**Swagger UI**

A **Swagger UI** egy régóta létező third party library. Arra való, hogy a RESTful webszolgáltatásokhoz egy felületet, egy API dokumentációt lehet vele generálni. Ráadásul ez nem csak egy statikus oldal, hanem még arra is lehetőséget biztosít, hogy a REST webszolgáltatásokat ott ki is tudjuk próbálni. Ezenkívül képes arra is, hogy *OpenAPI Specification* szabványnak megfelelő API leírást generáljon le. Ez egy JSON vagy YAML formátumú egyszerű dokumentum, ami leírja a webszolgáltatásokat. Leírja tehát, hogy az erőforrások milyen URL-en érhetőek el, milyen paramétereket lehet átadni, milyen formátumú adatokat lehet beküldeni, illetve milyen formátumban adja vissza a lekért adatokat, valamint hogy milyen metódusokat használhatunk lekérésre, létrehozásra, módosításra, törlésre. Az OpenAPI Specificatont a Swagger fejlesztői hozták létre, régen ezt *Swagger Specification*-nek hívták.

A Swagger UI keretrendszer-független. Nem kapcsolódik a Springhez, hanem van Spring-integrációja is. A forráskód alapján képes legenerálni a dokumentációt, de ha szeretnénk bele egyéb leírásokat is tenni, akkor ezt annotációk használatával tehetjük meg.

Ezenkívül létezik még a *Springdoc-OpenAPI project*, ami egy integrációs projekt, és arra való, hogy megkönnyítse a Spring és a Swagger UI összekapcsolását. Ha ezt felvesszük függőségként a pom.xml-be az alább látható módon, akkor a Swagger UI is automatikusan használható és elérhető a /swagger-ui.html címen. Az OpenAPI Specificationnek megfelelő dokumentum ekkor a /v3/api-docs címen JSON formátumban, a /v3/api-docs.yaml pedig YAML formátumban érhető el.

<dependency>

<groupId>org.springdoc</groupId>

<artifactId>springdoc-openapi-ui</artifactId>

<version>1.5.9</version>

</dependency>

**http://localhost:8080/swagger-ui/index.html**

Van lehetőség ezt a dokumentumot személyre szabni is. A globális részeket (a dokumentum elejét) úgy lehet testreszabni, hogy egy @Configuration annotációval ellátott osztályban felveszünk egy OpenAPI típusú beant (@Bean annotációval ellátva), és ennek az info() metódusával lehet megadni például a dokumentum címét, verziószámát vagy akár egy hosszabb szöveges leírást is.

@Bean

**public** OpenAPI customOpenAPI() {

**return** **new** OpenAPI()

.info(**new** Info()

.title("Employees API")

.version("1.0.0")

.description("Operations with employees"));

}

A Swagger UI figyelembe veszi a Bean Validation annotációkat is, tehát például egy olyan attribútumot, amelyen a @NotNull annotáció szerepel, piros csillaggal jelöl meg a dokumentációban, hogy azt nem lehet üresen hagyni. Személyreszabható az is a @Schema annotációval, hogy az egyes attribútumok mellett mi jelenjen meg névként a felhasználói felületen, sőt, egy értéket is meg lehet adni példaként. Ezt a példa értéket a Swagger bele fogja generálni a dokumentációba is, és ez segítségünkre lehet az API kipróbálásában is.

**public** **class** CreateEmployeeCommand {

@NotNull

@Schema(description="name of the employee", example = "John Doe")

**private** String name;

}

A Swagger figyelembe veszi a Spring MVC annotációkat is, például egy @GetMapping annotációval ellátott metódusnál a felületen is az fog megjelenni, hogy egy HTTP GET metódussal lekérdezhető erőforrásról van szó. Meg lehet viszont mondani azt, hogy ne jelenjen meg a dokumentációban az osztály neve (pl. EmployeesController), hanem a @Tag annotációval lehet megadni egyedi elnevezést. Magáról az adott metódusról az @Operation annotációval tudunk megadni rövidebb és hosszabb leírást. Meg lehet adni azt is, hogy a metódus milyen státuszkódokkal térjen vissza. Itt a Swagger figyelembe veszi a Spring MVC konfigurációt, vagyis tudja, hogy ez a metódus sima lefutás esetén milyen státuszkóddal tér vissza, és ezt külön definiálás nélkül is belegenerálja a dokumentációba. De @ApiResponse annotációval felvehetjük azt is, hogy a metódus milyen további státuszkódokkal képes még visszatérni, ezek mit jelentenek, és a dokumentációban természetesen ez is meg fog jelenni.

@RestController

@RequestMapping("/api/employees")

@Tag( name = "Operations on employees")

**public** **class** EmployeesController {

@GetMapping("/{id}")

@Operation(summary = "Find employee by id",

description = "Find employee by id.")

@ApiResponse(responseCode = "404",

description = "Employee not found")

**public** EmployeeDto findEmployeeById(

@Parameter(description = "Id of the employee",

example = "12")

@PathVariable("id") long id) {

*// ...*

}

}

**Kérdések**

*Mi a Swagger UI és mire való?*

*Mit ír le? Ez milyen szabványnak felel meg?*

*Hogy tudunk*

*Milyen függőséget érdemes felvenni, hogy megkönnyítsük a springes integrációját?*

*Milyen segítséget kapunk ezzel és milyne urlen?*

*Hogy tudjuk a dokumentot személyre szabni?*

*Figyelembe veszi-e a Swagger a Bean Validaitonöket? Ha igen, hogy jelöli őket?*

*Mire jó a @Schema annotáció?*

*Hogy adunk meg egyedi elnevezést az osztályoknak?*

*Hogy adunk meg egyedi elnevezést a a http metódusokkal elérhető metódusoknak?*

*Hogy adhatjuk meg, hogy a metódus milyen további státuszkódokkal képes még visszatérni?*