**Giới thiệu**

Trong bài viết trước:

1. [RESTful API Document Tạo với Spring Boot + Swagger](https://loda.me/res-tful-api-document-tao-voi-spring-boot-swagger-loda1576222065972)

Tôi đã giới thiệu lí do vì sao cần Document, và cách tạo ra nó nhanh chóng với Swagger 2.

Trong bài này, tôi sẽ giới thiệu thêm phiên bản tiếp theo, một tiêu chuẩn mới nhất về RESTful document đó là OpenApi 3.0.

Tại sao k đặt tên là Swagger 3? câu chuyện đằng sau nó là việc SmartBear mua lại Swagger, và đổi tên Swagger Specification thành OpenApi Specification, bắt đầu tạo ra các tiêu chuẩn mới cho xây dựng Document, tuy nhiên vẫn xây dựng trên Swagger core.

OpenApi 3.0 là bản mới nhất được ra mắt tại thời điểm viết bài này.

Trong bài có đề cập các kiến thức:

1. [Spring Boot](https://loda.me/spring-boot-0-series-lam-chu-spring-boot-zero-to-hero-loda1558963914472)
2. [jpa](https://loda.me/spring-boot-11-huong-dan-spring-boot-jpa-my-sql-loda1558687596060)
3. [lombok](https://loda.me/general-huong-dan-su-dung-lombok-giup-code-java-nhanh-hon-69-loda1552789752787)

**cài đặt**

*pom.xml*

<?xml version="1.0" encoding="UTF-8"?>

<projectxmlns="http://maven.apache.org/POM/4.0.0"xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"xsi:schemaLocation="http://maven.apache.org/POM/4.0.0https://maven.apache.org/xsd/maven-4.0.0.xsd"><modelVersion>4.0.0</modelVersion><parent><groupId>org.springframework.boot</groupId><artifactId>spring-boot-starter-parent</artifactId><version>2.0.5.RELEASE</version><relativePath/> <!-- lookup parent from repository -->

</parent><groupId>me.loda.spring</groupId><artifactId>example-independent-maven-spring-project</artifactId><version>0.0.1-SNAPSHOT</version><name>example-independent-maven-spring-project</name><description>Demo project for Spring Boot</description><properties><java.version>1.8</java.version></properties><dependencies><dependency><groupId>org.springframework.boot</groupId><artifactId>spring-boot-starter-web</artifactId></dependency><dependency><groupId>org.springframework.boot</groupId><artifactId>spring-boot-devtools</artifactId><scope>runtime</scope><optional>true</optional></dependency><dependency><groupId>org.projectlombok</groupId><artifactId>lombok</artifactId><optional>true</optional></dependency><dependency><groupId>org.springframework.boot</groupId><artifactId>spring-boot-starter-test</artifactId><scope>test</scope></dependency><!--https://mvnrepository.com/artifact/org.springdoc/springdoc-openapi-core -->

<dependency><groupId>org.springdoc</groupId><artifactId>springdoc-openapi-core</artifactId><version>1.1.49</version></dependency><dependency><groupId>org.springdoc</groupId><artifactId>springdoc-openapi-ui</artifactId><version>1.1.49</version></dependency><!--spring jpa-->

<dependency><groupId>org.springframework.boot</groupId><artifactId>spring-boot-starter-data-jpa</artifactId></dependency><!--in memory database-->

<dependency><groupId>com.h2database</groupId><artifactId>h2</artifactId><scope>runtime</scope></dependency></dependencies><build><plugins><plugin><groupId>org.springframework.boot</groupId><artifactId>spring-boot-maven-plugin</artifactId></plugin></plugins></build></project>

springdoc là một thư viện java implementation của OpenApi Specification 3.0.

springdoc-openapi-core chứa core của swagger, giúp chúng ta khai báo document cho api.

springdoc-openapi-ui giúp chúng ta biểu diễn tài liệu dưới dạng web view, dễ nhìn và test.

**Tổng quan OpenApi 3.0**

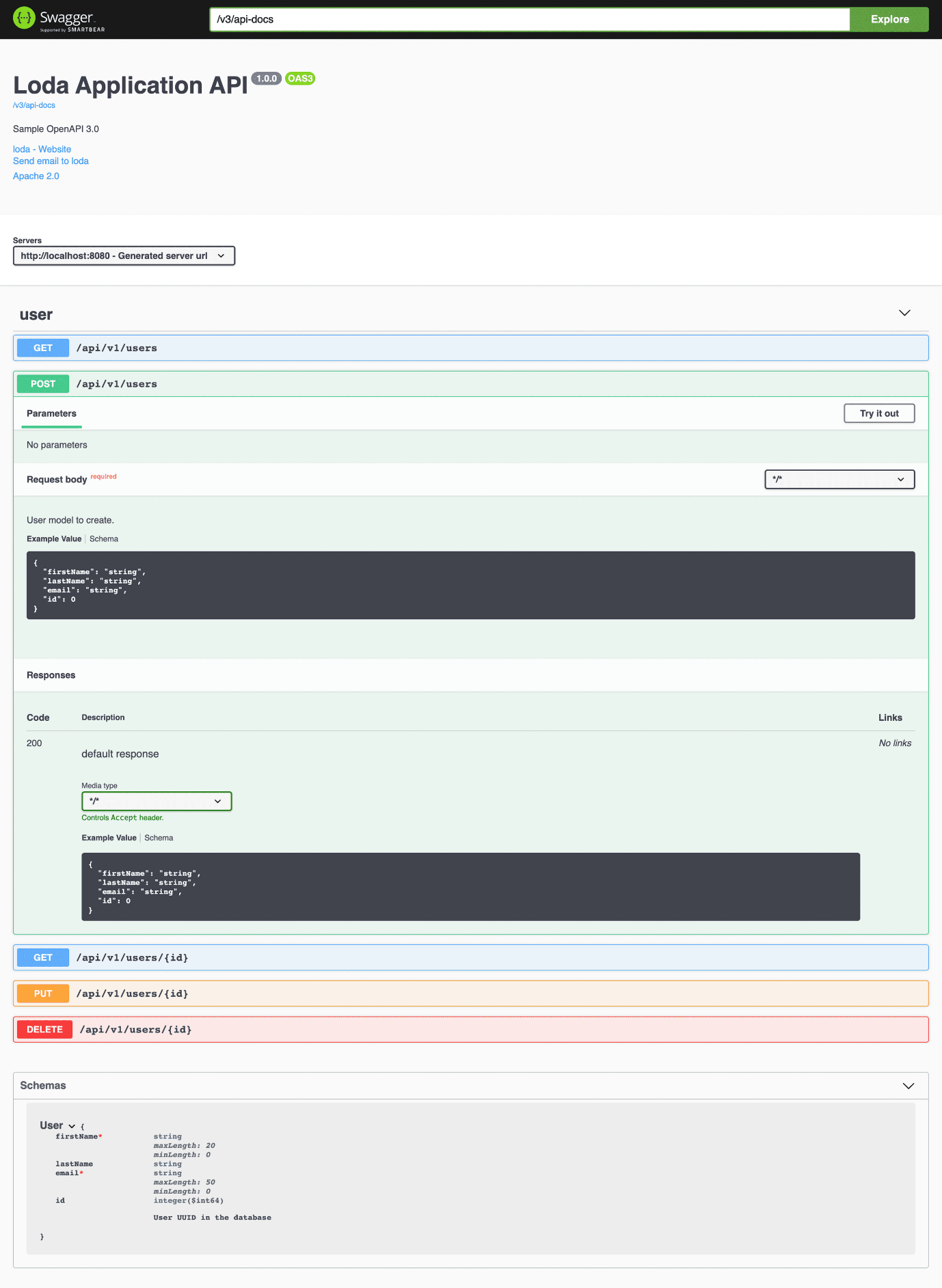
OpenApi 3.0 kế thừa và đổi mới khá nhiều các thành phần của Swagger2, khiến nó tường minh và dễ đọc hơn.

| SWAGGER2 | OPENAPI 3.0 | DESCRIPTION |
| --- | --- | --- |
| [@Api](https://loda.me/4fa00f8e6c9f44019fdf934acbf90a40) | @Tag | Đánh dấu 1 class là nơi chứa các API |
| [@ApiModel](https://loda.me/65c18ee584d64756976b6d742e15552b) | không còn | Đánh dấu 1 class là Swagger Model |
| [@ApiModelProperty](https://loda.me/033b99627676468bb783dec44962fa89) | @Schema | Bổ sung các thông tin cho |
| [@ApiOperation](https://loda.me/ea8e99c5fd1a4431864a2e35c27a2984) | @Operation | Mô tả cho một API và response của nó |
| [@ApiParam](https://loda.me/6ef866fad64c44a8a4ffe8335398f858) | @Parameter | Mô tả các parameter |
| [@ApiResponse](https://loda.me/db837ce32e3141df971b8ac3d3993c31) | @ApiResponse | Mô tả status code của response |
| [@ApiResponses](https://loda.me/b0ec8f2a659e4c298894788fa0ec81b5) | @ApiResponses | Mô tả danh sách các status code của response |

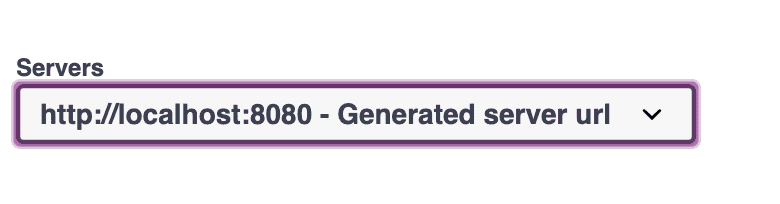
Có thể xem thêm tại: <https://github.com/OAI/OpenAPI-Specification/blob/master/versions/3.0.0.md>

Chúng ta đi vào thực hành thử nhé.

Đại loại sau khi làm xong, chúng ta sẽ có 1 web view document như thế này:



Ngoài đẹp hơn, mô tả chi tiết và dễ sử dụng hơn, thì có 1 phần rất hay, đó là chúng ta có thể chọn Server Url khi test API



**Prepare**

Tạo ra class Model

*User.java*

@Data

@Entity

@Table

@NoArgsConstructor

@AllArgsConstructor

public class User {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

@Schema(description = "User UUID in the database")

@JsonProperty("id")

private Long id;

private @NotBlank @Size(min = 0, max = 20) String firstName;

private String lastName;

private @NotBlank @Size(min = 0, max = 50) String email;

}

*UserRepository.java*

@Repository

public interface UserRepository extends JpaRepository<User, Long> {

}

Khi đã có được Model và Repository, chúng ta sẽ tạo Controller để thao tác liên quan tới User nhé.

Điểm khác biệt lớn nhất của OpenApi3 so với Swagger2 là hệ thống Annotation rất là dày đặc, đa phần các giá trị được biểu diễn và ăn khớp với nhau bằng Annotation và Class.

@RestController

@RequestMapping("/api/v1")

@RequiredArgsConstructor

@Tag(name = "user")

public class UserController {

private final UserRepository userRepository;

@Operation(description = "Xem danh sách User", responses = {

@ApiResponse(content = @Content(array = @ArraySchema(schema = @Schema(implementation = User.class))), responseCode = "200") })

@ApiResponses(value = {

@ApiResponse(responseCode = "200", description = "Thành công"),

@ApiResponse(responseCode = "401", description = "Chưa xác thực"),

@ApiResponse(responseCode = "403", description = "Truy cập bị cấm"),

@ApiResponse(responseCode = "404", description = "Không tìm thấy")

})

@GetMapping("/users")

public List<User> getAllUsers() {

return userRepository.findAll();

}

@GetMapping("/users/{id}")

public User getUser(@PathVariable("id") Long id) {

return userRepository.findById(id).orElse(new User());

}

@PostMapping("/users")

public User createUser(

@Valid

@Parameter(description = "User model to create.", required = true, schema = @Schema(implementation = User.class))

@RequestBody User user) {

return userRepository.save(user);

}

@PutMapping("/users/{id}")

public User updateUser(@PathVariable("id") Long id, @Valid @RequestBody User user) {

user.setId(id);

return userRepository.save(user);

}

@DeleteMapping("/users/{id}")

public void deleteUser(@PathVariable("id") Long id) {

userRepository.deleteById(id);

}

}

Âu khê, Chỉ cần như vậy là chạy được rồi.

Nếu đầy đủ hơn, thì bạn nên config OpenApi cho dự án của chúng ta để bổ sung các thông tin tổng quan.

**Config OpenApi 3.0**

Chỉ cần tạo ra đối tượng OpenAPI và cung cấp các thông tin cần thiết.

@Configuration

public class OpenApiConfig {

@Bean

public OpenAPI customOpenAPI() {

return new OpenAPI()

// Thiết lập các server dùng để test api

.servers(Lists.newArrayList(

new Server().url("http://localhost:8080"),

new Server().url("https://user.loda.me")

))

// info

.info(new Info().title("Loda Application API")

.description("Sample OpenAPI 3.0")

.contact(new Contact()

.email("loda.namnh@gmail.com")

.name("loda")

.url("https://loda.me/"))

.license(new License()

.name("Apache 2.0")

.url("http://www.apache.org/licenses/LICENSE-2.0.html"))

.version("1.0.0"));

}

}

OpenApi3.0 thì bạn không cần @Enable mà chỉ cần add springdoc-openapi-core vào dependencies thôi là nó tự động gen ra tài liệu rồi.

@SpringBootApplication

public class App {

public static void main(String[] args) {

SpringApplication.run(App.class, args);

}

}

Bây giờ, chạy thử và vào địa chỉ <http://localhost:8080/swagger-ui.html> để xem thành quả nhé.

**Kết**

Tới đây bạn có thể sử dụng **OpenAPI 3.0** thoải mái rồi.