

Rest Web Service with Spring boot

The goal of this lab work is to develop a small application for cars renting.

The functionalities to be implemented are:

- Get a list of unrented cars
- Rent a car
- Get back a car

Required software

Java (JDK version)

Eclipse

Gradle project creation

Use the Spring Initializer (<https://start.spring.io/>) to create a project:

- Don't forget to choose Gradle Project
- Give it a name (Artifact)
- Add Web as dependencies (library)
- Add DevTools as dependencies

The screenshot shows the Spring Initializer web form. At the top, 'Maven Project' and 'Gradle Project' are tabs, with 'Gradle Project' selected. Below this are tabs for 'Java', 'Kotlin', and 'Groovy', with 'Java' selected. A row of version numbers is shown: '2.2.0 M1', '2.2.0 (SNAPSHOT)', '2.1.4 (SNAPSHOT)', '2.1.3' (which is highlighted with a green underline), and '1.5.19'. Below the versions, the 'Group' field contains 'com.example' and the 'Artifact' field contains 'CarRental'. A 'More options' button is visible. At the bottom, there are two sections: 'Search dependencies to add' and 'Dependencies selected'. The search field contains 'Web, Security, JPA, Actuator, Devtools...'. The 'Dependencies selected' section lists two items: 'Web [Web]' with the description 'Servlet web application with Spring MVC and Tomcat', and 'DevTools [Core]' with the description 'Spring Boot Development Tools'.

Download and unzip the project (outside the Eclipse workspace)

Open a command line window (project location).

Use the following command to build the project (download libraries, compilation...):

- gradlew build under windows
- ./gradlew build under Linux

Add the plugin Eclipse (or idea if you want to use IntelliJ instead of Eclipse) to the configuration file build.gradle:

```
apply plugin: 'java'
apply plugin: 'eclipse'
apply plugin: 'idea'
```

Use the following command to convert the project in an Eclipse project:

- gradlew eclipse under windows or gradlew idea for IntelliJ
- ./gradlew eclipse under Linux or ./gradlew idea for IntelliJ

Import the project under Eclipse: File->Import->General-> Existing project into workspace ...
select the project directory

Launch the main program: /src/main/java/package.../*Application.java

Spring boot coding

Write a class annotated with Controller implementing the car rental service (see <https://github.com/charroux/CarService> for an example).

Then code the following features:

- Get the features of a car:
 - URI: .../cars/plateNumber
 - http GET
 - Json response: { " plateNumber" : "11AA22" , " numberOfSeats" : 5, "price" : 100 }
- Rent a car:
 - URI: .../plateNumber?rent=true
 - http PUT
 - Send Json inside the http body: { "begin" : "11/11/2017" , " end" : "1/1/2018" }
- Get back the car:
 - URI: .../plateNumber?rent=false
 - http PUT

Advice: you can use the following templates.

```
@RequestMapping(value = "/cars", method = RequestMethod.GET)
@ResponseStatus(HttpStatus.OK)
@ResponseBody
public List<Car> listOfCars(){
}
```

```

@RequestMapping(value = "/cars/{plateNumber}", method = RequestMethod.GET)
@ResponseStatus(HttpStatus.OK)
@ResponseBody
public Car aCar(@PathVariable("plateNumber") String plateNumber) throws Exception{
}

@RequestMapping(value = "/cars/{plateNumber}", method = RequestMethod.DELETE)
@ResponseStatus(HttpStatus.OK)
public void getBack(@PathVariable("plateNumber") String plateNumber) throws Exception{
}

@RequestMapping(value = "/cars/{plateNumber}", method = RequestMethod.PUT)
@ResponseStatus(HttpStatus.OK)
public void rent(@PathVariable("plateNumber") String plateNumber) throws Exception{
}

@RequestMapping(value = "/voiture/{plateNumber}", method = RequestMethod.PUT)
@ResponseStatus(HttpStatus.OK) public void
rentAndGetBack(@PathVariable("plateNumber") String plateNumber,
@RequestParam(value="rent", required = true)boolean rent) throws Exception{
}

@RequestMapping(value = "/cars/{plateNumber}", method = RequestMethod.PUT)
public void rent(@PathVariable("plateNumber") String plateNumber, @RequestParam(value="rent",
required = true)boolean rent, @RequestBody Dates dates){
}

```

Test your application

Test the web service inside a web browser: <http://localhost:8080/cars>

Use a plugin for web browser like RestClient or Postman to test your application.