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-test</artifactId>

<scope>test</scope>

</dependency>

<dependency>

<groupId>com.jayway.jsonpath</groupId>

<artifactId>json-path</artifactId>

<scope>test</scope>

</dependency>

</dependencies>

<properties>

<java.version>1.8</java.version>

</properties>

<build>

<plugins>

<plugin>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-maven-plugin</artifactId>

</plugin>

</plugins>

</build>

<repositories>

<repository>

<id>spring-releases</id>

<url>https://repo.spring.io/libs-release</url>

</repository>

</repositories>

<pluginRepositories>

<pluginRepository>

<id>spring-releases</id>

<url>https://repo.spring.io/libs-release</url>

</pluginRepository>

</pluginRepositories>

</project>

3.Create the resource representation class

It will handle GET requests for /greeting, optionally with a name parameter in the query string. The GET request should return a 200 OK response with JSON in the body

{"content": " "}

4.Provide a plain old java object with fields, constructors, and accessors for the content data:

src/main/java/hello/Resource.java

package hello;

public class Resource

{

private final String content;

public Resource(String content)

{

this.content = content;

}

public String getContent() {

return content;

}

}

5.Create a resource controller

src/main/java/hello/ResourceController.java

package hello;

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

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

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

@RestController

public class ResourceController

{

private static final String template = "%s";

@RequestMapping("/resource")

public Resource resource(@RequestParam(value="name", defaultValue=" ") String name)

{

return new Greeting(String.format(template, name));

}

}

6.Make the application executable

src/main/java/hello/Application.java

package hello;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class Application {

public static void main(String[] args) {

SpringApplication.run(Application.class, args);

}

}

7. Create a single jar file for the project

mvn clean install

7.Run the application

The .jar file is found in <project>/target and run following command

java -jar sample-rest-service-1.0.jar

Note: By default it is running in tomcat server on 8080 port. Make sure that it is free

The jar file can be also used outside the project and can be renamed .

8.Test the service

http://localhost:8080/greeting

{"content":" "}

Provide a request parameter with

http://localhost:8080/greeting?name=User

{"content":"User"}