



Lab 2

REST web service documented with Swagger


This lab will show you how you can document your Spring Boot REST web service using Swagger.

1. Starting point

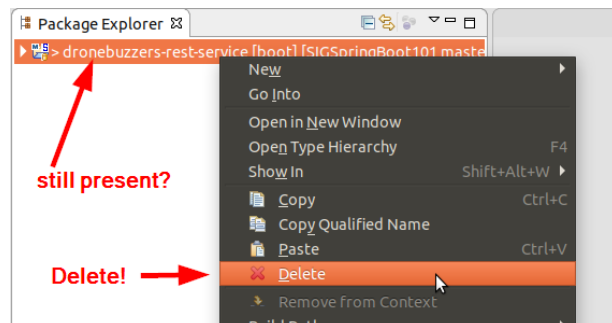
The starting point for this lab is to have the provided VirtualBox machine up-and-running:

- You are logged in under user/password: developer/welcome01
- You have updated the labs running the `git pull` command in the lab workspace directory `/home/developer/projects/SIGSpringBoot101`

The basis for this lab is the service that was created in Lab 1. The code of this service is located in directory `/home/developer/projects/SIGSpringBoot101/lab 2/dronebuzzers`.

Start by opening Eclipse  with `/home/developer/projects/SIGSpringBoot101` as the Workspace directory.

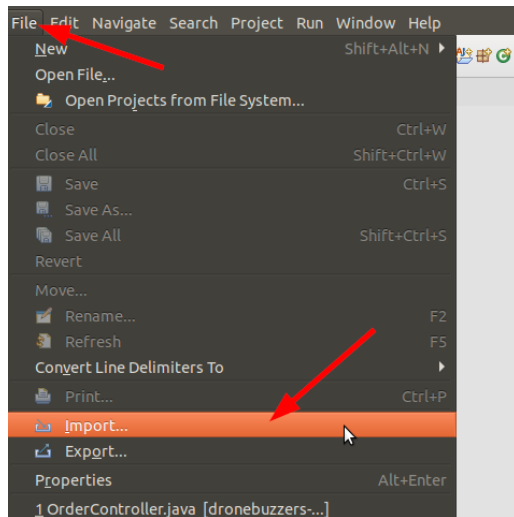
If the project `dronebuzzers-rest-service` is still present from the previous lab, you have to delete that first. Right-click the project in the Package Explorer and click Delete:



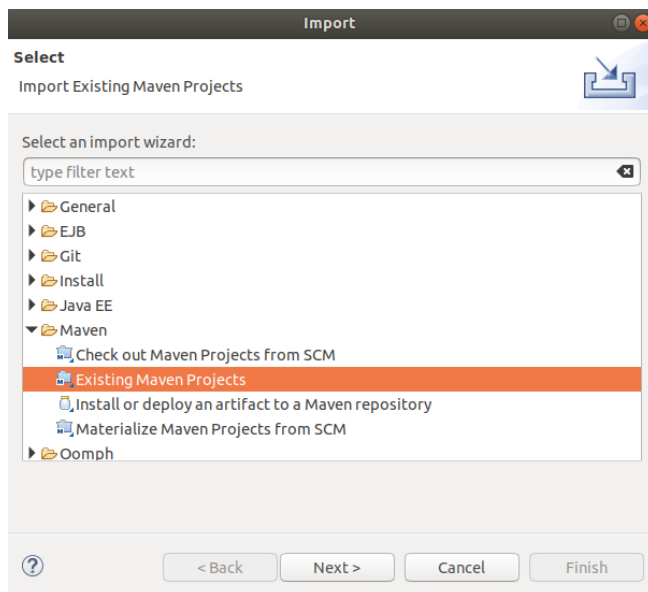


Spring Boot 101 - intro, demo & hands-on with Java, REST APIs, Containers & Cloud

Now, we will import the project that will be used in this Lab. In the menu bar of STS Eclipse, click on File and then Import:



In the resulting Import pop-up, select 'Existing Maven Projects':

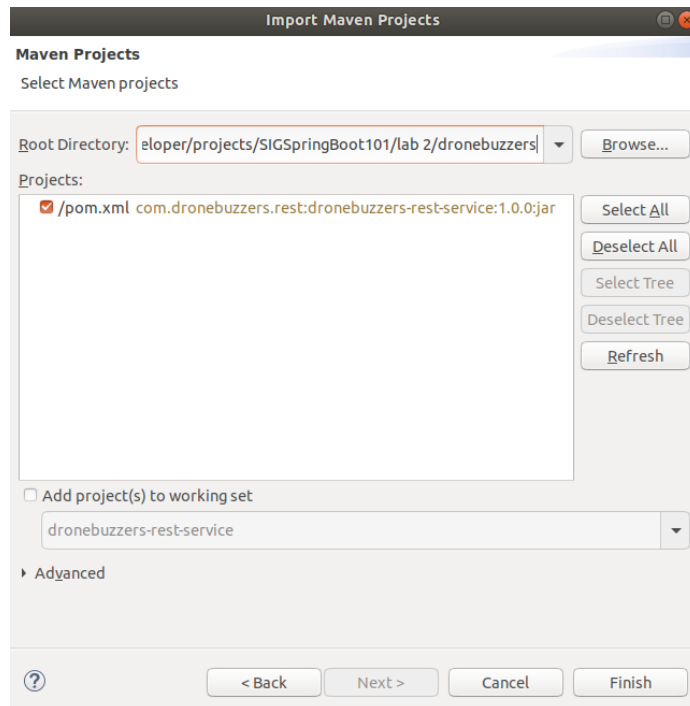


Click Next and then:

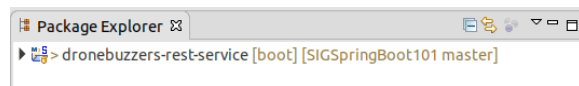
- set the Root Directory: /home/developer/projects/SIGSpringBoot101/lab 2/dronebuzzers
- select the pom file



Spring Boot 101 - intro, demo & hands-on with Java, REST APIs, Containers & Cloud



Click Finish and the dronebuzzers-rest-service project should become visible in the Package Explorer:



2. Code changes for documentation

Adding documentation to the service is surprisingly simple:

- Step 1: add the Maven dependencies
- Step 2: add the Docket bean

Step 1: add the Maven dependencies

In the pom.xml file, add the dependencies that can be found in file:

```
/home/developer/projects/SIGSpringBoot101/lab 2/input/mvn-dependencies.xml
```



Spring Boot 101 - intro, demo & hands on with Java, REST APIs, Containers & Cloud

The pom.xml will then look like:

```
dronebuzzers-rest-service/pom.xml
<project.reporting.outputEncoding>UTF-8</project.reporting.outputEncoding>
<java.version>1.8</java.version>
</properties>

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

  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-test</artifactId>
    <scope>test</scope>
  </dependency>

  <dependency>
    <groupId>io.springfox</groupId>
    <artifactId>springfox-swagger2</artifactId>
    <version>2.7.0</version>
  </dependency>

  <dependency>
    <groupId>io.springfox</groupId>
    <artifactId>springfox-swagger-ui</artifactId>
    <version>2.7.0</version>
  </dependency>
</dependencies>

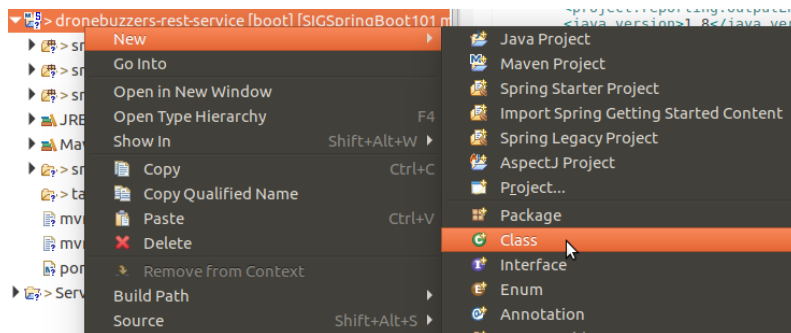
<build>
  <plugins>
    <plugin>
```

The top dependency adds the Springfox support for the Swagger specification. If only this dependency is added, a 'machine-readable' specification is available from the running service. The bottom dependency adds a the Springfox Swagger UI implementation, for a human readable interface specification.

Step 2: add the Docket bean

To make the API specification/documentation available, a so-called Docket bean has to be added.

Right-click the project and add a java class:



Complete the form like shown below:

- Package: com.dronebuzzers.rest.config
- Name: SwaggerConfig



Spring Boot 101 - intro, demo & handson with Java, REST APIs, Containers & Cloud

New Java Class

Java Class
Create a new Java class.

Source folder: Browse...

Package: Browse...

☐ Enclosing type: Browse...

Name:

Modifiers: ☒ public ☐ package ☐ private ☐ protected
☐ abstract ☐ final ☐ static

Superclass: Browse...

Interfaces: Add... Remove

Which method stubs would you like to create?
☐ public static void main(String[] args)
☐ Constructors from superclass
☒ Inherited abstract methods

Do you want to add comments? (Configure templates and default value [here](#))
☐ Generate comments

Cancel Finish

Next, replace the code of SwaggerConfig.java by the content of file:

```
/home/developer/projects/SIGSpringBoot101/lab 2/input/SwaggerConfig.java
```

The resulting code will look like:

```
package com.dronebuzzers.rest.config;

import org.springframework.context.annotation.Bean;

@Configuration
@EnableSwagger2
public class SwaggerConfig {

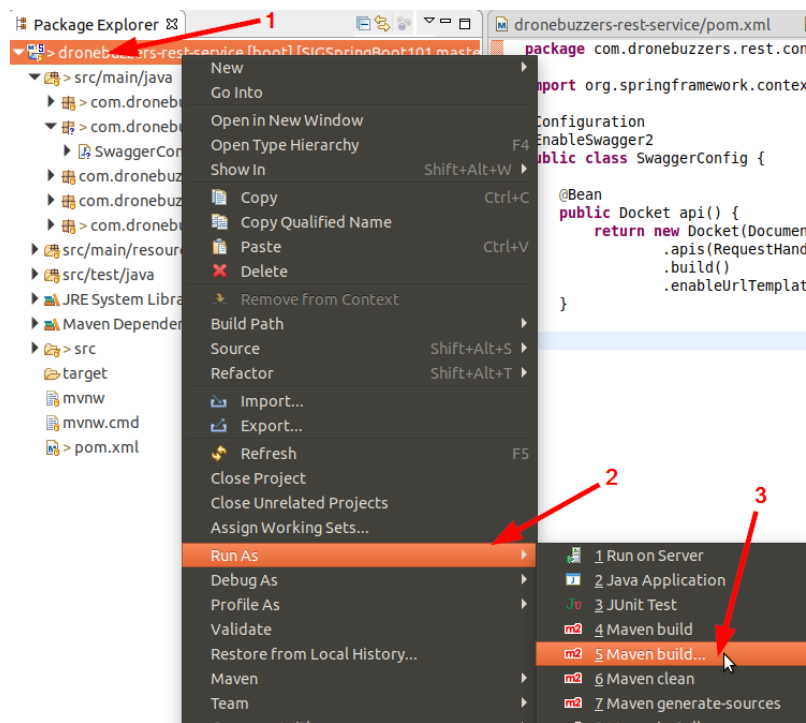
    @Bean
    public Docket api() {
        return new Docket(DocumentationType.SWAGGER_2).select()
            .apis(RequestHandlerSelectors.basePackage("com.dronebuzzers.rest.controller")).paths(PathSelectors.any())
            .build()
            .enableUrlTemplating(true);
    }
}
```

The Docket configuration identifies what part of the api is added to the Swagger documentation. In our case, all operations that are found under the package `com.dronebuzzers.rest.controller` will become visible in the Swagger documentation.



3. Run

Now that the code is ready, it is time to build the code: right-click the project, click 'Run As' and select the option 'Maven build...':

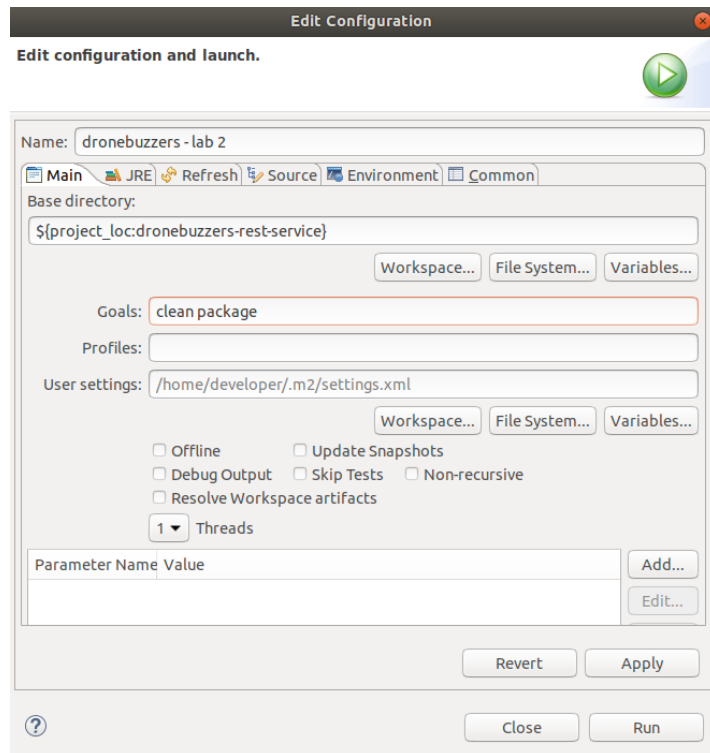


The pop-up window is shown. Complete the window like shown below:

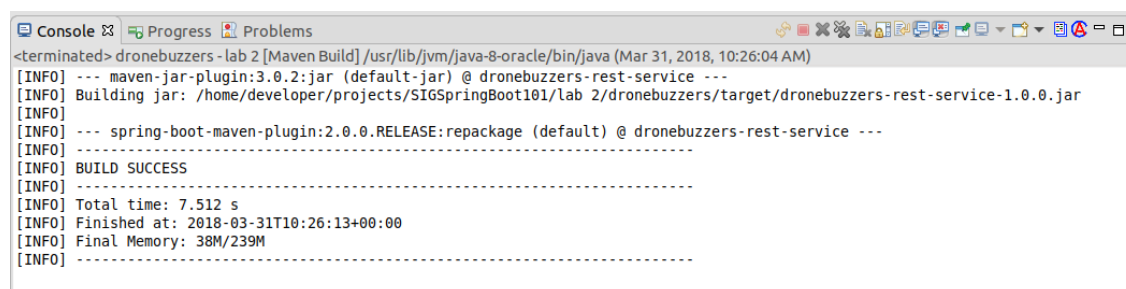
- Name: dronebuzzers – lab 2
- Goals: clean package



Spring Boot 101 - intro, demo & hands-on with Java, REST APIs, Containers & Cloud



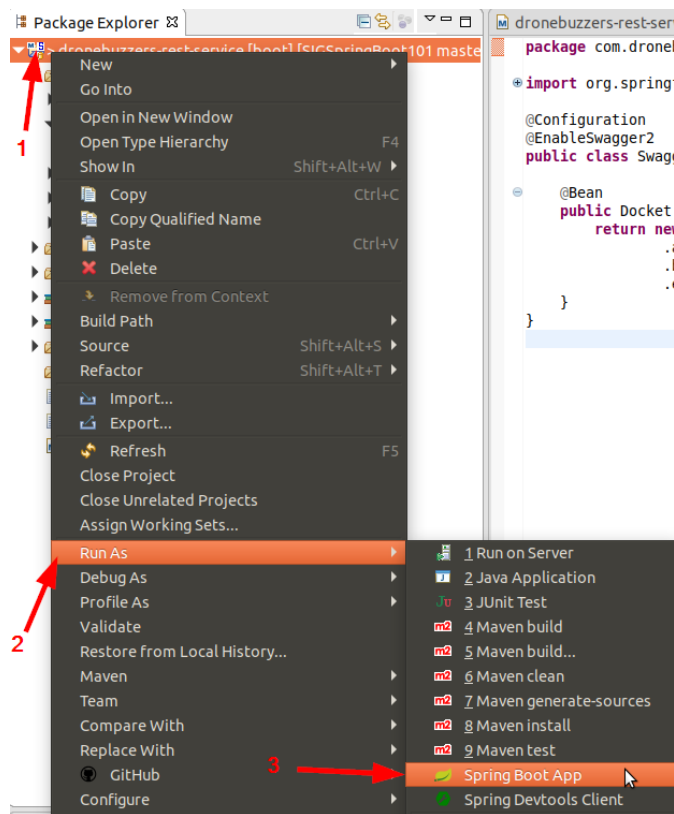
Complete like shown above and click Run. Check in the console that the code is built successfully:



The code is built: time to run it: right-click the project, select 'Run As' and then 'Spring Boot App':



Spring Boot 101 - intro, demo & hands on with Java, REST APIs, Containers & Cloud



Verify the Console window for errors...(there shouldn't be any)

4. Verify documentation

Now, there are two types of documentation that are available.

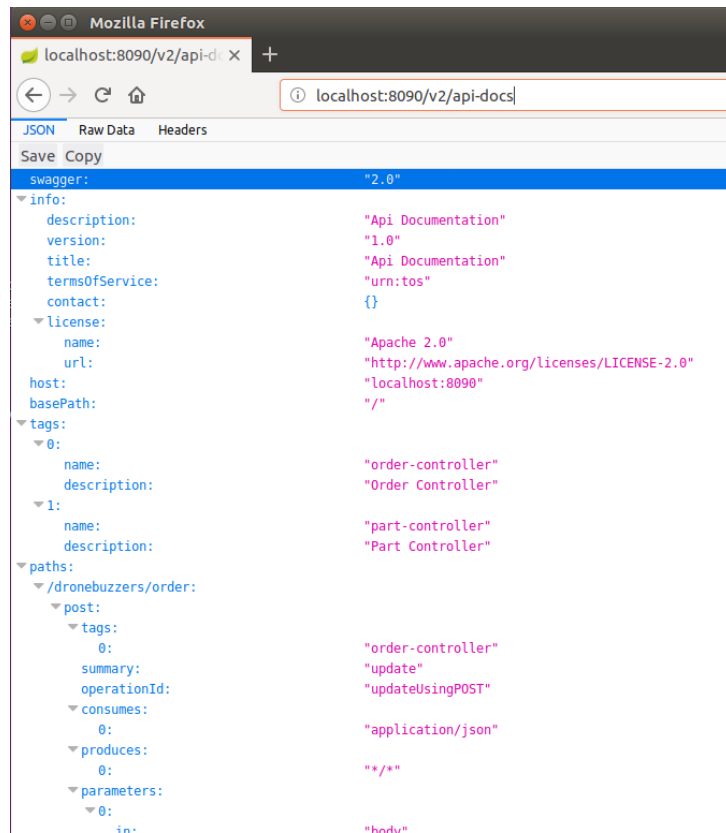
- Machine readable documentation: <http://localhost:8090/v2/api-docs>
- Human readable documentation: <http://localhost:8090/swagger-ui.html>

Machine readable documentation

The Url <http://localhost:8090/v2/api-docs> will return the Swagger API 2.0 specification document:



Spring Boot 101 - intro, demo & handson with Java, REST APIs, Containers & Cloud



Human readable documentation

The Url <http://localhost:8090/swagger-ui.html> will bring you to a more human readable document format.



Spring Boot 101 - intro, demo & handson with Java, REST APIs, Containers & Cloud

Spend some time on inspecting both urls.

Imagine that you are a developer who wants to make use of the Parts API. You do not get the code that created in Lab 1. However, you *will* get access to the API documentation. Is that enough for you to develop the software that consumes the API operations?

And imagine that you are the API developer – who has to make very little effort in order to provide API consumers with the documentation they require.

Adding Swagger documentation in this way is a low effort way to help both the producer and consumer with their work. But, the automatically generated documentation is still pretty limited. However, the API developer can add additional documentation like examples to the code ... with minimal effort. See the pointer below.

5. Additional pointers

- How to configure the Docket:
<https://springfox.github.io/springfox/docs/current/#springfox-configuration-and-demo-applications>
- How to insert additional documentation in the code:
<https://springfox.github.io/springfox/docs/current/#support-for-documentation-from-property-file-lookup>