

Blog sobre Java EE

Estás aquí: [Inicio](#)/[Spring](#)/[Spring-REST](#)/[@RepositoryRestResource](#) y Spring Framework

@RepositoryRestResource y Spring Framework

8 noviembre, 2018 por [Cecilio Álvarez Caules](#) — 4 comentarios

El uso de la anotación **@RepositoryRestResource** nos puede ser muy práctica en muchas ocasiones cuando queremos construir arquitecturas REST complejas de una forma rápida dentro de Spring Framework. Normalmente cuando trabajamos con Spring Framework es relativamente común utilizar Spring Data para automatizar la gestión de repositorios. Vamos a ver un ejemplo, para ello en primer lugar mostraremos las dependencias de Maven vía Spring Boot.

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3
3   <modelVersion>4.0.0</modelVersion>
4
5   <groupId>com.example</groupId>
6   <artifactId>demo</artifactId>
7   <version>0.0.1-SNAPSHOT</version>
8   <packaging>jar</packaging>
9
10  <name>demo</name>
11  <description>Demo project for Spring Boot</description>
12
13  <parent>
14    <groupId>org.springframework.boot</groupId>
15    <artifactId>spring-boot-starter-parent</artifactId>
16    <version>2.1.0.RELEASE</version>
17    <relativePath/> <!-- lookup parent from repository -->
18  </parent>
19
20  <properties>
21    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
22    <project.reporting.outputEncoding>UTF-8</project.reporting.outputEn
23    <java.version>1.8</java.version>
24  </properties>
25
26  <dependencies>
27    <dependency>
28      <groupId>org.springframework.boot</groupId>
29      <artifactId>spring-boot-starter</artifactId>
30    </dependency>
31
32    <dependency>
33      <groupId>mysql</groupId>
34      <artifactId>mysql-connector-java</artifactId>
```

JAVA SE

SPRING

JAVA EE

JAVASCRIPT

FRAMEWORKS JS

ARQUITECTURA

MIS LIBROS

MIS CURSOS

```
42     </dependencies>
43
44     <build>
45         <plugins>
46             <plugin>
47                 <groupId>org.springframework.boot</groupId>
48                 <artifactId>spring-boot-maven-plugin</artifactId>
49             </plugin>
50         </plugins>
51     </build>
52
53
54 </project>
```

Una vez definidas las dependencias es momento de ver el contenido del fichero

application.properties que define una serie de parametrizaciones por defecto para JPA y la url de acceso REST para los repositorios.

```
spring.datasource.url=jdbc:mysql://localhost:8889/springjpa
```

```
spring.datasource.username=root
```

```
spring.datasource.password=root
```

```
spring.datasource.driver.class=com.mysql.jdbc.Driver
```

```
spring.jpa.properties.hibernate.dialect = org.hibernate.dialect.MySQL5Dialect
```

```
spring.data.rest.basePath=/webapi
```

Una vez configuradas estas dos cosas el siguiente paso es construir el código necesario para trabajar con JPA y servicios REST. En primer lugar definimos la clase:

```
1  package com.arquitecturajava.springrest;
2
3  import javax.persistence.Entity;
4  import javax.persistence.Id;
5
6  @Entity
7  public class Persona {
8
9      @Id
10     private String nombre;
11     private String apellidos;
12     private int edad;
13     public String getNombre() {
14         return nombre;
15     }
16     public void setNombre(String nombre) {
17         this.nombre = nombre;
18     }
19     public String getApellidos() {
20         return apellidos;
21     }
22     public void setApellidos(String apellidos) {
```

```

30     }
31     public Persona(String nombre, String apellidos, int edad) {
32         super();
33         this.nombre = nombre;
34         this.apellidos = apellidos;
35         this.edad = edad;
36     }
37     public Persona() {
38         super();
39     }
40
41
42 }

```

@RepositoryRestResource

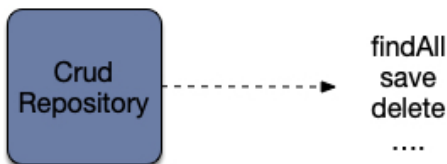
En segundo lugar tenemos que implementar el interface de Spring Data que se encargará de automatizar la creación del repositorio y publicarlo vía REST.

```

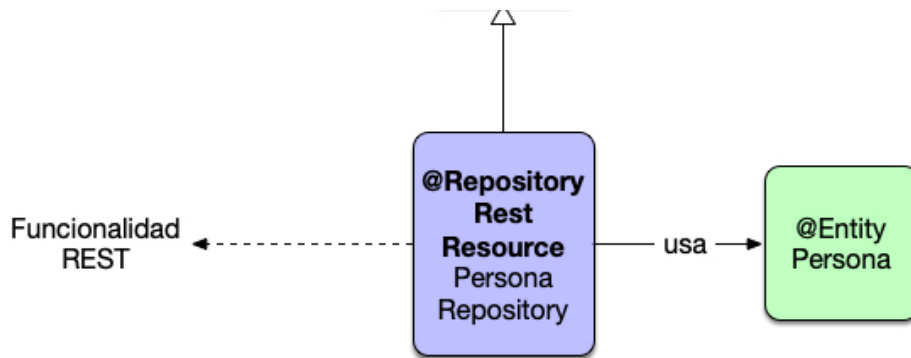
1  package com.arquitecturajava.springrest;
2
3  import org.springframework.data.repository.CrudRepository;
4  import org.springframework.data.rest.core.annotation.RepositoryRestResource
5
6  @RepositoryRestResource(path="/personas")
7  public interface PersonaRepository extends CrudRepository<Persona, String>{
8
9
10
11 }

```

Usamos el interface CrudRepository para implementar de forma automática las operaciones de persistencia más básicas.



Añadimos a este interface la anotación **@RepositoryRestResource** esta anotación se encargará de publicar el repositorio como servicio REST y acceder a las operaciones fundamentales que



Una vez hecho esto el siguiente paso es arrancar la aplicación de Spring Boot y comprobar que en la URL personas accedemos la información de estas sin problema.

```

{
  "_embedded" : {
    "personae" : [ {
      "apellidos" : "sanchez",
      "edad" : 30,
      "_links" : {
        "self" : {
          "href" : "http://localhost:8080/webapi/personas/ana"
        },
        "persona" : {
          "href" : "http://localhost:8080/webapi/personas/ana"
        }
      }
    }, {
      "apellidos" : "perez",
      "edad" : 20,
      "_links" : {
        "self" : {
          "href" : "http://localhost:8080/webapi/personas/pedro"
        },
        "persona" : {
          "href" : "http://localhost:8080/webapi/personas/pedro"
        }
      }
    }
  ],
  "_links" : {
    "self" : {
      "href" : "http://localhost:8080/webapi/personas"
    },
    "profile" : {
      "href" : "http://localhost:8080/webapi/profile/personas"
    }
  }
}

```

Como vemos se accede a la información y además disponemos de links directos a un mayor detalle. El uso de Spring Data REST puede ser muy útil en nuestros proyectos si tenemos enfoque de arquitecturas SPA por ejemplo. Aprendamos a usar este framework y eliminemos muchos de los problemas y dudas que nos aparecen cuando construimos este tipo de arquitecturas.

[JAVA SE](#)[SPRING](#)[JAVA EE](#)[JAVASCRIPT](#)[FRAMEWORKS JS](#)[ARQUITECTURA](#)[MIS LIBROS](#)[MIS CURSOS](#)[3. Spring MVC Flash Attributes](#)[4. Swagger documentando nuestro API REST](#)[5. Arquitecturas REST y sus niveles](#)[6. Spring Data REST](#)

Archivada en: [Spring-REST](#)

Comentarios



Americo Sierra dice

9 noviembre, 2018 en 17:15

Hola Cecilio, con esta anotación se podría simplificar algunos microservicios.

Gracias por compartir tus conocimientos.

[Responder](#)



Cecilio Álvarez Caules dice

9 noviembre, 2018 en 20:27

Claro que sí porque publicarías de forma automática los repositorios que tienes

[Responder](#)



Jose Zafra dice

9 noviembre, 2018 en 0:12

Buen Aporte.

[Responder](#)

[JAVA SE](#)[SPRING](#)[JAVA EE](#)[JAVASCRIPT](#)[FRAMEWORKS JS](#)[ARQUITECTURA](#)[MIS LIBROS](#)[MIS CURSOS](#)

gracias 😊

[Responder](#)

Deja un comentario

Tu dirección de correo electrónico no será publicada. Los campos obligatorios están marcados con *

Comentario

Nombre *

Correo electrónico *

Web

PUBLICAR COMENTARIO

Este sitio usa Akismet para reducir el spam. [Aprende cómo se procesan los datos de tus comentarios.](#)

BUSCAR

JAVA SE

SPRING

JAVA EE

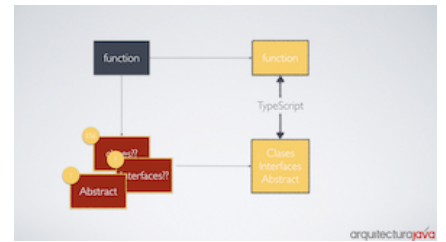
JAVASCRIPT

FRAMEWORKS JS

ARQUITECTURA

MIS LIBROS

MIS CURSOS



Cupon 50%:TYPESCRIPT2018

Mis Cursos de Java Gratuitos

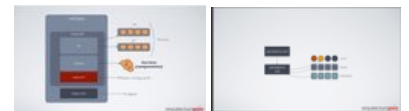
Java Herencia

Java JDBC



Servlets

Intro JPA



Mis Cursos de Java

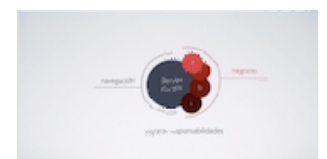
Programación Orientada a Objeto en Java



Java APIS Core



Java Web



Pack Java Core

[JAVA SE](#)[SPRING](#)[JAVA EE](#)[JAVASCRIPT](#)[FRAMEWORKS JS](#)[ARQUITECTURA](#)[MIS LIBROS](#)[MIS CURSOS](#)[Arquitectura Java basada con Spring](#)

POPULAR

[El patrón de inyección de dependencia y su utilidad](#)

[Enterprise Java Beans y su funcionamiento](#)

[Framework vs Librería dos conceptos importantes](#)

[Los Frameworks y su lado oscuro](#)

[¿Qué es un Java Maven Artifact ?](#)

[Utilizando un JSON Generator en Java](#)

[JDBC Prepared Statement y su manejo](#)

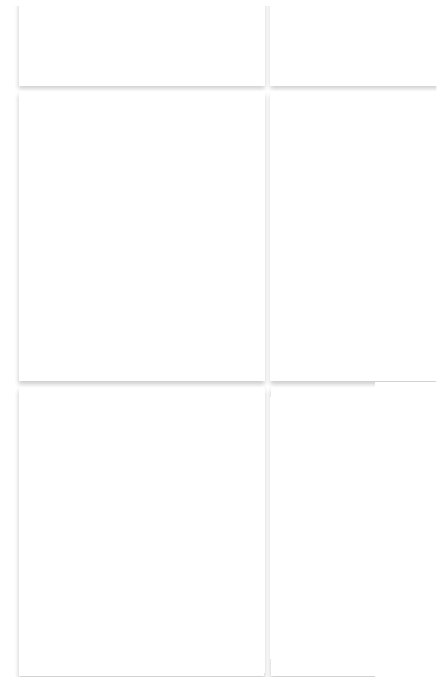
[JPA @Basic , optimizando los fetchings](#)

[Spring Boot JPA y su configuración](#)

[Static Method vs instance method y su uso correcto](#)

CONTACTO

contacto@arquitecturajava.com

[JAVA SE](#)[SPRING](#)[JAVA EE](#)[JAVASCRIPT](#)[FRAMEWORKS JS](#)[ARQUITECTURA](#)[MIS LIBROS](#)[MIS CURSOS](#)

LO MAS LEIDO

[¿Qué es Spring Boot?](#)[Java Constructores this\(\) y super\(\)](#)[Mi Nuevo Curso de Typescript](#)[Usando Java Session en aplicaciones web](#)[@RepositoryRestResource y Spring Framework](#)[Java Iterator vs ForEach](#)[Ejemplo de Java Singleton \(Patrones y ClassLoaders\)](#)[Spring Boot JPA y su configuración](#)[Introducción a Servicios REST](#)[Java Override y encapsulación](#)[Usando el patron factory](#)[Comparando java == vs equals](#)[¿Cuales son las certificaciones Java?](#)

[JAVA SE](#)[SPRING](#)[JAVA EE](#)[JAVASCRIPT](#)[FRAMEWORKS JS](#)[ARQUITECTURA](#)[MIS LIBROS](#)[MIS CURSOS](#)[¿Qué es Struts?](#)[Java 9 JShell y la utilidad de la consola](#)[Static Method vs instance method y su uso correcto](#)[¿Qué es un Microservicio?](#)[Los Frameworks y su lado oscuro](#)[Mis Libros](#)[Spring MVC Configuración \(I\)](#)[El patrón de inyección de dependencia y su utilidad](#)[¿ Que es REST ?](#)

Copyright © 2018 · [eleven40 Pro Theme](#) en [Genesis Framework](#) · [WordPress](#) · [Acceder](#)