Swagger

Swagger je alat koji ćemo iskoristiti za dokumentaciju našeg Restful web servisa. Kako bismo ubacili swagger u našu aplikaciju prvo treba da ubacimo **dva dependency-ja u pom.xml** fajl. Na centralnom maven-ovom repozitorijumu možete naći te dependency-je https://mvnrepository.com/artifact/io.springfox/springfox-swagger2/2.9.2 i https://mvnrepository.com/artifact/io.springfox/springfox-swagger-ui/2.9.2.

```
<dependency>
        <groupId>io.springfox</groupId>
        <artifactId>springfox-swagger2</artifactId>
        <version>2.9.2</version>
</dependency>
<dependency>
        <groupId>io.springfox</groupId>
        <artifactId>springfox-swagger-ui</artifactId>
        <version>2.0.2</version>
</dependency>
```

Kada su se biblioteke skinule i povezale sa projektom, potrebno je napraviti **klasu Swagger** u rva paketu. Iznad deklaracije klase dodajemo dve anotacije: **@Configuration** koja govori da je ova klasa konfiguraciona klasa što znaci da se u njoj mogu definisati novi bean-ovi. Druga anotacija je **@EnableSwagger2** koja omogućava upotrebu Swagger-a.

```
@EnableSwagger2
@Configuration
public class Swagger {

}
```

Zatim dodajemo jednu **konstantu tipa *Contact***.

Možete otvoriti deklaraciju ove klase i videti koje podatke treba da unesete u konstruktor (ne treba pisati ovaj konstruktor, već ga samo pozvati prilikom kreiranja konstante).

```
public Contact(String name, String url, String email) {
  this.name = name;
  this.url = url;
  this.email = email;
}
```

Deklaracija konstante tipa *Contact* u klasi *Swagger*.

```
public static final Contact DEFAULT_CONTACT = new Contact("Dejana
        Gladic", "https://github.com/DejanaGladic","dejanagladic@uns.ac.rs");
```

Sledeca **konstanta** koju pravimo je **tipa *ApiInfo.***

Možete otvoriti deklaraciju te klase i videti šta treba da definišete za konstruktor (ne treba pisati ovaj konstruktor, već ga samo pozvati prilikom kreiranja konstante).

```
public ApiInfo(
    String title,
    String description,
    String version,
    String termsOfServiceUrl,
    Contact contact,
    String license,
    String licenseUrl,
    Collection<VendorExtension> vendorExtensions) {
  this.title = title;
  this.description = description;
  this.version = version;
  this.termsOfServiceUrl = termsOfServiceUrl;
  this.contact = contact;
  this.license = license;
  this.licenseUrl = licenseUrl;
  this.vendorExtensions = newArrayList(vendorExtensions);
}
```

Za nas su bitna samo polja title, description, version i contact. Deklaracija konstante tipa *ApiInfo* u klasi *Swagger*.

```
public static final ApiInfo DEFAULT_API_INFO = new ApiInfo("Backend RVA Swagger ",
        "Razvoj viseslojnih aplikacija", "1.0", "", DEFAULT_CONTACT, "", "",
        new ArrayList<VendorExtension>());
```

Za kraj definisemo **metodu api()** koja vraća objekat tipa *Docket*. Iznad nje stavljamo anotaciju @**Bean** što znači da će povratna vrednost ove funkcije postati bean. Želimo da dokumentujemo samo api-je iz rva paketa.
Definisanje metode u klasi *Swagger*.

```
@Bean
Docket api() {
        return new Docket(DocumentationType.SWAGGER_2).select().
                apis(RequestHandlerSelectors.basePackage("rva")).build().apiInfo(DEFAULT_API_INFO);
}
```

Kada u browseru ukucate http://localhost:8083/swagger-ui.html trebalo bi da vidite dokumentaciju vašeg REST servisa.

Dokumentacija treba da bude što jasnija i deskriptivnija tako da ćemo dodati **opise za svaki kontroler i za svaku metodu**.

Iznad kontrolera stavljamo antoaciju @**Api** koja opisuje kontroler.

```
@RestController
@Api(tags = {"Artikl CRUD operacije"})
public class ArtiklRestController {}
```

Iznad metode stavljamo anotaciju @**ApiOperation** sa opisom metode.

```
@GetMapping("artikl")
@ApiOperation(value = "Vraća kolekciju svih artikala iz baze podataka")
        public Collection<Artikl> getArtikli() {}
```

Krajnji rezultat je:

Ovo treba uraditi za svaki kontroler.