Swagger2+Oauth2 导致无法访问页面无法请求

作者：sirwsl
原文链接：https://ld246.com/article/1635949183956
来源网站：链滴

故事

最近一直在加班，之前也没用国Oauth2进行权限控制，然后有点懵逼，导致配置了Swagger无法进访问。无法访问包括：

1、页面无法访问

2、页面可以访问，访问接口报401没得权限

解决方法

解决页面访问问题

由于配置了Oauth2，因此需要对ResourceServerConfigurerAdapter中的configure方法进行重写下

具体内容如下：

```
import com.yth.oauth.component.YthUserAuthenticationConverter;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.context.annotation.Configuration;
import org.springframework.security.config.annotation.web.builders.HttpSecurity;
import org.springframework.security.oauth2.config.annotation.web.configuration.EnableReso
rceServer;
import org.springframework.security.oauth2.config.annotation.web.configuration.ResourceSe
verConfigurerAdapter;
import org.springframework.security.oauth2.config.annotation.web.configurers.ResourceServe
SecurityConfigurer;
import org.springframework.security.oauth2.provider.token.DefaultAccessTokenConverter;
import org.springframework.security.oauth2.provider.token.RemoteTokenServices;
import org.springframework.security.oauth2.provider.token.UserAuthenticationConverter;
import org.springframework.web.client.RestTemplate;

import javax.annotation.Resource;

/**
 * @author sirwsl
 */
@Configuration
@EnableResourceServer
public class ResourceServerConfiguration extends ResourceServerConfigurerAdapter {


    @Resource
    private RestTemplate restTemplate;

    @Resource
    private RemoteTokenServices remoteTokenServices;


    @Override
    public void configure(HttpSecurity http) throws Exception {
```

```
        //排除认证
        http.authorizeRequests().antMatchers(
                "/webjars/**",
                "/resources/**",
                "/swagger-ui.html",
                "/swagger-resources/**",
                "/v2/api-docs",
                "/login",
                "/doc.html",
                "/static",
                "/",
                "/oa/**").permitAll()
                // 配置order访问控制，必须认证后才可以访问
                .antMatchers("/**/**").authenticated();
    }

    @Override
    public void configure(ResourceServerSecurityConfigurer resources) {
        DefaultAccessTokenConverter accessTokenConverter = new DefaultAccessTokenConvert
r();
        UserAuthenticationConverter userTokenConverter = new YthUserAuthenticationConvert
r();
        accessTokenConverter.setUserTokenConverter(userTokenConverter);
        remoteTokenServices.setRestTemplate(restTemplate);
        remoteTokenServices.setAccessTokenConverter(accessTokenConverter);
        resources.tokenServices(remoteTokenServices);
    }
}
```

解决接口401问题

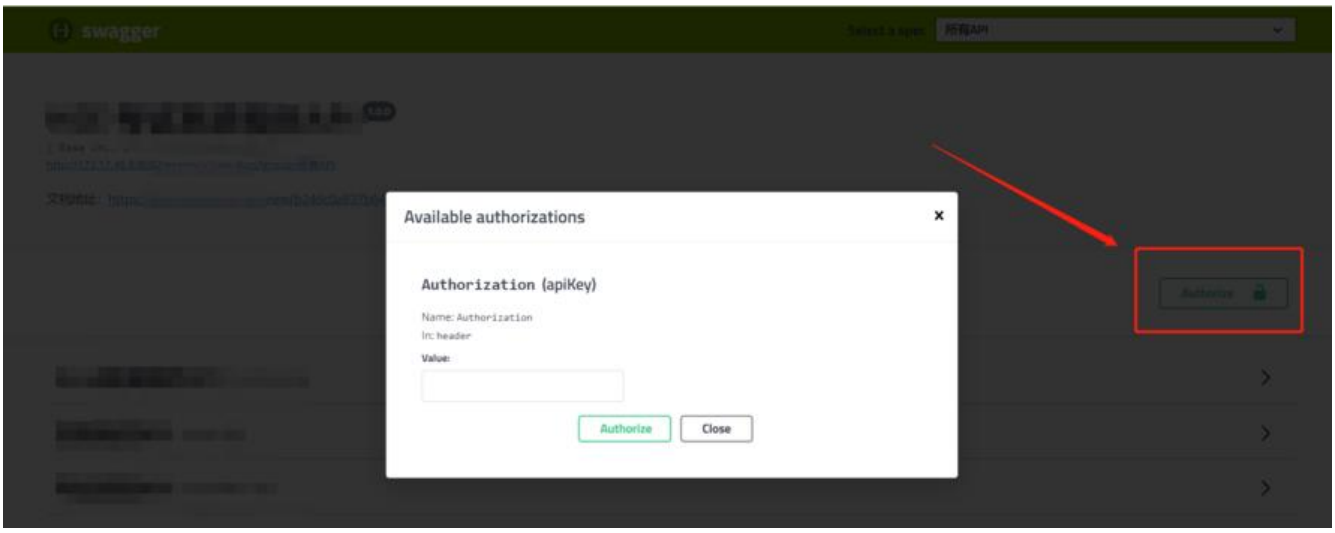
接口401因为没有授权，这个时候就需要考虑如何授权问题，因此我们对原有的swagger增加一些代如下：

```
import com.google.common.base.Predicates;
import io.swagger.annotations.ApiOperation;
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import springfox.documentation.builders.ApiInfoBuilder;
import springfox.documentation.builders.PathSelectors;
import springfox.documentation.builders.RequestHandlerSelectors;
import springfox.documentation.service.*;
import springfox.documentation.spi.DocumentationType;
import springfox.documentation.spi.service.contexts.SecurityContext;
import springfox.documentation.spring.web.plugins.Docket;
import springfox.documentation.swagger2.annotations.EnableSwagger2;

import java.util.ArrayList;
import java.util.List;

/**
 * swagger配置类
 * http://localhost:8080/swagger-ui.html
```

```
 * @author sirwsl
 */
@Configuration
@EnableSwagger2
public class SwaggerConfig {

    private final String AUTH_SERVER = "exams";

    @Bean
    public Docket createApi() {
        return new Docket(DocumentationType.SWAGGER_2)
                .select()
                .apis(RequestHandlerSelectors.withMethodAnnotation(ApiOperation.class))
                .apis(Predicates.or(RequestHandlerSelectors.basePackage("com.yth.exam.api"),
                        RequestHandlerSelectors.basePackage("com.yth.course.api"),
                        RequestHandlerSelectors.basePackage("com.yth.score.api")))
                .paths(PathSelectors.any())
                .build()
                .apiInfo(apiInfo())
                .securitySchemes(securitySchemes())
                .securityContexts(securityContexts())
                .groupName("所有API");
    }

    @Bean
    public Docket createExamApi() {
        return new Docket(DocumentationType.SWAGGER_2)
                .apiInfo(examApi())
                .select()
                .paths(PathSelectors.any())
                .paths(PathSelectors.ant("/xxx/**"))
                .build()
                .groupName("xxAPI")
                .pathMapping("/");
    }


    @Bean
    public Docket createDailyApi() {
        return new Docket(DocumentationType.SWAGGER_2)
                .apiInfo(dailyApi())
                .select()
                .paths(PathSelectors.ant("/xxx/**"))
                .paths(PathSelectors.any())
                .build()
                .groupName("日常xxxAPI")
                .pathMapping("/");
    }



    public ApiInfo apiInfo() {
        return  new ApiInfoBuilder().title("API")
                .description("文档地址支持md语法")
```

```
                .version("1.0.0").build();
    }

    private ApiInfo examApi() {
        return new ApiInfoBuilder().title("xxxAPI").description("").version("1.0.0").build();
    }

    private ApiInfo dailyApi() {
        return new ApiInfoBuilder().title("xxxAPI").description("").version("1.0.0").build();
    }

    /**
    * 从这里开始是增加授权部分，具体的授权可以自己更改，这里采用token的方式
    **/

    private List<ApiKey> securitySchemes() {
        List<ApiKey> apiKeys = new ArrayList<>();
        apiKeys.add(new ApiKey("Authorization", "Authorization", "header"));
        return apiKeys;
    }

    private List<SecurityContext> securityContexts() {
        List<SecurityContext> securityContexts = new ArrayList<>();
        securityContexts.add(SecurityContext.builder()
                .securityReferences(defaultAuth())
                .forPaths(PathSelectors.regex("^(?!auth).*$")).build());
        return securityContexts;
    }

    private List<SecurityReference> defaultAuth() {
        AuthorizationScope authorizationScope = new AuthorizationScope("global", "accessEver
thing");
        AuthorizationScope[] authorizationScopes = new AuthorizationScope[1];
        authorizationScopes[0] = authorizationScope;
        List<SecurityReference> securityReferences = new ArrayList<>();
        securityReferences.add(new SecurityReference("Authorization", authorizationScopes));
        return securityReferences;
    }


}
```

实现效果如下



进入页面后会多了一个权限按钮，点击后可以添加token，然后就可以访问

原文链接：Swagger2+Oauth2 导致无法访问页面无法请求