

1. Config swagger2 / swagger-ui pour spring-mvc

à ajouter dans pom.xml

```
<dependency>
  <groupId>io.springfox</groupId>
  <artifactId>springfox-swagger2</artifactId>
  <version>2.9.2</version>
</dependency>

<dependency>
  <groupId>io.springfox</groupId>
  <artifactId>springfox-swagger-ui</artifactId>
  <version>2.9.2</version>
</dependency>
```

en plus de

```
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-web</artifactId>
</dependency>
```

MySwaggerConfig (à placer à côté de *SpringWsApplication* comportant *main()*)

```
package fr.xyz.springws;
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import org.springframework.context.annotation.Profile;
import springfox.documentation.builders.ApiInfoBuilder;
import springfox.documentation.builders.PathSelectors;
import springfox.documentation.builders.RequestHandlerSelectors;
import springfox.documentation.service.ApiInfo;
import springfox.documentation.spi.DocumentationType;
import springfox.documentation.spring.web.plugins.Docket;
import springfox.documentation.swagger2.annotations.EnableSwagger2;

@Configuration
@Profile("swagger")
@EnableSwagger2
public class SwaggerConfig {
    // avec <a href="/swagger-ui.html">description api rest via swagger 2</a>
    // dans index.html
    //et app.setAdditionalProfiles("initData","swagger");
    //dans le main()
    //et dépendances springfox-swagger2 , springfox-swagger-ui dans pom.xml

    @Bean
    public Docket api() {
        return new Docket(DocumentationType.SWAGGER_2)
            .select()
            .apis(RequestHandlerSelectors.any())
            .apis(RequestHandlerSelectors.basePackage("fr.xyz.springws.rest"))
            .paths(PathSelectors.any())
            .build()
            .apiInfo(apiInfo());
    }
    private ApiInfo apiInfo() {
        return new ApiInfoBuilder()
            .title("My Spring-Mvc REST APIs").version("1.0.0").build();
    }
}
```

et en activant le profile spring "swagger" au démarrage de l'application ;


```
package fr.xyz.springws;
import org.springframework.boot.SpringApplication;
```


1. Config swagger2 / swagger-ui pour spring-mvc

```
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.context.ConfigurableApplicationContext;
@SpringBootApplication
public class SpringWsApplication {
    public static void main(String[] args) {
        SpringApplication app = new SpringApplication(SpringWsApplication.class);
        app.setAdditionalProfiles("initData", "swagger");
        ConfigurableApplicationContext context = app.run(args);
        System.out.println("http://localhost:8080/spring-ws");
    }
}
```

dans **index.html** (ou ailleurs) :

```
<a href="/v2/api-docs">description swagger2(json) de l'api REST</a> <br/>
...
<a href="/swagger-ui.html">documentation Api REST g n r e dynamiquement par swagger2 et swagger-ui</a>
```

 **swagger**

default (/v2/api-docs) 

Explore

My REST API (serverSpring MVC)

Api pour Devise

Created by DJA1
See more at www.afcepf.fr
[Contact the developer](#)
[License of API](#)

rest-devise-service : Rest Devise Service Show/Hide List Operations Expand Operations

GET	/rest/devise	devisesByCriteria
POST	/rest/devise	saveOrUpdateDevise
DELETE	/rest/devise/{codeDev}	deleteDeviseByCode
GET	/rest/devise/{codeDev}	deviseByCode

ws-auth : Ws Auth Show/Hide List Operations Expand Operations

ws-confidentiel : Ws Confidentiel Show/Hide List Operations Expand Operations

[BASE URL: /serverSpringMvc/ws , API VERSION: API TOS]

NB: Selon le contexte applicatif, il faudra peut être paramétrer la sécurité de façon à pouvoir accéder à la documentation "swagger" générée :

@Configuration

@EnableWebSecurity

```
public class WebSecurityConfig extends WebSecurityConfigurerAdapter
    //....

private static final String[] SWAGGER_AUTH_WHITELIST = {
    "/swagger-resources/**", "/swagger-ui.html", "/v2/api-docs", "/webjars/**"
};

protected void configure(HttpSecurity http) throws Exception {
    // configuration partielle à compléter:
    http.authorizeRequests()
        .antMatchers("/rest/devise-api/public/**").permitAll()
        .antMatchers(SWAGGER_AUTH_WHITELIST).permitAll()
        .anyRequest().authenticated();
}
}
```

Configuration de l'api via "annotations swagger" :

```
package org.mycontrib.backend.dto;

import io.swagger.annotations.ApiModelProperty;

public class ResConv {

    @ApiModelProperty(value = "amount to convert", example = "100")
    //or @ApiModelProperty(value = "${SomeModel.someProperty}", ...)
    //with SomeModel.someProperty=amount to convert in application.properties
    private Double amount;

    @ApiModelProperty(value = "source currency code", example = "EUR")
    private String source;

    ...
}
```

et dans une classe de `@RestController` :

```
...
import io.swagger.annotations.ApiOperation;
import io.swagger.annotations.ApiParam;
...
@RestController
@RequestMapping(value="/rest/devise-api/public" , headers="Accept=application/json")
public class PublicDeviseRestCtrl {
...
@RequestMapping(value="/convert" , method=RequestMethod.GET)
@ApiOperation(value = "convert amount from source to target currency",
    notes = "exemple: convert?source=EUR&target=USD&amount=100")
    public ResConv convertir(
        @RequestParam("amount")
        @ApiParam(value = "amount to convert", example = "100")
        Double montant,
        @RequestParam("source")
        @ApiParam(value = "source currency code", example = "EUR")
        String source,
        @RequestParam("target")
        @ApiParam(value = "target currency code", example = "USD")
        String cible) {
        Double res = convertisseur.convertir(montant, source, cible);
        return new ResConv(montant, source, cible,res);
    }
....
```