

Programación Frontend y Backend

BLOQUE SPRING

Spring MVC y Spring REST

01

Spring MVC



GOBIERNO
DE ESPAÑA

MINISTERIO
DE INDUSTRIA, COMERCIO
Y TURISMO



Escuela de
organización
industrial



Unión Europea
Fondo Social Europeo
Iniciativa de Empleo Juvenil
El FSE invierte en tu futuro



Spring MVC

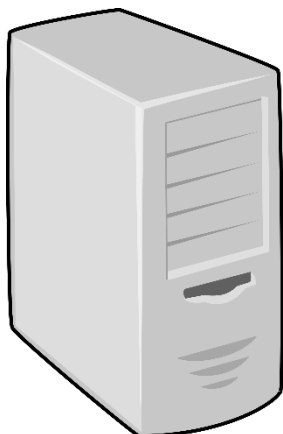
- Patrón Modelo Vista Controlador
- Mapeo de rutas
- Validación
- Manejo de Excepciones
- Interceptores

Spring MVC

Conceptos Básicos



NAVEGADOR
(HTML + CSS + JS)



PRESENTACIÓN

(Controllers, formularios, convertidores, vistas...)

SERVICIOS DE APLICACIÓN

(Services)

DOMINIO

(Modelo, persistencia, repositorios...)



DATA

(SQL, NoSQL, scripts, procedimientos...)



GOBIERNO
DE ESPAÑA

MINISTERIO
DE INDUSTRIA, COMERCIO
Y TURISMO



Escuela de
organización
industrial



Unión Europea
Fondo Social Europeo
Iniciativa de Empleo Juvenil
El FSE invierte en tu futuro



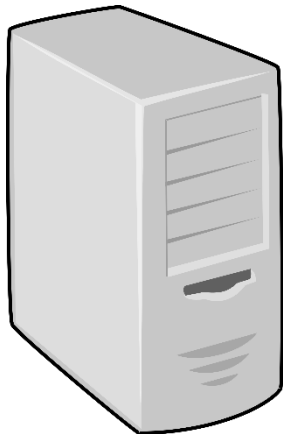
SISTEMA NACIONAL DE
**GARANTÍA
JUVENIL**

Spring MVC

Conceptos Básicos



NAVEGADOR
(HTML + CSS + JS)



Controller

Service

Service

Service

Repository

Repository

Repository



DATA

(SQL, NoSQL, scripts, procedimientos...)



Controlador

Un controlador es la clase encargada de responder las peticiones web

- Se identifican con **@Controller**
- **@RequestMapping** mapea el comienzo de nuestro controlador.
- **@RequestMapping** mapea la ruta con un método.
- **@ResponseBody** el resultado del método se devuelve directamente como texto plano sin requerir una vista. Útil para servicios JSON o XML.

Controlador

@Controller

```
@Controller
@RequestMapping("/")
public class HomeController {
    @RequestMapping("home")
    public @ResponseBody String home() {
        return "hello world";
    }
}
```



GOBIERNO
DE ESPAÑA

MINISTERIO
DE INDUSTRIA, COMERCIO
Y TURISMO



Escuela de
organización
industrial



Unión Europea
Fondo Social Europeo
Iniciativa de Empleo Juvenil
El FSE invierte en tu futuro



SISTEMA NACIONAL DE
**GARANTÍA
JUVENIL**

Controlador – Formas de mapeo (@RequestMapping)

Por ruta:

@RequestMapping("path")

Por método HTTP:

@RequestMapping("path", method=RequestMethod.GET)

Pueden ser: GET, POST, PUT, DELETE, OPTIONS y TRACE

Por presencia de parámetros en la petición o en su cabecera:

@RequestMapping("path", method=RequestMethod.GET, params="foo")

En función de "Content-Type" o "Accept" de la petición:

@RequestMapping("path", consumes="application/json")

Controlador – Obtención de datos de la petición.

Parámetro de la petición (método GET o POST):

@RequestParam("name")

Un grupo de parámetros de la petición en un JavaBean propio:

JavaBean (con getName()/setName() para cada parámetro)

Parámetro de la URL ("/order/{var}"):

@PathVariable("var")

Dato de la cabecera:

@RequestHeader("name")

Valor de una cookie:

@CookieValue("name")

Cuerpo de la petición completo (útil para servicios XML o JSON):

@RequestBody

Vista

Una vista no es más que una plantilla que describe como se debe generar un documento HTML o un fragmento de HTML.



Modelo

Cuando hablamos de Modelo, hablamos de Modelo de persistencia, la implementación que hayamos escogido com JDBC, JPA o Spring Data entre otros.

02

Spring
REST



GOBIERNO
DE ESPAÑA

MINISTERIO
DE INDUSTRIA, COMERCIO
Y TURISMO



Escuela de
organización
industrial



Unión Europea
Fondo Social Europeo
Iniciativa de Empleo Juvenil
El FSE invierte en tu futuro



Servicio Web

Es una tecnología que utiliza un conjunto de protocolos y estándares que sirven para intercambiar datos entre aplicaciones. Las aplicaciones de software desarrolladas en lenguajes de programación diferentes (Angular, JSP, etc), y ejecutadas sobre cualquier plataforma, pueden utilizar los servicios web para intercambiar información.

Tipos de Servicio Web:

- *SOAP*
- *REST*

Servicio Web REST (*Representational State Transfer*)

Es un estilo de arquitectura para desarrollar aplicaciones cliente-servidor centrado en la transferencia de representaciones de recursos a través de peticiones y respuestas.

Se utilizan para crear APIS que utilizarán clientes distribuidos a través de internet, este estilo es apropiado para hacer CRUD.

Servicio Web REST – Principios

- Identificación de recursos Todos los recursos tienen un ID único.

<http://example.com/customers/1234>

<http://example.com/orders/2007/10/776654>

<http://example.com/products>

<http://example.com/processes/salary-increase-234>

<http://example.com/orders/2007/11>

<http://example.com/products?color=green>

Servicio Web REST – Principios

- **Métodos estándar HTTP nos permite:**
 - **GET** – Obtener una representación de recursos.
 - **POST** – Crear un recurso o invocar un proceso.
 - **PUT** – Actualizar un recurso.
 - **DELETE** – Eliminar un recurso.




/orders
GET - list all orders
PUT - unused
POST - add a new order
DELETE - unused

/orders/{id}
GET - get order details
PUT - update order
POST - add item
DELETE - cancel order

/customers
GET - list all customers
PUT - unused
POST - add new customer
DELETE - unused

/customers/{id}
GET - get customer details
PUT - update customer
POST - unused
DELETE - delete customer

/customers/{id}/orders
GET - get all orders for customer
PUT - unused
POST - add order
DELETE - cancel all customer orders

Tipo de recurso	POST	GET	PUT	DELETE	PATCH
/coleccion	Crea un elemento	Lista los elementos de la colección		Borra la colección	
/coleccion/e		Lista datos del elemento	Actualiza datos del elemento	Borra el elemento	Actualiza parcialmente el elemento

@Controller

@RequestMapping(value = "/users")

public class UserController {

@RequestMapping(method = RequestMethod.GET, params = { "page", "size" })

@ResponseBody

public List<User> list(

@RequestParam(value = "page", defaultValue = "1") int page,

@RequestParam(value = "size", defaultValue = "20") int size) {

return userService.list(page,size);

}

@RequestMapping(value =("/{userId}", method = RequestMethod.GET)

@ResponseBody

public User read(@PathVariable(value = "userId") long userId) {

User user = userService.read(userId);

return user;

}

@RequestMapping(method = RequestMethod.POST)

@ResponseBody

public User create(@RequestBody @Valid User user) {

return userService.create(user);

}

...

...

```
@RequestMapping(value = "/{userId}", method = RequestMethod.PUT)
@ResponseStatus(value = HttpStatus.NO_CONTENT)
public void update(@PathVariable(value = "userId") long userId,
                  @RequestBody @Valid User user) {
    userService.update(user);
}

@RequestMapping(value = "/{userId}", method = RequestMethod.DELETE)
@ResponseStatus(value = HttpStatus.NO_CONTENT)
public void delete(@PathVariable(value = "userId") long userId) {
    userService.delete(userId);
}

@ExceptionHandler(Exception.class)
@ResponseStatus(HttpStatus.INTERNAL_SERVER_ERROR)
@ResponseBody
public String handleServerError(Exception ex) {
    return ex.getMessage();
}
}
```

@RestController



@RestController = @Controller + @ResponseBody



GOBIERNO
DE ESPAÑA

MINISTERIO
DE INDUSTRIA, COMERCIO
Y TURISMO



Escuela de
organización
industrial



Unión Europea
Fondo Social Europeo
Iniciativa de Empleo Juvenil
El FSE invierte en tu futuro



@Controller

```
@RequestMapping(value = "/users")
public class UserController {
```

```
    @RequestMapping(method = RequestMethod.GET, params = { "page", "size" })
```

@ResponseBody

```
    public List<User> list(
        @RequestParam(value = "page", defaultValue = "1") int page,
        @RequestParam(value = "size", defaultValue = "20") int size) {
        return userService.list(page,size);
    }
}
```

@RestController

```
@RequestMapping(value = "/users")
public class UserController {
```

```
    @RequestMapping(method = RequestMethod.GET, params = { "page", "size" })
```

```
    public List<User> list(
        @RequestParam(value = "page", defaultValue = "1") int page,
        @RequestParam(value = "size", defaultValue = "20") int size) {
        return userService.list(page,size);
    }
}
```

03

Ejemplo



GOBIERNO
DE ESPAÑA

MINISTERIO
DE INDUSTRIA, COMERCIO
Y TURISMO



Escuela de
organización
industrial



Unión Europea
Fondo Social Europeo
Iniciativa de Empleo Juvenil
El FSE invierte en tu futuro



Repository

```
public interface HelloWorldRepository {  
    public String saluda();  
}
```

```
import org.springframework.stereotype.Repository;
```

```
@Repository
```

```
public class HelloWorldRepositoryImpl implements HelloWorldRepository {  
    public String saluda() {  
        return "Hola";  
    }  
}
```


Service

```
public interface HelloWorldService {  
    public String saluda();  
}
```

```
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.stereotype.Service;  
import es.eoi.springboot.repository.HelloWorldRepository;
```

```
@Service
```

```
public class HelloWorldServiceImpl implements HelloWorldService {
```

```
    @Autowired
```

```
    private HelloWorldRepository repository;
```

```
    public String saluda() {  
        return repository.saluda();  
    }
```

```
}
```



GOBIERNO
DE ESPAÑA

MINISTERIO
DE INDUSTRIA, COMERCIO
Y TURISMO



Escuela de
organización
industrial



Unión Europea
Fondo Social Europeo
Iniciativa de Empleo Juvenil
El FSE invierte en tu futuro



Controller

```
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.http.HttpStatus;  
import org.springframework.http.ResponseEntity;  
import org.springframework.web.bind.annotation.RequestMapping;  
import org.springframework.web.bind.annotation.RequestMethod;  
import org.springframework.web.bind.annotation.RequestParam;  
import org.springframework.web.bind.annotation.RestController;  
import es.eoi.springboot.service>HelloWorldService;
```

```
@RestController  
@RequestMapping(value = "/helloWorld")  
public class HelloWorldController {
```

```
@Autowired  
private HelloWorldService service;
```

...

Controller

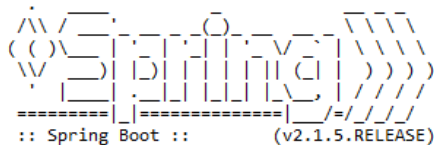
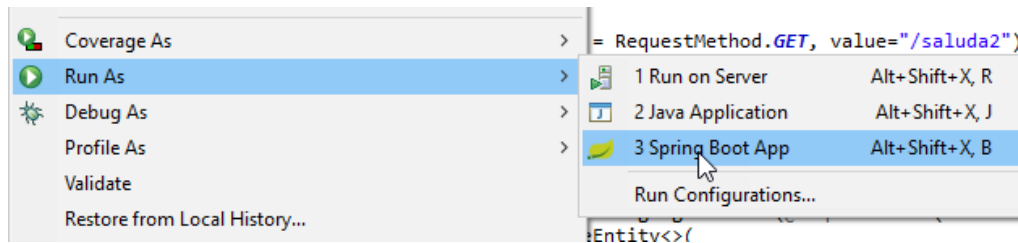
...

```
@RequestMapping(method = RequestMethod.GET, value="/saluda1")  
    public String saluda() {  
        return service.saluda();  
    }
```

```
@RequestMapping(method = RequestMethod.GET, value="/saluda2")  
    public ResponseEntity<String> saludaEntity() {  
        return new ResponseEntity<>(  
            service.saluda(), HttpStatus.OK);  
    }
```

```
@RequestMapping(method = RequestMethod.POST, value="/saluda3")  
    public ResponseEntity<String> getNombre(  
        @RequestParam(value = "name") String name) {  
        return new ResponseEntity<>(  
            service.saluda()  
                .concat(" ")  
                .concat(name), HttpStatus.OK);  
    }
```

Publicación de la Aplicación



```
2019-05-19 12:29:17.128 INFO 12856 --- [
2019-05-19 12:29:17.133 INFO 12856 --- [
2019-05-19 12:29:18.064 INFO 12856 --- [
2019-05-19 12:29:18.081 INFO 12856 --- [
2019-05-19 12:29:18.083 INFO 12856 --- [
2019-05-19 12:29:18.208 INFO 12856 --- [
2019-05-19 12:29:18.208 INFO 12856 --- [
2019-05-19 12:29:18.427 INFO 12856 --- [
2019-05-19 12:29:18.557 INFO 12856 --- [
2019-05-19 12:29:18.559 INFO 12856 --- [
```

```
main] e.e.s.EjemploSpringBootApplication
main] e.e.s.EjemploSpringBootApplication
main] o.s.b.w.embedded.tomcat.TomcatWebServer
main] o.apache.catalina.core.StandardService
main] org.apache.catalina.core.StandardEngine
main] o.a.c.c.C.[Tomcat].[localhost].[/]
main] o.s.web.context.ContextLoader
main] o.s.s.concurrent.ThreadPoolTaskExecutor
main] o.s.b.w.embedded.tomcat.TomcatWebServer
main] e.e.s.EjemploSpringBootApplication
```

```
: Starting EjemploSpringBootApplication on ALC-H0YP0N2 with PID 12856 (C:\Users\
: No active profile set, falling back to default profiles: default
: Tomcat initialized with port(s): 8080 (http)
: Starting service [Tomcat]
: Starting Servlet engine: [Apache Tomcat/9.0.19]
: Initializing Spring embedded WebApplicationContext
: Root WebApplicationContext: initialization completed in 1039 ms
: Initializing ExecutorService 'applicationTaskExecutor'
: Tomcat started on port(s): 8080 (http) with context path ''
: Started EjemploSpringBootApplication in 1.676 seconds (JVM running for 2.701)
```

PUERTO 8080



GOBIERNO
DE ESPAÑA

MINISTERIO
DE INDUSTRIA, COMERCIO
Y TURISMO



Escuela de
organización
industrial



Unión Europea
Fondo Social Europeo
Iniciativa de Empleo Juvenil
El FSE invierte en tu futuro



Prueba de la API REST

Protocolo	Entorno	Puerto	Path		
http	localhost	8080	/helloWorld/saluda1	Controller	Método

http://localhost:8080/helloWorld/saluda1

Prueba de la API REST

← → ↻ ⓘ localhost:8080/helloWorld/saluda1

Hola

← → ↻ ⓘ localhost:8080/helloWorld/saluda2

Hola

Prueba de la API REST

← → ↻ ⓘ localhost:8080/helloWorld/saluda3

Whitelabel Error Page

This application has no explicit mapping for /error, so you are seeing this as a fallback.

Sun May 19 13:01:28 CEST 2019

There was an unexpected error (type=Method Not Allowed, status=405).

Request method 'GET' not supported




Spring REST

Ejemplo


Prueba de la API REST






Boomerang - SOAP & REST Client
Ofrecido por: Ashwin K
Seamlessly integrate and test SOAP & REST services.
★★★★★ 750 Herramientas para desarrolladores

[Probar ahora](#)




Rest Web Service Client
Ofrecido por: Wenkui
REST cliente de servicios Web para desarrollar y probar las API de s
★★★★★ 105 Herramientas para desarrolladores

[Añadir a Chrome](#)



Yet Another REST Client
Ofrecido por: [yet-another-rest-client.com](#)
A free and easy-to-use REST Client. Use it to develop, test and debu
★★★★★ 93 Herramientas para desarrolladores

[Añadir a Chrome](#)



Restlet Client - REST API Testing
Ofrecido por: Talend
Visually create and run single HTTP requests as well as complex scenarios. A
★★★★★ 3.264 Herramientas para desarrolladores

[Valorar](#)

...

Prueba de la API REST

The screenshot shows the REST Client application interface. The left sidebar contains a 'REPOSITORY' section with 'MY DRIVE' and 'EOI' folders. Under 'EOI', there are three items: 'Saluda 1', 'Saluda 2', and 'Saluda 3' (selected). The main area displays the configuration for 'Saluda 3'.

Saluda 3

METHOD: POST

SCHEME :// HOST [":" PORT] [PATH ["?" QUERY]]

URL: `http://localhost:8080/helloWorld/saluda3?name=Javier` (length: 52 bytes)

QUERY PARAMETERS:

- ☒ name = Javier

+ Add query parameter

HEADERS: 1/2

Form | BODY

Buttons: Save, Send, Top, Bottom, 2Request, Copy, Download

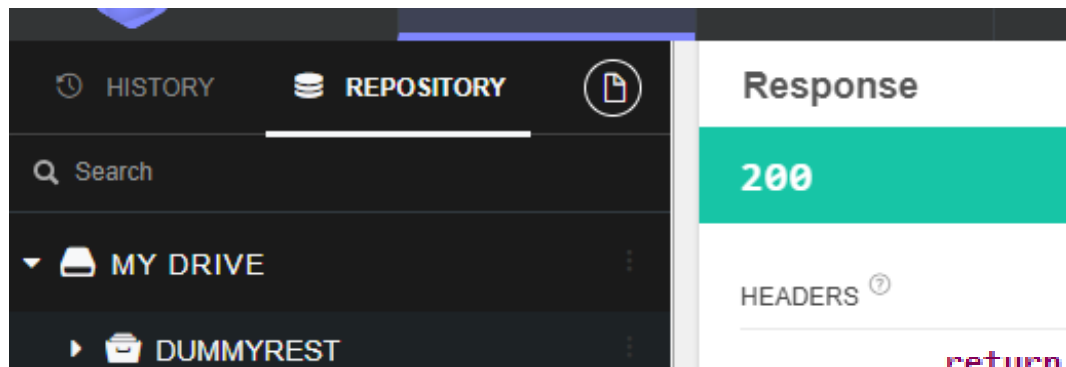


Prueba de la API REST

The screenshot shows a REST client interface with a dark theme. The left sidebar has tabs for 'CLIENT', 'REQUESTS', and 'SCENARIOS'. Under 'REQUESTS', there's a 'REPOSITORY' tab with a search bar and a list of items: 'MY DRIVE', 'DUMMYREST', 'EOI', 'Saluda 1', 'Saluda 2', 'Saluda 3' (selected), and 'EQUIFAX'. The main panel shows a 'Response' for a 200 status code. The headers section lists: 'Content-Type: text/plain;charset=UTF-8', 'Content-Length: 11 bytes', and 'Date: Sun, 19 May 2019 11:07:49 GMT'. The body section shows 'Hola Javier' on a yellow background. At the bottom right, it says 'length: 11 bytes'. The top right of the interface has links for 'Settings', 'Pricing', 'Help', and 'Sign in'.

¡YA TENEMOS NUESTRA API REST CONSTRUIDA, DESPLEGADA Y FUNCIONANDO!

Códigos de Respuesta



HEADERS ?

```
return new ResponseEntity<>(  
    service.saluda()  
        .concat(" ")  
        .concat(name), HttpStatus.OK);
```

<https://developer.mozilla.org/es/docs/Web/HTTP/Status>

Códigos de Respuesta

```
@GetMapping("/age")
ResponseBody<String> age(@RequestParam("yearOfBirth") int yearOfBirth) {

    if (isInFuture(yearOfBirth)) {
        return new ResponseEntity<>("Year of birth cannot be in the future", HttpStatus.BAD_REQUEST);
    }

    return new ResponseEntity<>("Your age is " + calculateAge(yearOfBirth), HttpStatus.OK);
}
```

04

Configuración



GOBIERNO
DE ESPAÑA

MINISTERIO
DE INDUSTRIA, COMERCIO
Y TURISMO



Escuela de
organización
industrial



Unión Europea
Fondo Social Europeo
Iniciativa de Empleo Juvenil
El FSE invierte en tu futuro



Añadimos la siguiente dependencia a las que ya tenemos:

```
<dependency>  
  <groupId>org.springframework.boot</groupId>  
  <artifactId>spring-boot-starter-web</artifactId>  
</dependency>
```


05

Ejercicios



GOBIERNO
DE ESPAÑA

MINISTERIO
DE INDUSTRIA, COMERCIO
Y TURISMO



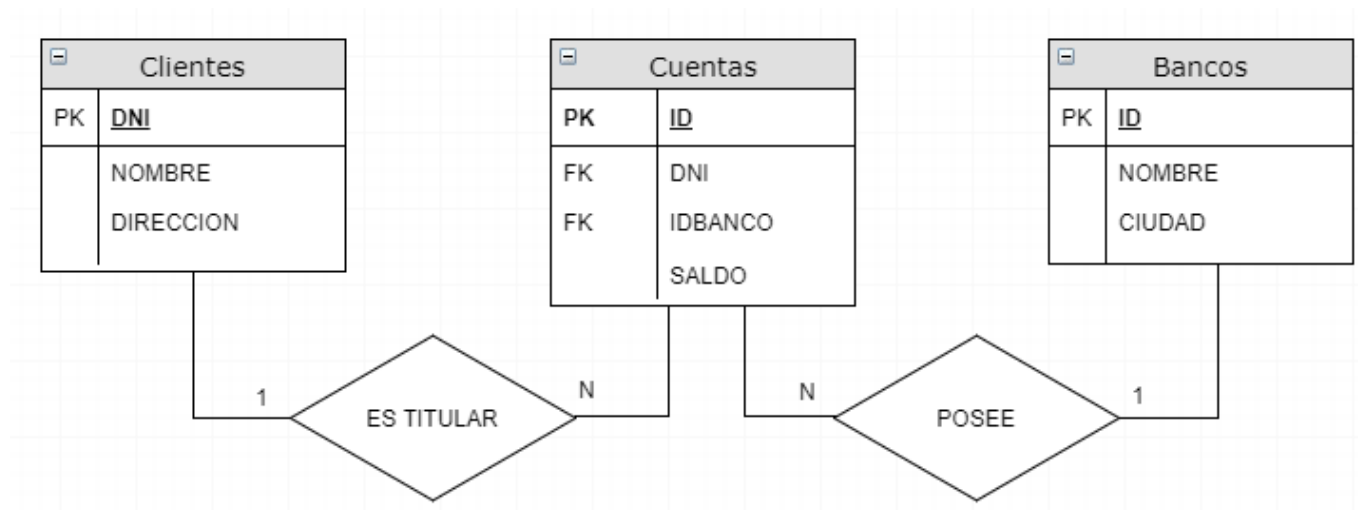
Escuela de
organización
industrial



Unión Europea
Fondo Social Europeo
Iniciativa de Empleo Juvenil
El FSE invierte en tu futuro



Ejercicio – Ampliación del Ejercicio JPA



Ejercicio – Ampliación del Ejercicio JPA

La idea es extraer la clase MundoBancario.java del ejercicio anterior y que toda la funcionalidad se haga a través de una **API REST**, debemos tener en cuenta que debemos crear un proyecto:

Spring Boot + Spring Web + JPA + JAVA 8

Crearemos un Proyecto Spring Boot configurado de la siguiente forma:

Group ID	es.eoi.mundoBancario
Artifact ID	MundoBancarioREST
Name	MundoBancarioREST

Ejercicio – Ampliación del Ejercicio JPA

Nuestra API REST ofrecerá los siguientes servicios:

CientesController

- CRUD Cliente
- findAll()

BancosController

- CRUD Banco
- findAll()

Ejercicio – Ampliación del Ejercicio JPA

Nuestra API REST ofrecerá los siguientes servicios:

CuentasController

- CRUD Cuentas
- findAll()
- findByCliente()
- findByBanco()