Spring REST Hands-on Part 2 - Answers

**HTTP Request and Response**

Request Format Example:  
GET /hello.txt HTTP/1.1  
User-Agent: curl/7.16.3 libcurl/7.16.3 OpenSSL/0.9.7l zlib/1.2.3  
Host: www.example.com  
  
Response Format Example:  
HTTP/1.1 200 OK  
Date: Mon, 27 Jul 2009 12:28:53 GMT  
Server: Apache  
Content-Type: text/plain  
  
Hello World! My payload includes a trailing CRLF.

**Need and Benefits of RESTful Web Services**

- Lightweight and scalable  
- Stateless communication  
- Based on HTTP methods (GET, POST, PUT, DELETE)  
- Easy to maintain and flexible  
- Widely adopted and compatible with different platforms

**GET Method - Hello World Example**

Controller: HelloController.java  
@RestController  
public class HelloController {  
 private static final Logger LOGGER = LoggerFactory.getLogger(HelloController.class);  
   
 @GetMapping("/hello")  
 public String sayHello() {  
 LOGGER.info("START");  
 LOGGER.info("END");  
 return "Hello World!!";  
 }  
}  
  
Access: http://localhost:8083/hello

**REST - Country Web Service**

Controller: CountryController.java  
@RequestMapping("/country")  
public Country getCountryIndia() throws IOException {  
 ApplicationContext context = new ClassPathXmlApplicationContext("country.xml");  
 Country country = context.getBean("country", Country.class);  
 return country;  
}  
  
Response:  
{  
 "code": "IN",  
 "name": "India"  
}

**REST - Get All Countries**

@GetMapping("/countries")  
public List<Country> getAllCountries() {  
 ApplicationContext context = new ClassPathXmlApplicationContext("country.xml");  
 List<Country> countries = context.getBean("countryList", List.class);  
 return countries;  
}

**REST - Get Country by Code**

@GetMapping("/countries/{code}")  
public Country getCountry(@PathVariable String code) throws CountryNotFoundException {  
 return countryService.getCountry(code.toUpperCase());  
}  
  
// In CountryService.java  
public Country getCountry(String code) throws CountryNotFoundException {  
 List<Country> countries = ...; // load from XML  
 return countries.stream()  
 .filter(c -> c.getCode().equalsIgnoreCase(code))  
 .findFirst()  
 .orElseThrow(() -> new CountryNotFoundException());  
}

**CountryNotFoundException.java**

@ResponseStatus(value = HttpStatus.NOT\_FOUND, reason = "Country not found")  
public class CountryNotFoundException extends Exception {  
}

**MockMVC Testing - Load Controller**

@Autowired  
private CountryController countryController;  
  
@Test  
public void contextLoads() {  
 assertNotNull(countryController);  
}

**MockMVC Testing - Get Country**

@Autowired  
private MockMvc mvc;  
  
@Test  
public void testGetCountry() throws Exception {  
 mvc.perform(get("/country"))  
 .andExpect(status().isOk())  
 .andExpect(jsonPath("$.code").exists())  
 .andExpect(jsonPath("$.code").value("IN"))  
 .andExpect(jsonPath("$.name").value("India"));  
}

**MockMVC Testing - Exception Case**

@Test  
public void testGetCountryException() throws Exception {  
 mvc.perform(get("/country/az"))  
 .andExpect(status().isNotFound())  
 .andExpect(status().reason("Country not found"));  
}