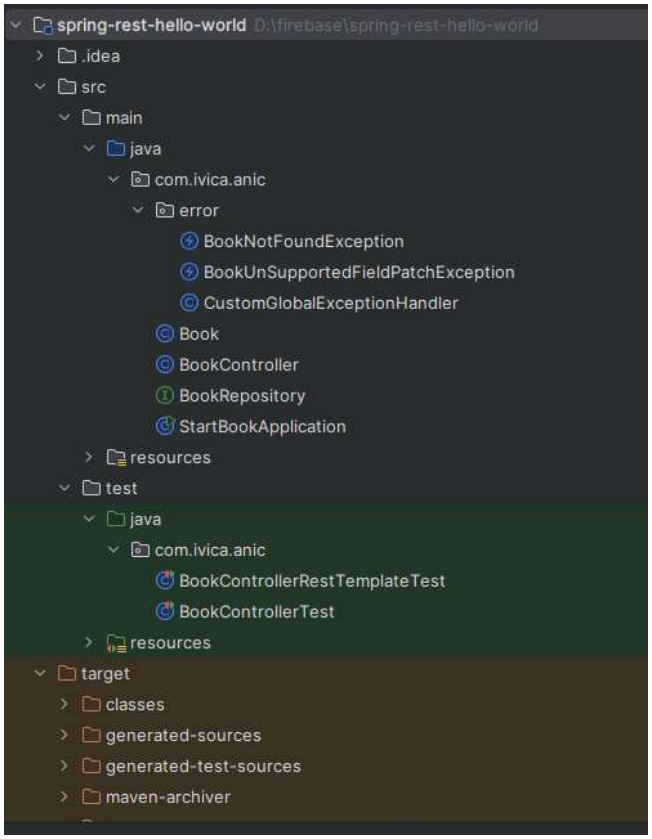
Spring REST Hello World Example Ivica Anic

In this article, we will show you how to develop a Spring Boot REST style web service to handle CRUD operations from a H2 In-memory database.

Technologies used :

- Spring Boot 2.1.2.RELEASE
- Spring 5.1.4.RELEASE
- Spring Data JPA 2.1.4.RELEASE
- H2 In-memory Database 1.4.197
- Tomcat Embed 9.0.14
- JUnit 4.12
- Maven 3
- Java 8

1. Project Directory



2. Maven

Include `spring-boot-starter-web` for Spring MVC and REST structure, `spring-boot-starter-data-jpa` for CRUD repository.

pom.xml

```
<?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>

    <artifactId>spring-rest-hello-world</artifactId>
    <packaging>jar</packaging>
    <name>Spring Boot REST API Example</name>
    <version>1.0</version>

    <parent>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-parent</artifactId>
        <version>2.1.2.RELEASE</version>
    </parent>

    <!-- Java 8 -->
```

```
    <properties>
        <java.version>1.8</java.version>
        <downloadSources>true</downloadSources>
        <downloadJavadocs>true</downloadJavadocs>
    </properties>

    <dependencies>

        <!-- spring mvc, rest -->
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-web</artifactId>
        </dependency>

        <!-- jpa, crud repository -->
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-data-jpa</artifactId
        </dependency>

        <!-- in-memory database  -->
        <dependency>
            <groupId>com.h2database</groupId>
            <artifactId>h2</artifactId>
        </dependency>

        <!-- unit test rest -->
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-test</artifactId>
            <scope>test</scope>
        </dependency>

        <!-- test patch operation need this -->
        <dependency>
            <groupId>org.apache.httpcomponents</groupId>
            <artifactId>httpclient</artifactId>
            <version>4.5.7</version>
            <scope>test</scope>
        </dependency>

        <!-- hot swapping, disable cache for template, enable li
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-devtools</artifactId>
            <optional>true</optional>
        </dependency>

    </dependencies>
```

```
    <build>
        <plugins>

            <plugin>
                <groupId>org.springframework.boot</groupId>
                <artifactId>spring-boot-maven-plugin</artifactId
                <configuration>
                    <addResources>true</addResources>
                </configuration>
            </plugin>

            <plugin>
                <groupId>org.apache.maven.plugins</groupId>
                <artifactId>maven-surefire-plugin</artifactId>
                <version>2.22.0</version>
            </plugin>

        </plugins>

    </build>
</project>
```

Project dependencies.

```
$ mvn dependency:tree
```

```
[INFO] org.springframework.boot:spring-rest-hello-world:jar:1.0
[INFO] +- org.springframework.boot:spring-boot-starter-web:jar:2
[INFO] |  +- org.springframework.boot:spring-boot-starter:jar:2.
[INFO] |  |  +- org.springframework.boot:spring-boot-starter-log
[INFO] |  |  |  +- ch.qos.logback:logback-classic:jar:1.2.3:comp
[INFO] |  |  |  |  \- ch.qos.logback:logback-core:jar:1.2.3:comp
[INFO] |  |  |  +- org.apache.logging.log4j:log4j-to-slf4j:jar:2
[INFO] |  |  |  |  \- org.apache.logging.log4j:log4j-api:jar:2.1
[INFO] |  |  |  \- org.slf4j:jul-to-slf4j:jar:1.7.25:compile
[INFO] |  |  +- javax.annotation:javax.annotation-api:jar:1.3.2:
[INFO] |  |  \- org.yaml:snakeyaml:jar:1.23:runtime
[INFO] |  +- org.springframework.boot:spring-boot-starter-json:j
[INFO] |  |  +- com.fasterxml.jackson.core:jackson-databind:jar:
[INFO] |  |  |  +- com.fasterxml.jackson.core:jackson-annotation
[INFO] |  |  |  \- com.fasterxml.jackson.core:jackson-core:jar:2
[INFO] |  |  +- com.fasterxml.jackson.datatype:jackson-datatype-
[INFO] |  |  +- com.fasterxml.jackson.datatype:jackson-datatype-
[INFO] |  |  \- com.fasterxml.jackson.module:jackson-module-para
[INFO] |  +- org.springframework.boot:spring-boot-starter-tomcat
[INFO] |  |  +- org.apache.tomcat.embed:tomcat-embed-core:jar:9.
[INFO] |  |  +- org.apache.tomcat.embed:tomcat-embed-el:jar:9.0.
[INFO] |  |  \- org.apache.tomcat.embed:tomcat-embed-websocket:j
```

```
[INFO] |  +- org.hibernate.validator:hibernate-validator:jar:6.0
[INFO] |  |  +- javax.validation:validation-api:jar:2.0.1.Final:
[INFO] |  |  +- org.jboss.logging:jboss-logging:jar:3.3.2.Final:
[INFO] |  |  \- com.fasterxml:classmate:jar:1.4.0:compile
[INFO] |  +- org.springframework:spring-web:jar:5.1.4.RELEASE:co
[INFO] |  |  \- org.springframework:spring-beans:jar:5.1.4.RELEA
[INFO] |  \- org.springframework:spring-webmvc:jar:5.1.4.RELEASE
[INFO] |     +- org.springframework:spring-aop:jar:5.1.4.RELEASE
[INFO] |     +- org.springframework:spring-context:jar:5.1.4.REL
[INFO] |     \- org.springframework:spring-expression:jar:5.1.4.
[INFO] +- org.springframework.boot:spring-boot-starter-data-jpa:
[INFO] |  +- org.springframework.boot:spring-boot-starter-aop:ja
[INFO] |  |  \- org.aspectj:aspectjweaver:jar:1.9.2:compile
[INFO] |  +- org.springframework.boot:spring-boot-starter-jdbc:j
[INFO] |  |  +- com.zaxxer:HikariCP:jar:3.2.0:compile
[INFO] |  |  \- org.springframework:spring-jdbc:jar:5.1.4.RELEAS
[INFO] |  +- javax.transaction:javax.transaction-api:jar:1.3:com
[INFO] |  +- javax.xml.bind:jaxb-api:jar:2.3.1:compile
[INFO] |  |  \- javax.activation:javax.activation-api:jar:1.2.0:
[INFO] |  +- org.hibernate:hibernate-core:jar:5.3.7.Final:compil
[INFO] |  |  +- javax.persistence:javax.persistence-api:jar:2.2:
[INFO] |  |  +- org.javassist:javassist:jar:3.23.1-GA:compile
[INFO] |  |  +- net.bytebuddy:byte-buddy:jar:1.9.7:compile
[INFO] |  |  +- antlr:antlr:jar:2.7.7:compile
[INFO] |  |  +- org.jboss:jandex:jar:2.0.5.Final:compile
[INFO] |  |  +- org.dom4j:dom4j:jar:2.1.1:compile
[INFO] |  |  \- org.hibernate.common:hibernate-commons-annotatio
[INFO] |  +- org.springframework.data:spring-data-jpa:jar:2.1.4.
[INFO] |  |  +- org.springframework.data:spring-data-commons:jar
[INFO] |  |  +- org.springframework:spring-orm:jar:5.1.4.RELEASE
[INFO] |  |  +- org.springframework:spring-tx:jar:5.1.4.RELEASE:
[INFO] |  |  \- org.slf4j:slf4j-api:jar:1.7.25:compile
[INFO] |  \- org.springframework:spring-aspects:jar:5.1.4.RELEAS
[INFO] +- com.h2database:h2:jar:1.4.197:compile
[INFO] +- org.springframework.boot:spring-boot-starter-test:jar:
[INFO] |  +- org.springframework.boot:spring-boot-test:jar:2.1.2
[INFO] |  +- org.springframework.boot:spring-boot-test-autoconfi
[INFO] |  +- com.jayway.jsonpath:json-path:jar:2.4.0:test
[INFO] |  |  \- net.minidev:json-smart:jar:2.3:test
[INFO] |  |     \- net.minidev:accessors-smart:jar:1.2:test
[INFO] |  |        \- org.ow2.asm:asm:jar:5.0.4:test
[INFO] |  +- junit:junit:jar:4.12:test
[INFO] |  +- org.assertj:assertj-core:jar:3.11.1:test
[INFO] |  +- org.mockito:mockito-core:jar:2.23.4:test
[INFO] |  |  +- net.bytebuddy:byte-buddy-agent:jar:1.9.7:test
[INFO] |  |  \- org.objenesis:objenesis:jar:2.6:test
[INFO] |  +- org.hamcrest:hamcrest-core:jar:1.3:test
[INFO] |  +- org.hamcrest:hamcrest-library:jar:1.3:test
[INFO] |  +- org.skyscreamer:jsonassert:jar:1.5.0:test
```

```
[INFO] |  |  \- com.vaadin.external.google:android-json:jar:0.0.
[INFO] |  +- org.springframework:spring-core:jar:5.1.4.RELEASE:c
[INFO] |  |  \- org.springframework:spring-jcl:jar:5.1.4.RELEASE
[INFO] |  +- org.springframework:spring-test:jar:5.1.4.RELEASE:t
[INFO] |  \- org.xmlunit:xmlunit-core:jar:2.6.2:test
[INFO] +- org.apache.httpcomponents:httpclient:jar:4.5.7:test
[INFO] |  +- org.apache.httpcomponents:httpcore:jar:4.4.10:test
[INFO] |  \- commons-codec:commons-codec:jar:1.11:test
[INFO] \- org.springframework.boot:spring-boot-devtools:jar:2.1.
[INFO]    +- org.springframework.boot:spring-boot:jar:2.1.2.RELE
[INFO]    \- org.springframework.boot:spring-boot-autoconfigure:
```

3. Spring REST

3.1 A REST controller to create the following REST API endpoints :

HTTP Method	URI	Description
GET	/books	List all books.
POST	/books	Save a book.
GET	/books/{id}	Find a book where id = {:id}.
PUT	/books/{id}	Update a book where id = {:id}, or save it.
PATCH	/books/{id}	Update a single field where id = {:id}.
DELETE	/books/{id}	Delete a book where id = {:id}.

BookController.java

```
package com.ivica.anic;

import com.ivica.anic.error.BookNotFoundException;
import com.ivica.anic.error.BookUnSupportedFieldPatchException;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.util.StringUtils;
import org.springframework.web.bind.annotation.*;

import java.util.List;
```

```
import java.util.Map;

@RestController
public class BookController {

    @Autowired
    private BookRepository repository;

    // Find
    @GetMapping("/books")
    List<Book> findAll() {
        return repository.findAll();
    }

    // Save
    //return 201 instead of 200
    @ResponseStatus(HttpStatus.CREATED)
    @PostMapping("/books")
    Book newBook(@RequestBody Book newBook) {
        return repository.save(newBook);
    }

    // Find
    @GetMapping("/books/{id}")
    Book findOne(@PathVariable Long id) {
        return repository.findById(id)
                .orElseThrow(() -> new BookNotFoundException(id)
    }

    // Save or update
    @PutMapping("/books/{id}")
    Book saveOrUpdate(@RequestBody Book newBook, @PathVariable L

        return repository.findById(id)
                .map(x -> {
                    x.setName(newBook.getName());
                    x.setAuthor(newBook.getAuthor());
                    x.setPrice(newBook.getPrice());
                    return repository.save(x);
                })
                .orElseGet(() -> {
                    newBook.setId(id);
                    return repository.save(newBook);
                });
    }

    // update author only
    @PatchMapping("/books/{id}")
    Book patch(@RequestBody Map<String, String> update, @PathVar
```

```
        return repository.findById(id)
                .map(x -> {

                    String author = update.get("author");
                    if (!StringUtils.isEmpty(author)) {
                        x.setAuthor(author);

                        // better create a custom method to upda
                        return repository.save(x);
                    } else {
                        throw new BookUnSupportedFieldPatchExcep
                    }

                })
                .orElseGet(() -> {
                    throw new BookNotFoundException(id);
                });

    }

    @DeleteMapping("/books/{id}")
    void deleteBook(@PathVariable Long id) {
        repository.deleteById(id);
    }

}
```

4. Spring Data JPA

4.1 A model annotated with JPA annotations.

Book.java

```
package com.ivica.anic;

import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.Id;
import java.math.BigDecimal;

@Entity
public class Book {

    @Id
    @GeneratedValue
    private Long id;
```

```
    private String name;
    private String author;
    private BigDecimal price;

    //setters, getters, constructors...
}
```

4.2 Create a class and extends `JpaRepository`, it contains all the CRUD operations.

BookRepository.java

```
package com.ivica.anic;

import org.springframework.data.jpa.repository.JpaRepository;

// Spring Data magic :)
public interface BookRepository extends JpaRepository<Book, Long
}
```

5. Basic Error Handling

Create a `@ControllerAdvice` to handle the custom exceptions thrown by the controller.

CustomGlobalExceptionHandler

```
package com.ivica.anic.error;

import org.springframework.http.HttpStatus;
import org.springframework.web.bind.annotation.ControllerAdvice;
import org.springframework.web.bind.annotation.ExceptionHandler;
import org.springframework.web.servlet.mvc.method.annotation.Res

import javax.servlet.http.HttpServletResponse;
import java.io.IOException;

@ControllerAdvice
public class CustomGlobalExceptionHandler extends ResponseEntity

    // Let Spring handle the exception, we just override the sta
    @ExceptionHandler(BookNotFoundException.class)
    public void springHandleNotFound(HttpServletResponse respons
        response.sendError(HttpStatus.NOT_FOUND.value());
    }

    @ExceptionHandler(BookUnSupportedFieldPatchException.class)
    public void springUnSupportedFieldPatch(HttpServletResponse
```

```
        response.sendError(HttpStatus.METHOD_NOT_ALLOWED.value()
    }

}
```

BookNotFoundException.java

```
package com.ivica.anic.error;

public class BookNotFoundException extends RuntimeException {

    public BookNotFoundException(Long id) {
        super("Book id not found : " + id);
    }

}
```

BookUnSupportedFieldPatchException.java

```
package com.ivica.anic.error;

import java.util.Set;

public class BookUnSupportedFieldPatchException extends RuntimeE

    public BookUnSupportedFieldPatchException(Set<String> keys)
        super("Field " + keys.toString() + " update is not allow
    }

}
```

6. Spring Boot

6.1 Create a `@SpringBootApplication` class to start the REST web application, and insert 3 predefined book objects into the H2 In-memory database for demo.

StartBookApplication.java

```
package com.ivica.anic;

import org.springframework.boot.CommandLineRunner;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplicat
import org.springframework.context.annotation.Bean;

import java.math.BigDecimal;
```

```
@SpringBootApplication
public class StartBookApplication {

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

        // init bean to insert 3 books into h2 database.
    @Bean
    CommandLineRunner initDatabase(BookRepository repository) {
        return args -> {
            repository.save(new Book("A Guide to the Bodhisattva
            repository.save(new Book("The Life-Changing Magic of
            repository.save(new Book("Refactoring: Improving the
        };
    }
}
```

7. Demo

Start Spring Boot application, and test the REST API endpoints with `curl` command.

```
$ mvn spring-boot:run
```

7.1 Find All – `GET /books`

```
> curl -v localhost:8080/books

*   Trying ::1...
* TCP_NODELAY set
* Connected to localhost (::1) port 8080 (#0)
> GET /books HTTP/1.1
> Host: localhost:8080
> User-Agent: curl/7.55.1
> Accept: <strong>/</strong>
>
< HTTP/1.1 200
< Content-Type: application/json;charset=UTF-8
< Transfer-Encoding: chunked
< Date: Tue, 19 Feb 2019 06:33:57 GMT
<

[
        {"id":1,"name":"A Guide to the Bodhisattva Way of Life",
        {"id":2,"name":"The Life-Changing Magic of Tidying Up","
```

```
        {"id":3,"name":"Refactoring: Improving the Design of Exi
]
```

7.2 Find One – `GET /books/1`

```
> curl -v localhost:8080/books/1

*   Trying ::1...
* TCP_NODELAY set
* Connected to localhost (::1) port 8080 (#0)
> GET /books/1 HTTP/1.1
> Host: localhost:8080
> User-Agent: curl/7.55.1
> Accept: <strong>/</strong>
>
< HTTP/1.1 200
< Content-Type: application/json;charset=UTF-8
< Transfer-Encoding: chunked
< Date: Tue, 19 Feb 2019 06:37:26 GMT
<

{
        "id":1,
        "name":"A Guide to the Bodhisattva Way of Life",
        "author":"Santideva",
        "price":15.41
}
```

7.3 Test 404 – `GET /books/5`

```
> curl -v localhost:8080/books/5

*   Trying ::1...
* TCP_NODELAY set
* Connected to localhost (::1) port 8080 (#0)
> GET /books/5 HTTP/1.1
> Host: localhost:8080
> User-Agent: curl/7.55.1
> Accept: <strong>/</strong>
>
< HTTP/1.1 404
< Content-Type: application/json;charset=UTF-8
< Transfer-Encoding: chunked
< Date: Tue, 19 Feb 2019 06:38:45 GMT
<

{
        "timestamp":"2019-02-19T06:38:45.743+0000",
        "status":404,
```

```
        "error":"Not Found",
        "message":"Book id not found : 5",
        "path":"/books/5"
}
```

7.4 Test Save – `POST /books -d {json}`

```
#Run this on Windows, need \"

> curl -v -X POST localhost:8080/books -H "Content-type:applicat

Note: Unnecessary use of -X or --request, POST is already inferr
*   Trying ::1...
* TCP_NODELAY set
* Connected to localhost (::1) port 8080 (#0)
> POST /books HTTP/1.1
> Host: localhost:8080
> User-Agent: curl/7.55.1
> Accept: <strong>/</strong>
> Content-type:application/json
> Content-Length: 65
>
* upload completely sent off: 65 out of 65 bytes
< HTTP/1.1 201
< Content-Type: application/json;charset=UTF-8
< Transfer-Encoding: chunked
< Date: Tue, 19 Feb 2019 07:33:01 GMT
<
{
        "id":4,
        "name":
        "Spring REST tutorials",
        "author":"ivica.anic",
        "price":9.99
}
```

7.5 Test Update – `PUT /books/4 -d {json}`

```
> curl -v -X PUT localhost:8080/books/4 -H "Content-type:applica

*   Trying ::1...
* TCP_NODELAY set
* Connected to localhost (::1) port 8080 (#0)
> PUT /books/4 HTTP/1.1
> Host: localhost:8080
> User-Agent: curl/7.55.1
> Accept: <strong>/</strong>
> Content-type:application/json
> Content-Length: 59
```

```
>
* upload completely sent off: 59 out of 59 bytes
< HTTP/1.1 200
< Content-Type: application/json;charset=UTF-8
< Transfer-Encoding: chunked
< Date: Tue, 19 Feb 2019 07:36:49 GMT
<
{
        "id":4,
        "name":"Spring Forever",
        "author":"pivotal",
        "price":9.99
}

> curl localhost:8080/books/4
{"id":4,"name":"Spring Forever","author":"pivotal","price":9.99}
```

7.6 Test Update a 'author' field – `PATCH /books/4 -d {json}`

```
> curl -v -X PATCH localhost:8080/books/4 -H "Content-type:appli
*   Trying ::1...
* TCP_NODELAY set
* Connected to localhost (::1) port 8080 (#0)
> PATCH /books/4 HTTP/1.1
> Host: localhost:8080
> User-Agent: curl/7.55.1
> Accept: <strong>/</strong>
> Content-type:application/json
> Content-Length: 19
>
* upload completely sent off: 19 out of 19 bytes
< HTTP/1.1 200
< Content-Type: application/json;charset=UTF-8
< Transfer-Encoding: chunked
< Date: Tue, 19 Feb 2019 07:39:53 GMT

{
        "id":4,
        "name":"Spring Forever",
        "author":"oracle",
        "price":9.99
}
```

7.6.1 Test Update a 'name' field – `PATCH /books/4 -d {json}`

```
> curl -v -X PATCH localhost:8080/books/4 -H "Content-type:appli
*   Trying ::1...
* TCP_NODELAY set
```

```
* Connected to localhost (::1) port 8080 (#0)
> PATCH /books/4 HTTP/1.1
> Host: localhost:8080
> User-Agent: curl/7.55.1
> Accept: <strong>/</strong>
> Content-type:application/json
> Content-Length: 26
>
* upload completely sent off: 26 out of 26 bytes
< HTTP/1.1 405
< Content-Type: application/json;charset=UTF-8
< Transfer-Encoding: chunked
< Date: Tue, 19 Feb 2019 07:40:47 GMT
<
{
        "timestamp":"2019-02-19T07:40:47.740+0000",
        "status":405,
        "error":"Method Not Allowed",
        "message":"Field [name] update is not allow.",
        "path":"/books/4"
}
```

7.7 Test delete – `DELETE /books/4`

```
> curl -v -X DELETE localhost:8080/books/4
```

```
*   Trying ::1...
* TCP_NODELAY set
* Connected to localhost (::1) port 8080 (#0)
> DELETE /books/4 HTTP/1.1
> Host: localhost:8080
> User-Agent: curl/7.55.1
> Accept: <strong>/</strong>
>
< HTTP/1.1 200
< Content-Length: 0
< Date: Tue, 19 Feb 2019 07:44:24 GMT
<
```

```
> curl localhost:8080/books/4
{
        "timestamp":"2019-02-19T07:44:39.432+0000",
        "status":404,
        "error":"Not Found",
        "message":"Book id not found : 4",
        "path":"/books/4"
}
```

```
> curl localhost:8080/books
```

```
[
        {"id":1,"name":"A Guide to the Bodhisattva Way of Life",
        {"id":2,"name":"The Life-Changing Magic of Tidying Up","
        {"id":3,"name":"Refactoring: Improving the Design of Exi
]
```


Download Source Code

> $ git clone https://github.com/IvicaAnic/feature-spring-boot-tests
> $ cd spring-rest-hello-world
> $ mvn spring-boot:run