本项目还集成spring security,所以比较麻烦

因为spring security会拦截对应的css、js文件等

1. 使用swagger,首先在pom中引入jar依赖。

|  |
| --- |
| <dependency>  <groupId>io.springfox</groupId>  <artifactId>springfox-swagger2</artifactId>  <version>2.2.2</version>  </dependency>  <dependency>  <groupId>io.springfox</groupId>  <artifactId>springfox-swagger-ui</artifactId>  <version>2.2.2</version>  </dependency> |

2、创建Swagger2配置类

|  |
| --- |
| */\*\*  \** ***@Auther:*** *zj  \** ***@Date:*** *2018/6/24 22:07  \** ***@Description:添加Swagger2配置类*** *\*/* @Configuration @EnableSwagger2 //@EnableSwagger2来启动swagger注解 public class Swagger2 {   @Bean  public Docket createRestApi() {  return new Docket( DocumentationType.*SWAGGER\_2*)  .apiInfo(apiInfo())  .select()  .apis(RequestHandlerSelectors.*any*()) // 对所有api进行监控  .paths(PathSelectors.*any*()) // 对所有路径进行监控  .build();  }   private ApiInfo apiInfo() {  return new ApiInfoBuilder()  .title("Spring Boot中使用Swagger2构建RESTful APIs")  .contact("Zj")  .version("1.0")  .build();  }  } |

如上代码所示，通过@Configuration注解，让Spring来加载该类配置。再通过@EnableSwagger2注解来启用Swagger2。

再通过createRestApi函数创建Docket的Bean之后，apiInfo()用来创建该Api的基本信息（这些基本信息会展现在文档页面中）。select()函数返回一个ApiSelectorBuilder实例用来控制哪些接口暴露给Swagger来展现，本例采用指定扫描的包路径来定义，Swagger会扫描该包下所有Controller定义的API，并产生文档内容（除了被@ApiIgnore指定的请求）。

3、

#### 添加文档内容

在完成了上述配置后，其实已经可以生产文档内容，但是这样的文档主要针对请求本身，而描述主要来源于函数等命名产生，对用户并不友好，我们通常需要自己增加一些说明来丰富文档内容。如下所示，我们通过@ApiOperation注解来给API增加说明、通过@ApiImplicitParams、@ApiImplicitParam注解来给参数增加说明

类似于这样：

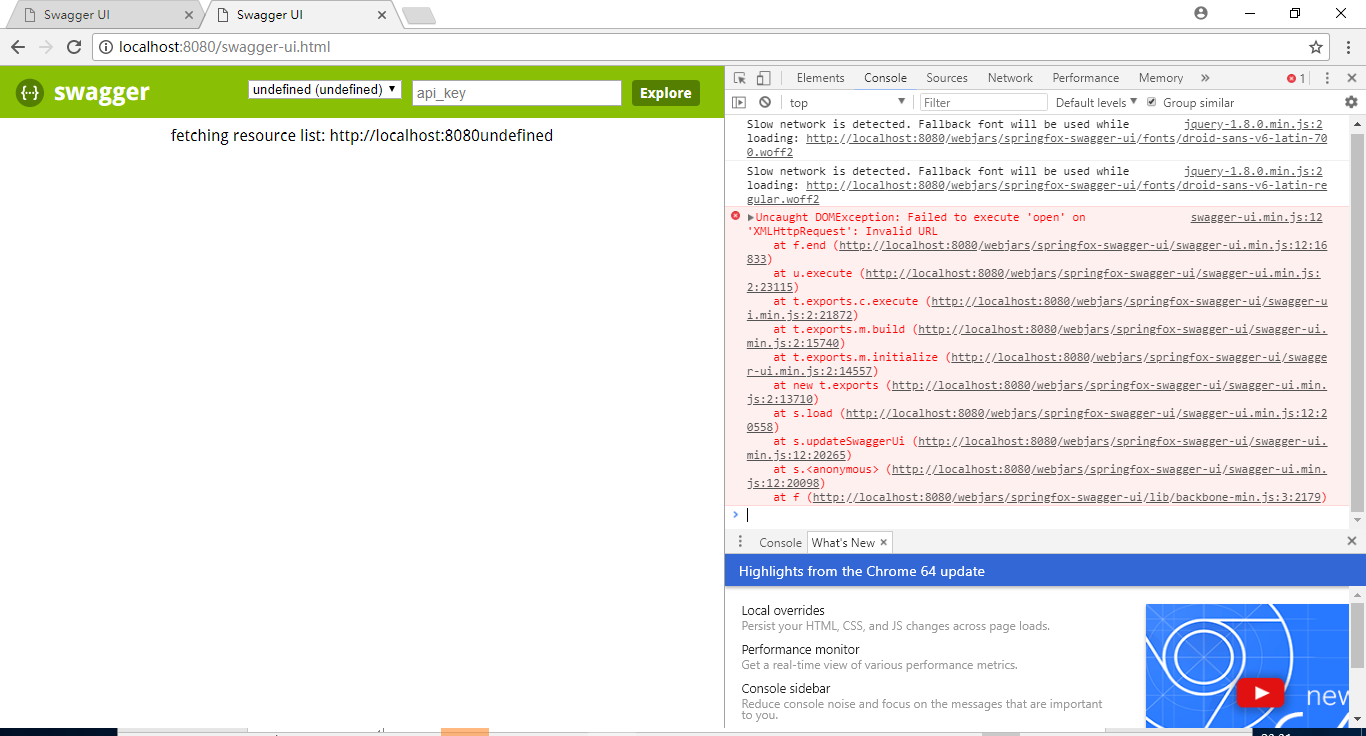


说明：

* @Api：用在类上，说明该类的作用
* @ApiOperation：用在方法上，说明方法的作用
* @ApiImplicitParams：用在方法上包含一组参数说明
* @ApiImplicitParam：用在@ApiImplicitParams注解中，指定一个请求参数的各个方面
  + paramType：参数放在哪个地方
    - header-->请求参数的获取：@RequestHeader
    - query-->请求参数的获取：@RequestParam
    - path（用于restful接口）-->请求参数的获取：@PathVariable
    - body（不常用）
    - form（不常用）
  + name：参数名
  + dataType：参数类型
  + required：参数是否必须传
  + value：参数的意思
  + defaultValue：参数的默认值
* @ApiResponses：用于表示一组响应
* @ApiResponse：用在@ApiResponses中，一般用于表达一个错误的响应信息
  + code：数字，例如400
  + message：信息，例如"请求参数没填好"
  + response：抛出异常的类
* @ApiModel：描述一个Model的信息（这种一般用在post创建的时候，使用@RequestBody这样的场景，请求参数无法使用@ApiImplicitParam注解进行描述的时候）
  + @ApiModelProperty：描述一个model的属性

4、访问

<http://localhost:8080/swagger-ui.html>

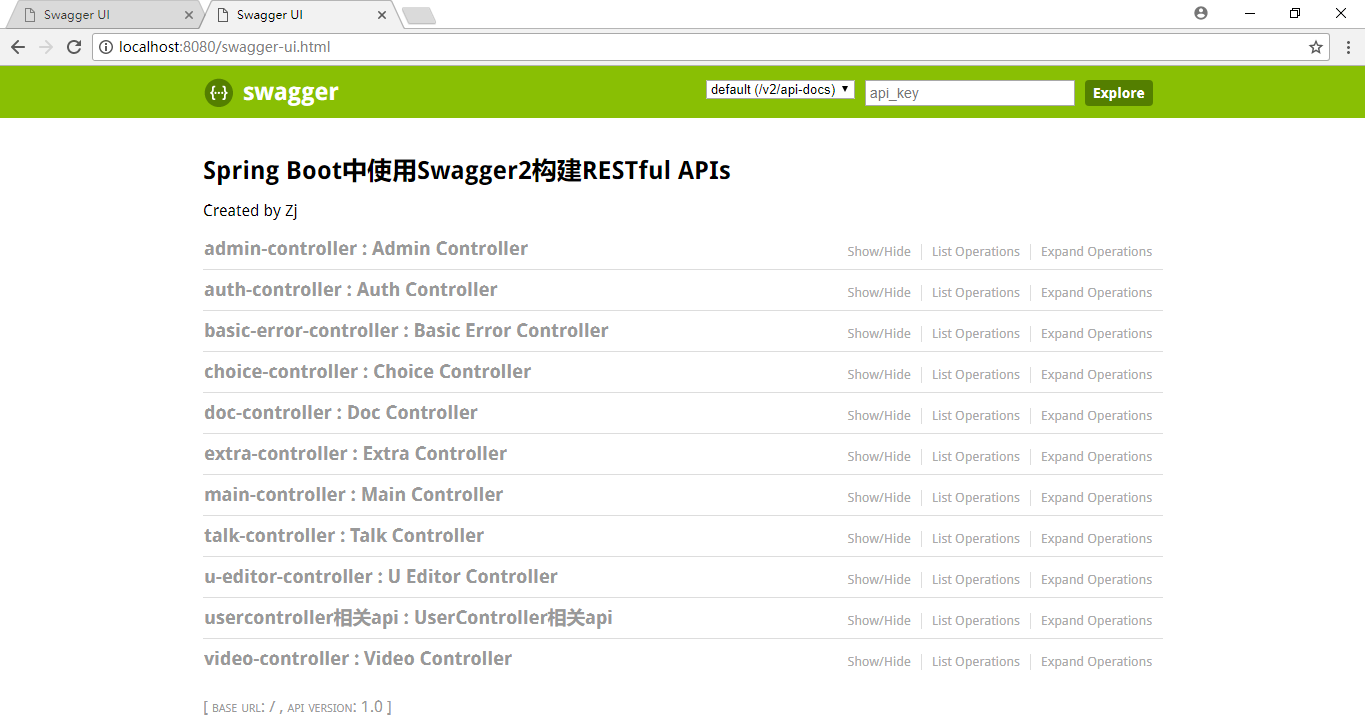
发现

打开f12，发现security拦截了静态资源

在spring security的配置文件中添加

|  |
| --- |
| .antMatchers( "/swagger-ui.html","/webjars/\*\*","/v2/\*\*","/swagger-resources/\*\*" ).permitAll() |

出现如下图所示：



表明成功

5、

#### API文档访问与调试

在上图请求的页面中，我们看到user的Value是个输入框？是的，Swagger除了查看接口功能外，还提供了调试测试功能，我们可以点击上图中右侧的Model Schema（黄色区域：它指明了User的数据结构），此时Value中就有了user对象的模板，我们只需要稍适修改，点击下方“Try it out！”按钮，即可完成了一次请求调用！

此时，你也可以通过几个GET请求来验证之前的POST请求是否正确。

相比为这些接口编写文档的工作，我们增加的配置内容是非常少而且精简的，对于原有代码的侵入也在忍受范围之内。因此，在构建RESTful API的同时，加入swagger来对API文档进行管理，是个不错的选择。

或者用postman测试接口，都可以

6

2018-6-28 补充

有没有发现这默认的页面贼丑

可以自定义swaggerUI界面的哟

Swagger 自定义UI界面 - CSDN博客

<https://blog.csdn.net/tuposky/article/details/77965139>

但后来发现，这个api文档太丑了！！！，改用aglio

参考文档

Spring Boot中使用Swagger2构建强大的RESTful API文档 | 程序猿DD

<http://blog.didispace.com/springbootswagger2/>

springboot结合swagger自动生成接口文档 - 高山\_流水 - 博客园

<https://www.cnblogs.com/xu-lei/p/7423883.html>

Swagger使用总结 - hmwz - 博客园

<https://www.cnblogs.com/h9527/p/5506956.html>

springboot+spring security 集成 swagger2 - OSC\_VBxJVz的个人空间 - 开源中国

<https://my.oschina.net/u/3362894/blog/1789937>

swagger常用注解说明 - 简书

<https://www.jianshu.com/p/12f4394462d5>