Building a RESTful Web Service

What you’ll build

You’ll build a service that will accept HTTP GET requests at:

http://localhost:8080/greeting

and respond with a [JSON](https://spring.io/understanding/JSON) representation of a greeting:

{"id":1,"content":"Hello, World!"}

You can customize the greeting with an optional name parameter in the query string:

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

The name parameter value overrides the default value of "World" and is reflected in the response:

{"id":1,"content":"Hello, User!"}

src/main/java/hello/Greeting.java

package hello;

public class Greeting {

private final long id;

private final String content;

public Greeting(long id, String content) {

this.id = id;

this.content = content;

}

public long getId() {

return id;

}

public String getContent() {

return content;

}

}

src/main/java/hello/GreetingController.java

package hello;

import java.util.concurrent.atomic.AtomicLong;

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

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

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

@RestController

public class GreetingController {

private static final String template = "Hello, %s!";

private final AtomicLong counter = new AtomicLong();

@RequestMapping("/greeting")

public Greeting greeting(@RequestParam(value="name", defaultValue="World") String name) {

return new Greeting(counter.incrementAndGet(),

String.format(template, name));

}

}

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);

}

}

## Test the service

Now that the service is up, visit <http://localhost:8080/greeting>, where you see:

{"id":1,"content":"Hello, World!"}

Provide a name query string parameter with <http://localhost:8080/greeting?name=User>. Notice how the value of the content attribute changes from "Hello, World!" to "Hello User!":

{"id":2,"content":"Hello, User!"}