

# Swagger con SpringDoc

DigitalHouse>



**Certified Tech  
Developer**

The Ultimate Degree

# Ejemplo de controller

Supongamos que tenemos en nuestra aplicación un controller para administrar libros.

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

    @Autowired
    private BookRepository repository;

    @GetMapping("/{id}")
    public Book findById(@PathVariable long id) {
        return repository.findById(id)
            .orElseThrow(() -> new BookNotFoundException());
    }
}
```

```
@GetMapping("/")
public Collection<Book> findBooks() {
    return repository.getBooks();
}

@PutMapping("/{id}")
@ResponseStatus(HttpStatus.OK)
public Book updateBook(
    @PathVariable("id") final String id, @RequestBody final Book book) {
    return book;
}
}
```

# Pasos

Ahora veremos un ejemplo paso a paso de cómo utilizar la librería SpringDoc.

Ejecutamos nuestra aplicación y nos dirigimos a la URL de Swagger



# 1 - Agregamos la dependencia dentro del POM

```
<dependency>  
  <groupId>org.springdoc</groupId>  
  <artifactId></artifactId>  
  <version>1.5.2</version>  
</dependency>
```

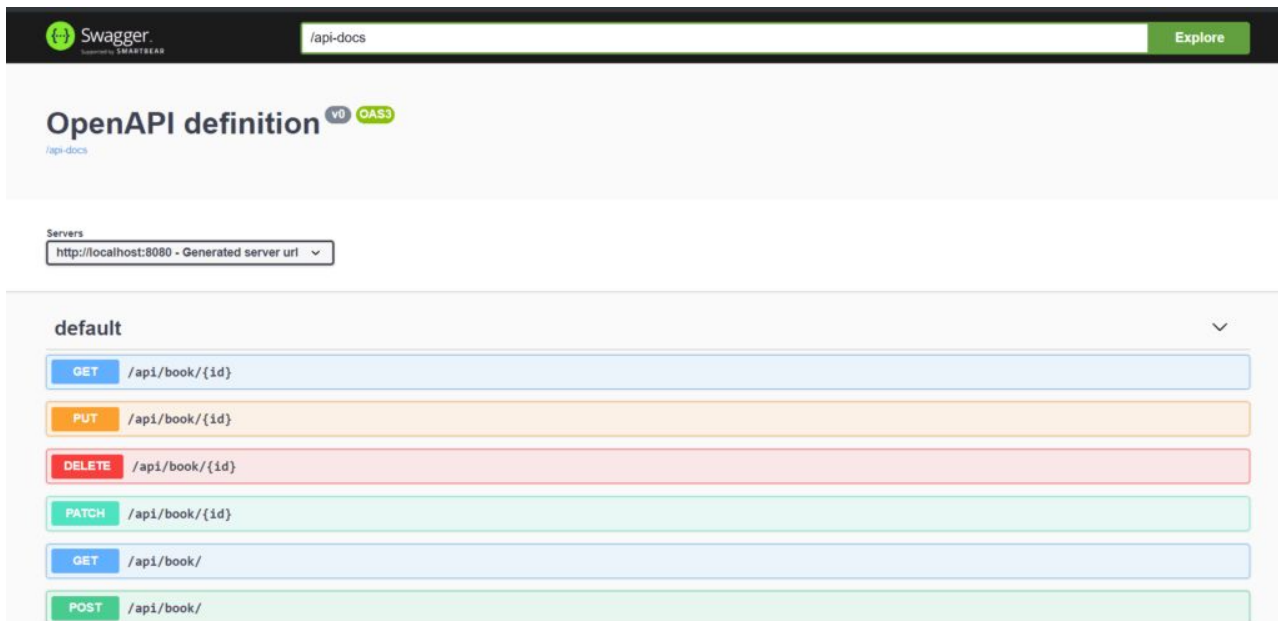
## 2- Ejecutamos nuestra aplicación y nos dirigimos a la URL de Swagger

Por defecto, la interfaz creada por swagger estará en:  
<http://localhost:8080/swagger-ui.html>



# 3 - Ingresamos a la URL

En la URL se encuentra la documentación de nuestra API en /api-docs.



En esta pantalla podemos ver todos los endpoints de nuestra aplicación y el detalle de cómo utilizar cada uno.

The screenshot displays a REST client interface for the endpoint `GET /api/book/`. The interface is divided into several sections:

- Method and Path:** A blue header bar shows the method `GET` and the path `/api/book/`.
- Parameters:** A section titled "Parameters" with a "Try it out" button. It indicates "No parameters" are present.
- Responses:** A section titled "Responses" containing a table of response details.

Code	Description	Links
200	default response	No links

Below the table, there is a "Media type" dropdown menu set to `*/*`, with a note: "Controls Accept header." Below this, there are links for "Example Value" and "Schema". The "Example Value" is displayed in a dark box as a JSON object:

```
[
  {
    "id": 0,
    "title": "string",
    "author": "string"
  }
]
```



# Requisitos en los campos

Podemos agregar requisitos en la documentación de Swagger con anotaciones en nuestras entidades, con `@NotNull`, `@NotBlank`, `@Size`, `@Min`, and `@Max`.

- **@Not null** indica que el campo no puede ser null.
- **@NotBlanck** indica que el campo tiene que tener 1 caracter o más.
- **@Size** se utiliza para indicar el tamaño que puede recibir.

Por ejemplo:

```
public class Book {  
  
    private long id;  
  
    @NotBlank  
    @Size(min = 0, max = 20)  
    private String title;  
  
    @NotBlank  
    @Size(min = 0, max = 30)  
    private String author;  
}
```

En Swagger veríamos:

### Schemas

**Book** ▾ {  
 id  
 title\*  
  
 author\*  
  
}

integer(\$int64)

string

maxLength: 20

minLength: 0

string

maxLength: 30

minLength: 0

DigitalHouse>