Sunday, July 31, 2022

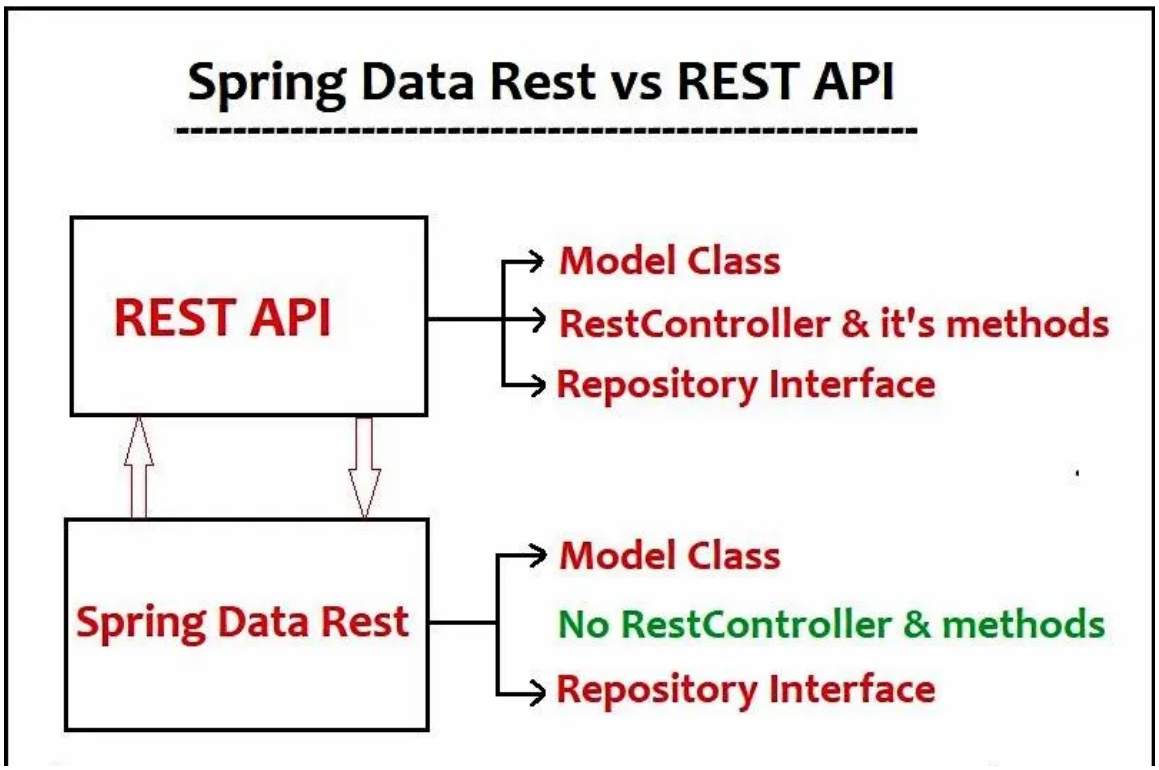
Making Java easy to learn

Java Technology and Beyond

How To Reduce Boilerplate Code In REST API Spring Data REST

java Spring Boot Spring Boot REST Spring Data REST *by devs5003 - October 25, 2020* 4

In previous article on REST API



Development, we have already seen that how we can develop a REST API with minimum lines of code. Still, we have more scope to reduce lines of code using a concept called 'Spring Data REST' in Spring Boot. So, our title of the article is 'How to reduce boilerplate code in REST API Spring Data REST'. In Spring Data REST concept we don't have to write even RestController and its methods. The Spring Data REST dependency will take care of it using HATEOAS (Hypertext as the Engine of Application State).

HATEOAS project is a library of APIs that we can use to easily create REST representations. Spring Data REST provides hyperlinks to access REST operations with the help of HATEOAS only. When some details of a resource are requested, you will provide the resource details as well as details of related resources and the possible actions you can perform on the resource. HATEOAS provides us this type of details. You might have guessed that we will learn Spring Data REST concepts using Spring Boot only. Let's start learning 'How to reduce boilerplate code in REST API Spring Data REST'.

Table of Contents (Click on links below to navigate) [hide]

What is Spring Data REST ?

Spring Data REST is a concept which minimizes a lot of manual work and provides basic implementation of CRUD functionality quite easy. It is built on top of Spring Data Project and also comes under the umbrella Spring Data Project.

How to configure Spring Data REST in Spring Boot Project ?

If you are using Maven tool, you need to add below dependency in your pom.xml which is the main dependency of our project. You will get feature of Spring Data REST from this dependency only. It's very easy to add if you are using Spring Boot Starter Project. You will find the same in steps given in below sections.

```
<dependencies>
     ...
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-data-rest</artifactId>
    </dependency>
     ...
</dependencies>
```

Where can we use Spring Data REST Concept ?

The applications which require only CRUD operations, we can use this concept to generate REST API and get rid of boilerplate codes. As mentioned earlier using this concept we can reduce not only the boilerplate code, but also get rid of memorizing the annotations at Controller and method levels.

What are the steps to generate REST API

using ‘Spring Data REST’ in Spring boot ?

Step #1 : Create Spring Boot Starter Project

If you are new in creating Spring Boot Starter Project, follow internal link.

Step #2 : Add dependencies in Spring Boot Starter Project

Spring Data JPA -> to use JPA repository interfaces
Rest Repositories -> to use Spring Data REST features (Main part of our project)
MySQL Driver -> to get use of MySQL Database
Lombok -> to avoid lots of boilerplate code, for more info you may follow internal link on Lombok.
Spring Boot DevTools -> to avoid restarting tomcat every time we change in the code

If you are using Maven tool, you can check the below dependency in your pom.xml which is the main dependency of our project. As mentioned earlier, you will get features of Spring Data REST from this dependency only.

spring-boot-starter-data-rest

```
<dependencies>
  ...
  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-data-rest</artifactId>
  </dependency>
  ...
</dependencies>
```

Step #3 : Write Model/Entity class

In order to generate the API, we are considering our previous entity ‘Invoice’.

Invoice.java

```java
package com.dev.invoice.rest.entity;

import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.Id;

import lombok.Data;

@Data
@Entity
public class Invoice {

    @Id
    @GeneratedValue
    private Long id;
    private String name;
    private Double amount;
    private Double finalAmount;
    private String number;
    private String receivedDate;
    private String type;
    private String vendor;
    private String comments;
}
```

Step #4 : Write Repository Interface

Interface InvoiceRepository extends from CrudRepository.

InvoiceRepository.java

```
1  package com.dev.invoice.rest.repo;
2
3  import org.springframework.data.repository.CrudRepository;
4
5  import com.dev.invoice.rest.entity.Invoice;
6
7  public interface InvoiceRepository extends CrudRepository<Invoice, Long>
8
9  }
```

Step #5 : Update application.properties file

application.properties

```
1  #  DB Connection Properties
2  spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
3  spring.datasource.url=jdbc:mysql://localhost:3306/REST_INVOICE
4  spring.datasource.username=root
5  spring.datasource.password=devs
6
7  # JPA Properites
8  spring.jpa.show-sql=true
9  spring.jpa.hibernate.ddl-auto=update
10 spring.jpa.database-platform=org.hibernate.dialect.MySQL8Dialect
```

That’s all for the coding part we need to write. Now observe how simple it is !!!

How to test the created REST API using Spring Data REST ?

Step #1 : Run the created Spring Boot Starter Project

To run the application right click on Project and select Run As >> Spring Boot

App.

Step #2 : Enter the URL as http://localhost:8080/ into Browser.

Once you enter the above URL and click enter, you will see the below output.

localhost:8080/


```
{
  "links": {
    "invoices": {
      "href": "http://localhost:8080/invoices"
    },
    "profile": {
      "href": "http://localhost:8080/profile"
    }
  }
}
```


The above output indicates that your URL for REST API will be 'http://localhost:8080/invoices'. The point to observe here is that it has taken 'invoices' in the path from the entity name (Invoice), converts the first letter to lowercase & adds suffix 's'.

Step #3 : Create URL for CRUD operations

Below URLs will work perfectly for the created REST API.

Base URL = http://localhost:8080/invoices

GET URL : http://localhost:8080/invoices [to get all Invoices]

GET URL : http://localhost:8080/invoices/{id} [to get invoice with a particular id]

POST URL: http://localhost:8080/invoices [for saving invoices]

PUT URL : http://localhost:8080/invoices/{id} [to modify an Invoice with a particular id]

DELETE URL : http://localhost:8080/invoices/{id} [to delete an Invoice with a particular id]

Step #4 : Test the created API

One of the best and easiest way to test the REST API is to use 'POSTMAN' software. If you need more details on testing, go through the internal link.

For more details on Spring Data & It's sub-modules, kindly visit spring.io.

Tagged boilerplate code in spring dataresT HATEOAS How to generate REST API without writing RestController ? How to reduce boilerplate code in REST API : Spring Data REST How to work with Spring Data REST ? java spring boilerplate reducing boilerplate code in REST API rest repositories spring boot RESTful without restcontroller spring boot data rest spring boot rest api boilerplate Spring Data REST spring data rest documentation spring data rest example spring data rest reference spring data rest tutorial spring rest api tutorial spring rest mysql example spring rest repositories what is boilerplate code in spring What is HATEOAS ? What is Spring Data REST?

‹ **Previous article**
How to develop REST CRUD API using Spring Boot ?

Next article ›
How to write REST Consumer API using Spring Boot RestTemplate

4 thoughts on "How to reduce boilerplate code in REST API Spring Data REST"

Mark Weichman November 17, 2020 at 10:58 AM

Hmm is anyone else having problems with the images on this blog loading? I'm trying to figure out if its a problem on my end or if it's the blog. Any suggestions would be greatly appreciated.

Reply

November 17, 2020 at 11:35 AM

Devs5003 Please let us know if others are also facing the same problem? We will fix it asap.

Reply

November 20, 2020 at 10:56 PM

Emelina Maxfield Hey there are using WordPress for your blog platform? I'm new to the blog world but I'm trying to get started and create my own. Do you need any html coding expertise to make your own blog? Any help would be really appreciated!

Reply

November 21, 2020 at 12:20 PM

Devs5003 No. not needed.

Reply

Leave a Reply

Name *

Email Address *

Website

nment *

☐ Yes, add me to your mailing list

Post Comment

FOLLOW US

RECENTLY PUBLISHED POSTS

- **Spring Boot Interview Questions**
- **Java Interface**
- **Spring Security Without WebSecurityConfigurerAdapter**
- **Spring Boot Batch Example CSV to MySQL Using JPA**
- **Spring Batch Tutorial**
- **Java 14 Features**
- **Java Features After Java 8**
- **Java Core Tutorial**
- **Spring Cloud Annotations With Examples**
- **How To Add JDK 18 Support in Eclipse**

DO YOU HAVE A QUERY ?

Submit your Query

Email *

Subscribe to receive Updates !