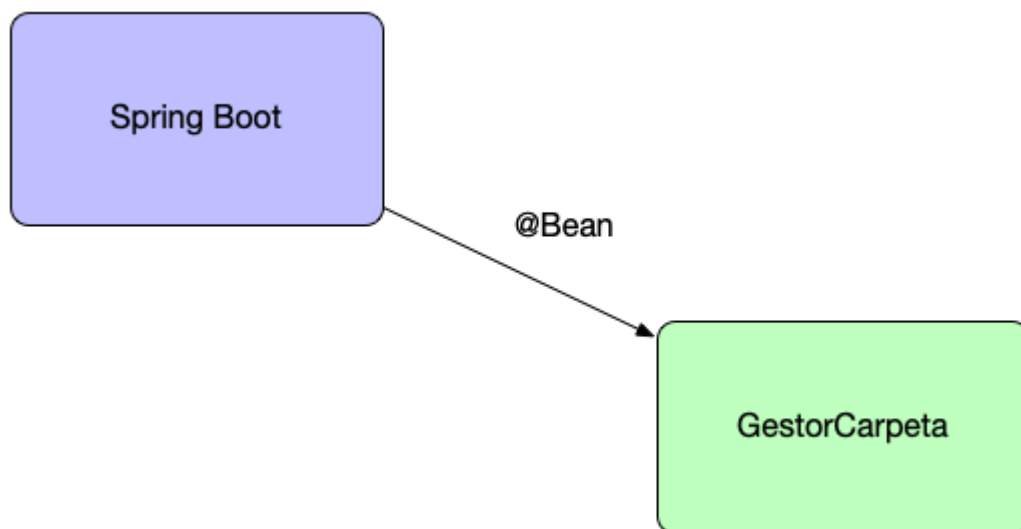


Spring @Bean es una de las anotaciones que más se usa en Spring Framework .

Normalmente nos sirve para configurar beans a medida. Es decir beans que necesitamos instanciar con alguna configuración por defecto . Normalmente cuando generamos Servicios o Repositorios estos no tienen grandes configuraciones que asociar ya que vienen dadas por el application.properties que es el fichero que configura todo por defecto a nivel de Spring Boot sin embargo hay ocasiones que algunos beans que puede ser necesario realizar una configuración a medida para luego usarlos en Servicios o Controllers como clases de apoyo. Vamos a verlo:

Spring @Bean

Lo primero que tendremos que hacer es generarnos ese Bean especial que queremos inicializar de forma personalizada.



El código del bean es :

```
package com.arquitecturajava.bean;

public class GestorCarpeta {

    private String ruta;
```

```

    private String filtro;
    public String getRuta() {
        return ruta;
    }
    public void setRuta(String ruta) {
        this.ruta = ruta;
    }
    public String getFiltro() {
        return filtro;
    }
    public void setFiltro(String filtro) {
        this.filtro = filtro;
    }
    public GestorCarpeta(String ruta, String filtro) {
        super();
        this.ruta = ruta;
        this.filtro = filtro;
        System.out.println("gestor creado");
    }
}

```

Una vez tenemos el bean definido es cuestión de ligarlo con Spring Boot a través de la inicialización usando Spring @bean.

```

package com.arquitecturajava.bean;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.context.annotation.Bean;

@SpringBootApplication
public class BeanApplication {

```

```

    public static void main(String[] args) {
        SpringApplication.run(BeanApplication.class, args);
    }
    @Bean
    public GestorCarpeta crearHilo() {
        return new GestorCarpeta("micarpeta", "mifiltro");
    }
}

```

Como podemos ver el bean se configura con “micarpeta” y “mifiltro” dos de las propiedades que podemos considerar que el gestor carpeta gestiona. Una vez hecho esto podemos usar las capacidades de Spring de Inyección de dependencia para inyectar el gestor a cualquier Servicio o Controlador . En este caso para hacer las cosas sencillas podemos usar un Controller.

```

package com.arquitecturajava.bean;

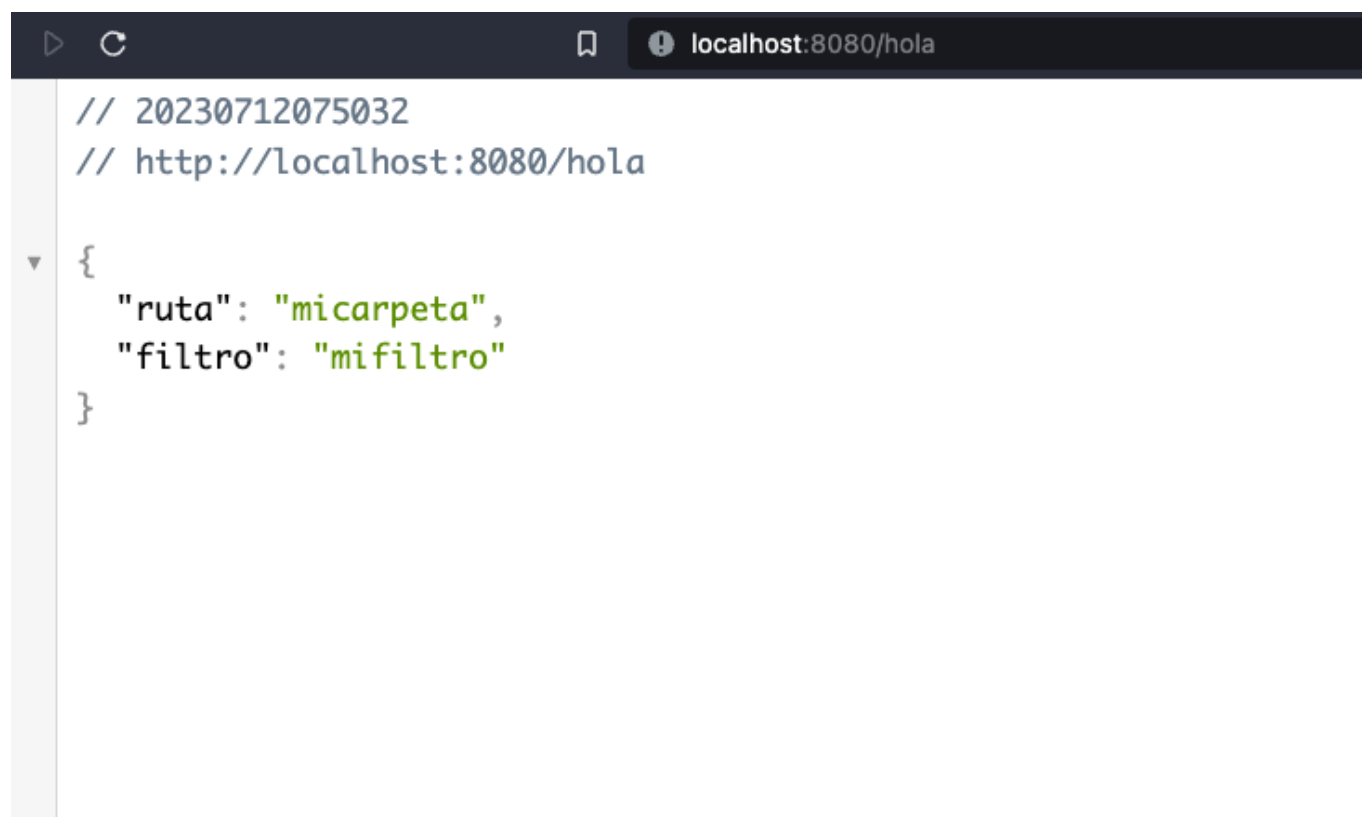
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

@RestController
public class HolaController {

    @Autowired
    GestorCarpeta gestor;
    @RequestMapping("/hola")
    public GestorCarpeta getGestor() {
        return gestor;
    }
}

```

Ya solo nos queda por invocar el servicio desde una url de /hola



```
// 20230712075032
// http://localhost:8080/hola

{
  "ruta": "micarpeta",
  "filtro": "mifiltro"
}
```

Tenemos configurado el @bean correctamente

Otros artículos relacionados

- [Spring Batch Hello World y su configuración](#)
- [Angular router y su configuración](#)
- [Angular resolve y manejo de rutas asincronas](#)
- [¿Que es un Java Bean?](#)
- [Utilizando Spring MVC Bean Validation](#)