spring-boot集成swagger

让程序员又爱又恨的是什么？当然是文档啦，使用没文档的黑盒子想骂娘，自己却压根都不想为自己的代码写文档。使用swagger可以自动为spring-boot api工程生成文档，再加些注解优化一下，一份完整友好的文档就搞定了。

在spring-boot工程里配置swagger

这里以gradle为例(你为什么还在用xml去配置工程?)

build.gradle

多加一行swagger的依赖

```
dependencies {

    compile('io.springfox:springfox-swagger2:2.6.1')

}
```

SwaggerConfig

创建SwaggerConfig类

```
@Configuration

@EnableSwagger2

public class SwaggerConfig {

  @Bean

  public Docket api() {

    return new Docket(DocumentationType.SWAGGER_2)

        .groupName("nap-api") // group name

        .genericModelSubstitutes(DeferredResult.class)

        .useDefaultResponseMessages(false)

        .forCodeGeneration(true)

        .apiInfo(apiInfo())

        .select()
```

```
                .apis(RequestHandlerSelectors.any())
                .paths(Predicates.not(PathSelectors.regex("/error"))) // 排除框架自动生成的error api
                .build();


    }


    private ApiInfo apiInfo() {
        return new ApiInfoBuilder()
                .title("NAP API Documentation")
                .description("Network Automation Platform API Documentation, generated by swagger2")
                .build();
    }
}
```

获取swagger json

运行你的spring-boot工程, 然后访问

```
http://localhost:8081/${context-path}/v2/api-docs?group=${groupName}
```

将返回的json内容粘贴保存到文件 test.json

使用swagger-ui渲染

json的阅读体验并不好，使用swagger-ui渲染得到最终的文档

下载swagger-ui

```
git clone https://github.com/swagger-api/swagger-ui.git
```

创建node express工程

config.json

```
{

  "name": "nap-doc",
```

```
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "",
  "license": "ISC"
}
```

index.js

```
var express = require('express');
var app = express();
app.use('/static', express.static('public'));
app.get('/', function (req, res) {
        res.send('Hello World!');
});



app.listen(3000, function () {
        console.log('API app listening on port 3000!');
});
```

运行

```
npm install express
mkdir public
cp -r ${path-to-swagger-ui}/dist/* public/
cp ${path-to-test-json} public/
#修改 public/index.html中的url为 "/static/test.json"
```

```
node index.js
```

运行之后访问下面的url就可以看到swagger-ui渲染test.json的结果

```
http://localhost:3000/static/index.html
```

一些添加注释的例子

api注释

```
@ApiOperation(value = "对登录信息做校验", notes = "jwt token包含在response header里的Authorization字段里")

@RequestMapping(value="", method = RequestMethod.POST)

public ResponseEntity<User> login(@RequestBody User user, HttpServletResponse response) throws Exception {

}
```

model注释

```
@ApiModelProperty(value = "凭据名", required = true)

@Column(name = "NAME", unique = true)

@NotNull

private String name;
```

隐藏model里的某个属性

```
@ApiModelProperty(hidden = true)

@Column(name = "DELETED")

@NotNull

private Boolean deleted;
```

在文档中添加http request header信息

```
@ApiOperation(value = "获取用户列表")

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

@PreAuthorize("hasRole('ROLE_ADMIN')")
```

```
public ResponseEntity<List<User>> list(@RequestHeader(value =
JwtWebTokenUtil.HEADER_STRING, defaultValue = "") String authorization) throws Exception {

    logger.info("Asking for list");

    return ResponseEntity.ok(us.getUsersByDeleted(false));

}
```