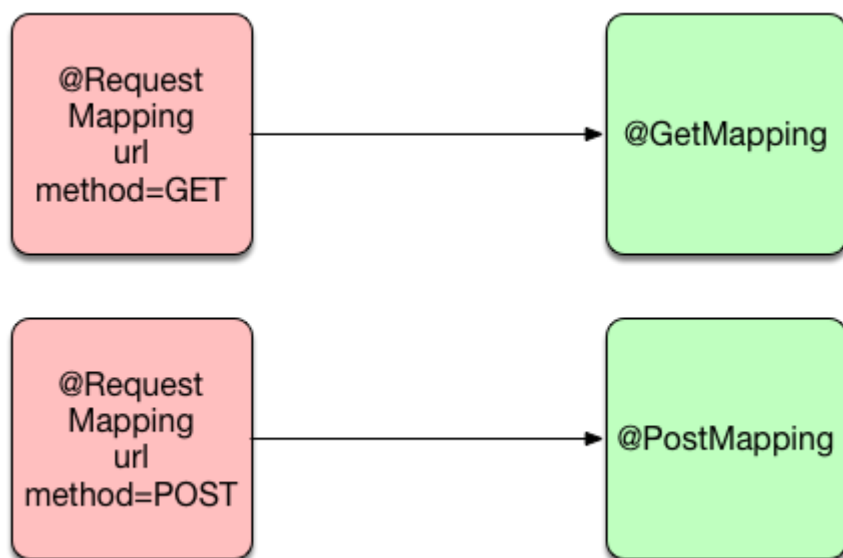


Acabamos de ver como usar Java equals y hashCode

Spring GetMapping es una anotación de Spring Framework que aparece con Spring 4.3 y nos permite simplificar el manejo de los diferentes métodos de Spring MVC y los @RequestMapping que a veces se hacen un poco pesados.



A partir de ahora nosotros podemos usar estas anotaciones y simplificar el código. Para ello vamos a construir un ejemplo utilizando Spring Boot y con las dependencias de MVC y Thymeleaf.

```
<?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>
```

```
mp;gt;4.0.0&amp;amp;amp;lt;/modelVersion&amp;amp;amp;a  
mp;mp;mp;gt;
```

```
&amp;amp;lt;groupId&amp;amp;amp;gt;  
;com.arquitecturajava&amp;amp;lt;/groupId&amp;amp;  
amp;mp;gt;
```

```
&amp;lt;artifactId&amp;amp;amp;  
;gt;spring&amp;lt;/artifactId&amp;amp;  
amp;mp;gt;
```

```
&amp;lt;version&amp;amp;amp;  
;0.0.1-
```

```
SNAPSHOT&amp;lt;/version&amp;amp;a  
mp;mp;gt;
```

```
&amp;lt;packaging&amp;amp;amp;  
gt;jar&amp;lt;/packaging&amp;amp;a  
mp;mp;gt;
```

```
&amp;lt;name&amp;amp;amp;gt;sp  
ring&amp;lt;/name&amp;amp;mp;  
gt;
```

```
&amp;lt;description&amp;amp;am  
p;gt;Demo project for Spring
```

```
Boot&amp;lt;/description&amp;amp;a  
mp;mp;gt;
```

```
&amp;lt;parent&amp;amp;amp;gt;
```

```
&amp;lt;groupId&amp;amp;amp;gt;  
;org.springframework.boot&amp;lt;/groupId&amp;  
amp;amp;mp;gt;
```

```
&amp;lt;artifactId&amp;amp;amp;  
;gt;spring-boot-starter-
```

```
parent&lt;/artifactId&lt;
&gt;
&lt;version&gt;
1.5.8.RELEASE&lt;/version&
&gt;
&lt;relativePath/&
&gt; &lt;!-- lookup parent from
repository --&gt;
&lt;/parent&
;
```

```
&lt;properties&
&gt;
&lt;project.build.sourceEncoding&a
mp;gt;UTF-8&lt;/project.bu
ild.sourceEncoding&
&lt;project.reporting.outputEncoding&a
mp;gt;UTF-8&lt;/projec
t.reporting.outputEncoding&
&lt;java.version&a
mp;gt;1.8&lt;/java.version&
&gt;
&lt;/properties&
&gt;
```

```
&lt;dependencies&
&gt;
&lt;dependency&
&lt;groupId&gt;
org.springframework.boot&lt;/groupId&
```

```
    <!-->
    <artifactId>spring-boot-starter-web</artifactId>
    <!-->
    </dependency>
    <!-->
```

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

```

&lt;/dependency&gt;
&lt;/dependencies&gt;

&lt;build&gt;
&lt;plugins&gt;
&lt;plugin&gt;
&lt;groupId&gt;org.springframework.boot&lt;/groupId&gt;
&lt;artifactId&gt;spring-boot-maven-
plugin&lt;/artifactId&gt;
&lt;/plugin&gt;
&lt;/plugins&gt;
&lt;/build&gt;

&lt;/project&gt;

```

El primer paso es ver el código de la aplicación principal:

```

package com.arquitecturajava.spring;

import org.springframework.boot.SpringApplication;

```

```
import org.springframework.boot.autoconfigure.SpringBootApplication;
import
org.springframework.web.servlet.config.annotation.WebMvcConfigurerAdap
ter;

@SpringBootApplication
public class Application extends WebMvcConfigurerAdapter {

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

Una vez tenemos esto construido el siguiente paso es dar de alta un controlador:

```
package com.arquitecturajava.spring;

import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.ResponseBody;

@Controller
public class HolaController {

    @GetMapping("/hola")
    @ResponseBody
    public String hola() {
```

```

        return "has hecho una peticion get";

    }

    @GetMapping("/formulario")
    public String hola2() {

        return "formulario";

    }

    @PostMapping("/holapost")
    @ResponseBody
    public String holaPost() {

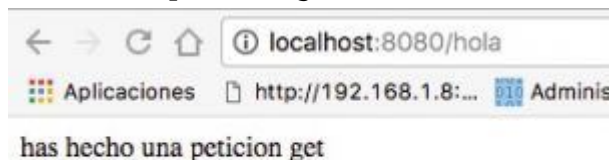
        return "has hecho una peticion post";

    }

}

```

Este controlador hace uso de las nuevas anotaciones de spring @GetMapping ,@PostMapping etc. Como se puede ver si solicitamos la url “/hola” nos devolverá “has hecho una peticion get” . Vamos a arrancar la aplicación de SpringBoot y probarlo.



Funciona perfectamente . Una vez que hemos realizado la primera petición solicitamos la

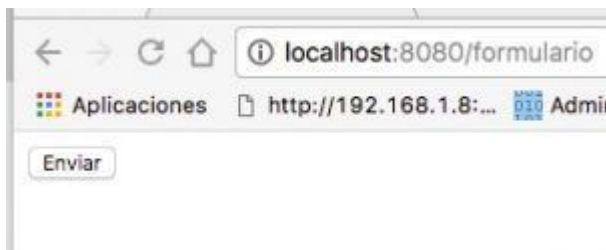
carga del formulario con “/formulario”.

```
&lt;html&gt;
&lt;body&gt;

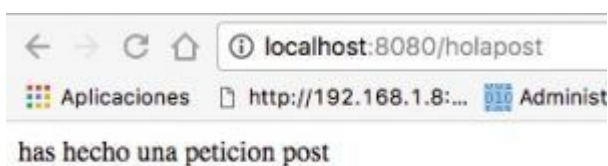
&lt;form action="holapost"
method="post"&gt;
&lt;input
type="submit"/&gt;
&lt;/form&gt;

&lt;/body&gt;
&lt;/html&gt;
```

Esto nos cargará una página html con un botón que realiza una petición post a la url “holapost”.



Pulsamos el botón y nos mostrará el resultado



Acabamos de usar spring GetMapping y PostMapping para trabajar con Spring MVC y

eliminar el uso de `@RequestMapping`.

Otros artículos relacionados

1. [Spring MVC Flash Attributes](#)
2. [Utilizando Spring MVC Bean Validation](#)
3. [Spring MVC Configuración \(I\)](#)
4. [Spring MVC](#)