1.Swagger概述

- `Swagger`号称世界上最流行的Api框架。
- Restful Api 文档在线自动生成工具 => Api 文档与API定义同步更新。
- 直接运行，可以在线测试API接口。
- 支持多种语言：Java、PHP。。。

在项目中使用`Swagger`需要使用`springbox`，包括两个组件：

- swagger2
- ui

2.SpringBoot整合

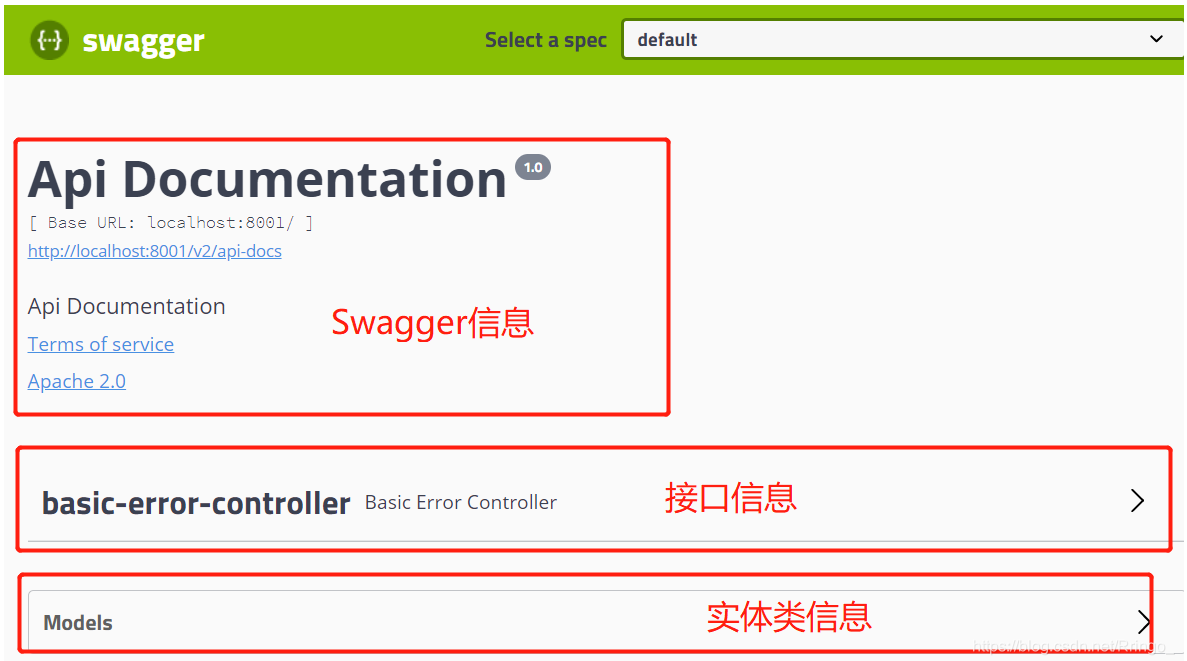
> 依赖

```
<properties>
    <java.version>1.8</java.version>
    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
    <project.reporting.outputEncoding>UTF-8</project.reporting.outputEncoding>

 <springfox.swagger2.version>2.9.2</springfox.swagger2.version>
    <springfox.swagger.ui>2.9.2</springfox.swagger.ui>
</properties>

<!--swagger-->
<dependency>
    <groupId>io.springfox</groupId>
    <artifactId>springfox-swagger2</artifactId>
    <version>${springfox.swagger2.version}</version>
</dependency>
<dependency>
    <groupId>io.springfox</groupId>
    <artifactId>springfox-swagger-ui</artifactId>
    <version>${springfox.swagger.ui}</version>
</dependency>
```

> @EnableSwagger2启动Swagger2

```
@EnableSwagger2
@SpringBootApplication
public class SwaggerApplication {
    public static void main(String[] args) {
        SpringApplication.run(SwaggerApplication.class, args);
    }
}
```

启动SpringBoot应用，访问 http://localhost:8001/swagger-ui.html 就可以使用Swagger了。



3.配置Swagger

```
package com.ymy.spring.boot.swagger.conf;

import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import
springfox.documentation.builders.RequestHandlerSelectors;
import springfox.documentation.service.ApiInfo;
import springfox.documentation.service.Contact;
import springfox.documentation.service.VendorExtension;
import springfox.documentation.spi.DocumentationType;
import springfox.documentation.spring.web.plugins.Docket;

import java.util.ArrayList;

```

```java
@Configuration
public class SwaggerConf {

    // 配置Swagger的Docket实例
    @Bean
    public Docket docket() {
        Docket docket = new Docket(DocumentationType.SWAGGER_2);

        // 配置基本信息
        docket.apiInfo(apiInfo())
                // 是否启用swagger 默认是true
                .enable(true)
                .select()
                // 指定要扫描的包

.apis(RequestHandlerSelectors.basePackage("com.ymy.spring.boot.swagger"))
                .build()
                // 设置分组 可以设置多个Docket
                .groupName("Ringo");

        return docket;
    }

    // 配置Swagger信息apiInfo
    private ApiInfo apiInfo() {
        // 作者信息
        Contact contact = new Contact("Ringo", "https://github.com/RingoTangs/LearningNote", "123@qq.com");

        ApiInfo apiInfo = new ApiInfo("Ringo的Swagger文档", "每天都要努力", "1.0",
                "https://github.com/RingoTangs/LearningNote", contact, "Apache 2.0",
                "http://www.apache.org/licenses/LICENSE-2.0", new ArrayList<VendorExtension>());
        return apiInfo;
    }

    // apiInfo()也可以这样写
    private ApiInfo apiInfo() {
        Contact contact = new Contact("Ringo", "", "1466637477@qq.com");
        ApiInfo apiInfo = new ApiInfoBuilder()
```

```
                .title("网课-课程中心API文档")
                .description("本文档描述了课程中心微服务接口定义")
                .version("1.0")
                .contact(contact)
                .build();
        return apiInfo;

}
```

4.Swagger注解

> 实体类

```
@Data
@AllArgsConstructor
@NoArgsConstructor
@ApiModel(value = "User", description = "用户实体类") //@ApiModel标注在实体类上,对实体类信息的描述
public class User {

    // @ApiModelProperty 标注在属性上，对属性的说明
    @ApiModelProperty(value = "用户名")
    private String username;

    @ApiModelProperty(value = "密码")
    private String password;
}
```

> 方法参数

```
//@Api(tags = {"UserController对User的数据进行操作"})
@RestController
public class UserController {

    // @ApiOperation 标注在方法上
    @ApiOperation(value = "返回User对象")
    @GetMapping("/user")
    public User getUser() {
        return new User("Ringo", "123");
    }

    // @ApiParam 标注在参数上
    @ApiOperation(value = "返回name")
    @GetMapping("/name")
```

```
15      public String getName(@ApiParam(name = "name", value =
"用户名") @RequestParam("name") String name) {
16          return "Hello " + name;
17      }
18  }
```