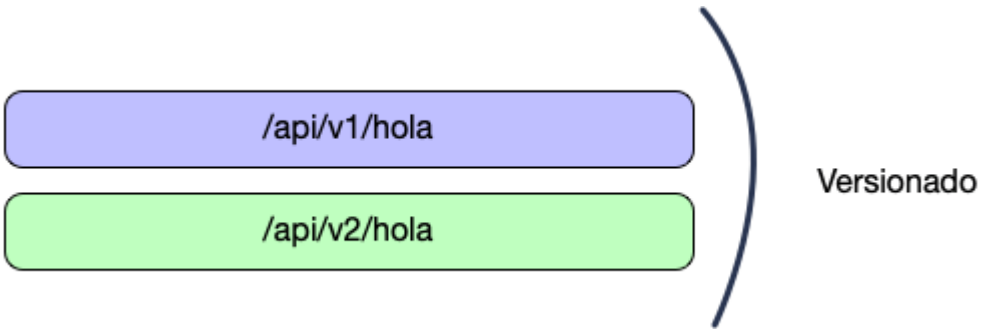
Poco a poco nos vamos acercando a Spring Framework 7 . Hoy por Hoy Spring y Spring Boot son prácticamente los estándares a la hora de desarrollar. En sus nuevas versiones traen alguna cosas interesantes entre las que yo destacaría la mejora del versionado de API REST. Hasta Spring Boot 3 si nosotros queríamos tener varias versiones del api teníamos que diseñar un controlador de la siguiente forma.

```
@RestController
@RequestMapping("/api")
public class ControladorHola {

    @GetMapping("/v1/hola")
    public String obtenerHolaV1() {
        return "Hola Versión 1";
    }

    @GetMapping("/v2/hola")
    public String obtenerHolaV2() {
        return "Hola Versión 2";
    }
}
```

Como podemos ver tenemos una url /api y entro de ella tenemos un versionado con lo cual dependiendo de la versión que solicitemos.



Esto nos puede parecer valido pero con el paso del tiempo tendremos muchísimos endpoints

ya que de entrada muchas urls se duplicaran pero puede ser que en el futuro al api se extienda todavía más y necesitemos de una tercera o cuarta versión como v3 o v4 en este caso puede acabar siendo una locura.

Spring 7 al rescate

A partir de la versión 7 de Spring (con Spring 4) podemos simplificar el manejo de versionado de URLs y mantener solo un EndPoint abierto enviando un parametro que especifique la versión.

```
@RestController
@RequestMapping("/api")
public class ControladorHola {

    @RequestMapping(value = "/hola", version = "1")
    public String obtenerHolaV1() {
        System.out.println("Versión 1");
        return "Versión 1";
    }

    @RequestMapping(value = "/hola", version = "2")
    public String obtenerHolaV2() {
        System.out.println("Versión 2");
        return "Versión 2";
    }
}
```

Como se puede ver ahora @RequestMapping permite el paso de un parámetro adicional y de esta forma podemos versionar el API sin problemas. Eso si tendremos que adjuntar en nuestra petición una cabecera adicional. Esta petición tiene que ser :

```
curl -H "Accept: application/vnd.myapi.v2+json"
```

```
http://localhost:8080/api/hola
```

De esta manera adjuntaremos una cabecera que define la versión 2 y con ello la respuesta será:

```
Versión 2
```

Otros artículos relacionados:

- Las versiones de Java y su historia
- Webinar: Servicio REST con Spring Framework
- ¿Qué es el Versionado Semántico?
- Spring Boot Kotlin y la reducción de código
- ¿Qué es Spring 5 y cuál es su importancia?