Lab 7 SOAP web service ... in Spring Boot

This lab will show you how you can create a SOAP web service using Spring Boot. We will create a SOAP web service for retrieving drone parts information. Where the 'default' with Spring Boot is SOAP version 1.1, this lab will implement a SOAP 1.2 version.

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`

1. Overview

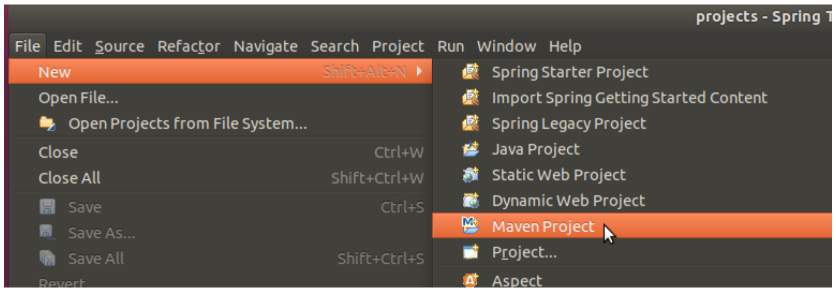
In this lab. We will start with a blanc maven project and add everything that is needed for the SOAP web service. The following steps will be done in this lab:

- Section 2: Project and the pom.xml
- Section 3: XML schema for messages
- Section 4: JAXB: Java classes for parts.xsd
- Section 5: Parts repository
- Section 6: Web service endpoint
- Section 7: Web service configuration
- Section 8: Runnable application
- Section 9: Run and Test the Soap web service

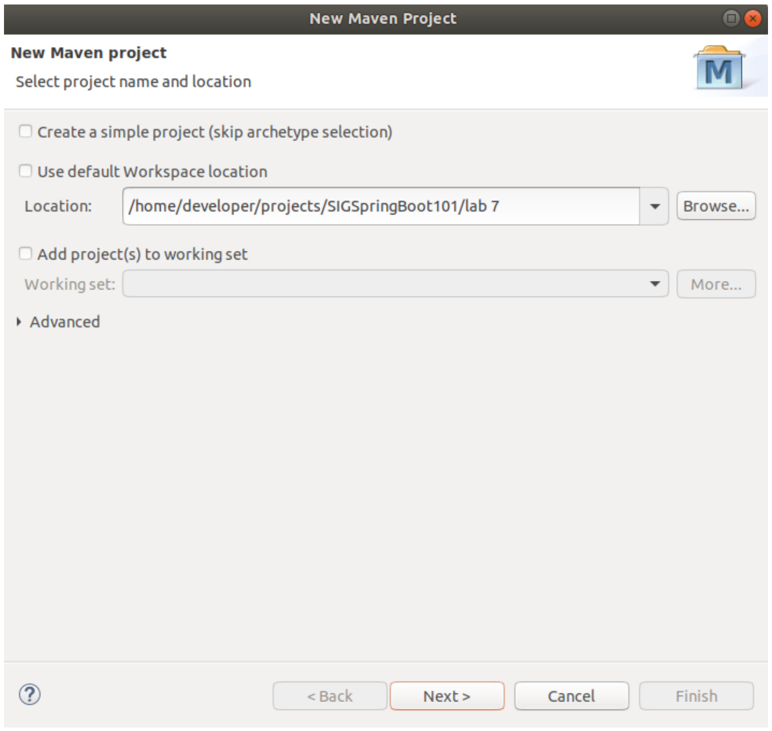
2. Project and the pom.xml

First, we are going to create an Eclipse STS project and update the maven pom file.

Start Eclipse STS and then create a maven project:



Complete the pop-up like shown below:


New Maven Project

New Maven project

Select project name and location

☐ Create a simple project (skip archetype selection)

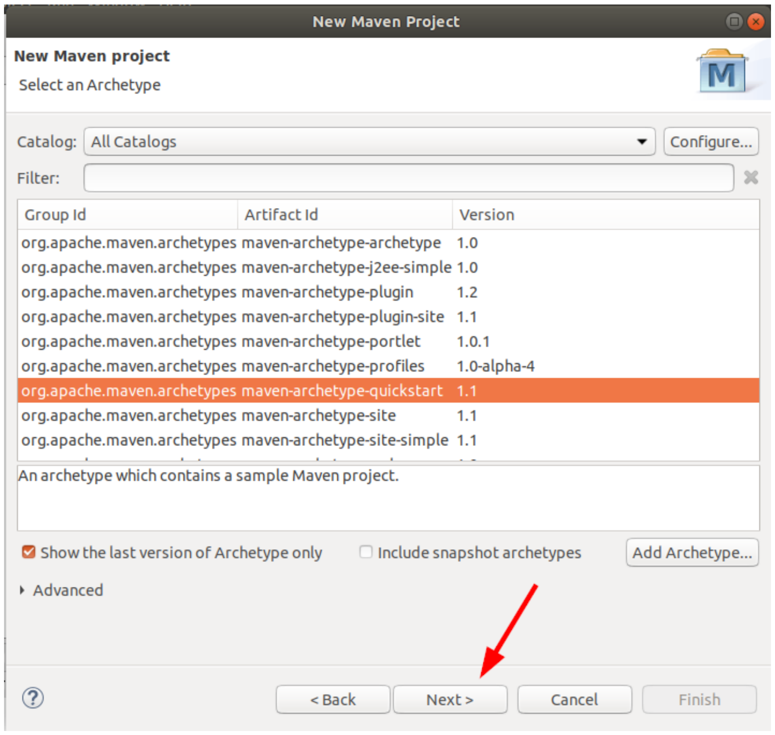
☐ Use default Workspace location

Location: /home/developer/projects/SIGSpringBoot101/lab 7

☐ Add project(s) to working set

Working set:

Click Next and set the archetype like shown below:



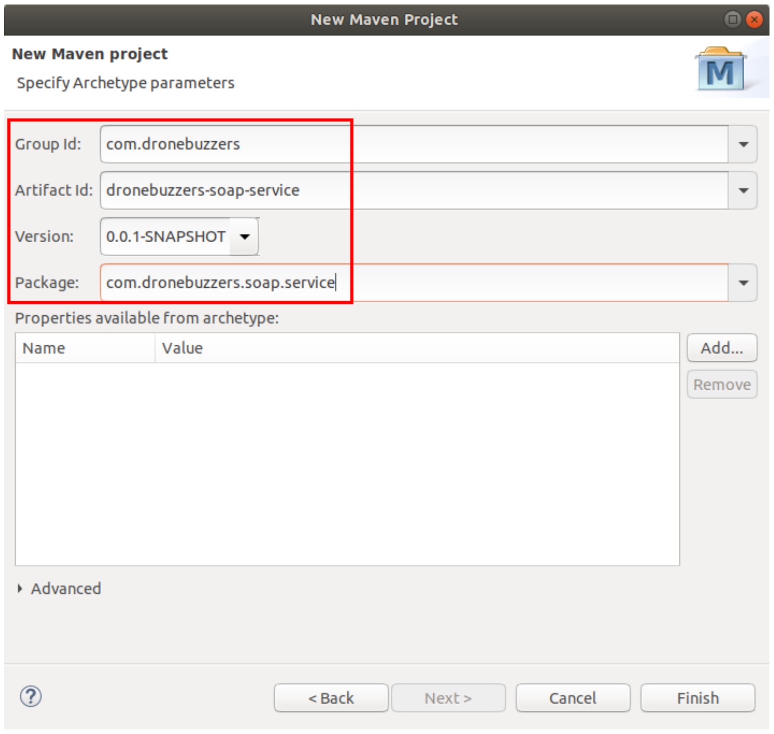
Group Id	Artifact Id	Version
org.apache.maven.archetypes	maven-archetype-archetype	1.0
org.apache.maven.archetypes	maven-archetype-j2ee-simple	1.0
org.apache.maven.archetypes	maven-archetype-plugin	1.2
org.apache.maven.archetypes	maven-archetype-plugin-site	1.1
org.apache.maven.archetypes	maven-archetype-portlet	1.0.1
org.apache.maven.archetypes	maven-archetype-profiles	1.0-alpha-4
org.apache.maven.archetypes	maven-archetype-quickstart	1.1
org.apache.maven.archetypes	maven-archetype-site	1.1
org.apache.maven.archetypes	maven-archetype-site-simple	1.1

An archetype which contains a sample Maven project.

☑ Show the last version of Archetype only
☐ Include snapshot archetypes

Click Next and complete like shown below:

- Group Id: com.dronebuzzers
- Artifact Id: dronebuzzers-soap-service
- Package: com.dronebuzzers.soap.service



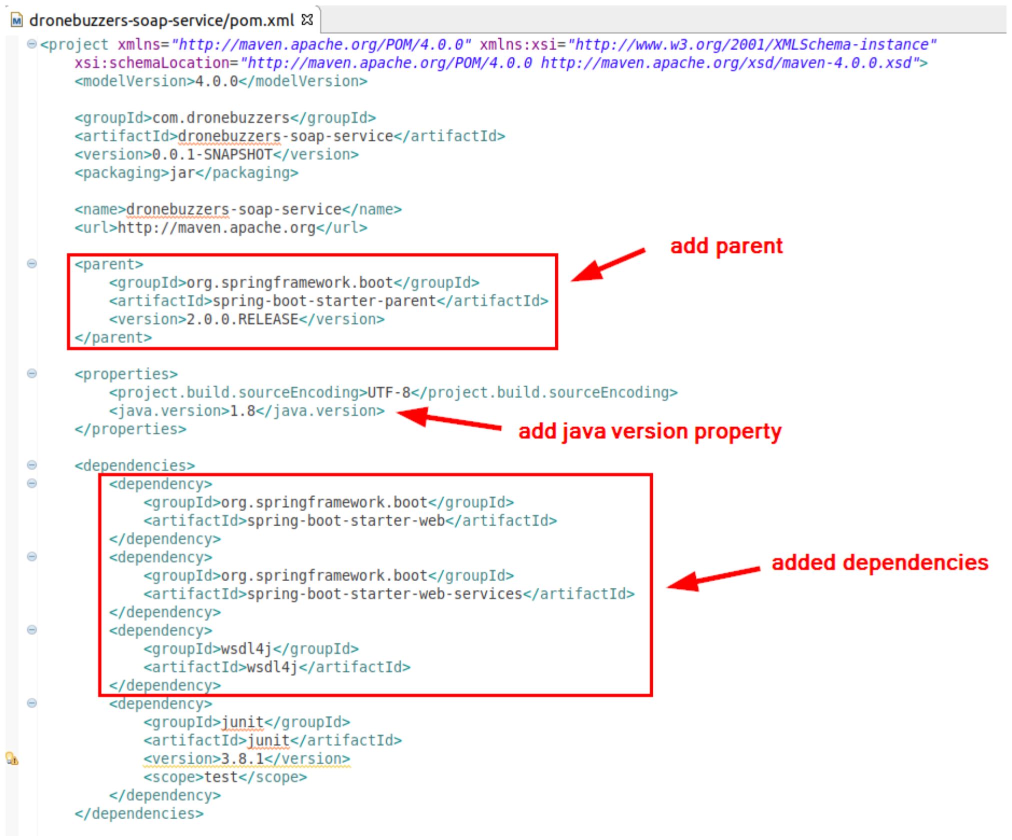
Group Id: com.dronebuzzers
Artifact Id: dronebuzzers-soap-service
Version: 0.0.1-SNAPSHOT
Package: com.dronebuzzers.soap.service

Click Finish

Now, the project is created.

Next, the pom.xml will be updated:

- Add the Spring Boot parent
- Add the Spring Boot dependencies, including those for the SOAP web service and WSDL's
- Add the Java 1.8 property


dronebuzzers-soap-service/pom.xml

```xml
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>

    <groupId>com.dronebuzzers</groupId>
    <artifactId>dronebuzzers-soap-service</artifactId>
    <version>0.0.1-SNAPSHOT</version>
    <packaging>jar</packaging>

    <name>dronebuzzers-soap-service</name>
    <url>http://maven.apache.org</url>

    <parent>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-parent</artifactId>
        <version>2.0.0.RELEASE</version>
    </parent>

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

    <dependencies>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-web</artifactId>
        </dependency>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-web-services</artifactId>
        </dependency>
        <dependency>
            <groupId>wsdl4j</groupId>
            <artifactId>wsdl4j</artifactId>
        </dependency>
        <dependency>
            <groupId>junit</groupId>
            <artifactId>junit</artifactId>
            <version>3.8.1</version>
            <scope>test</scope>
        </dependency>
    </dependencies>
```

- … and add the build plugin:

```xml
</dependencies>

<build>
    <plugins>
        <plugin>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-maven-plugin</artifactId>
        </plugin>
    </plugins>
</build>
```

No need to type it all, just copy the pom.xml from
`<workspace>/SIGSpringBoot101/lab 7/input/pom.xml`

Finally, delete the generated App.java class (located in package: com.dronebuzzers.soap.service)

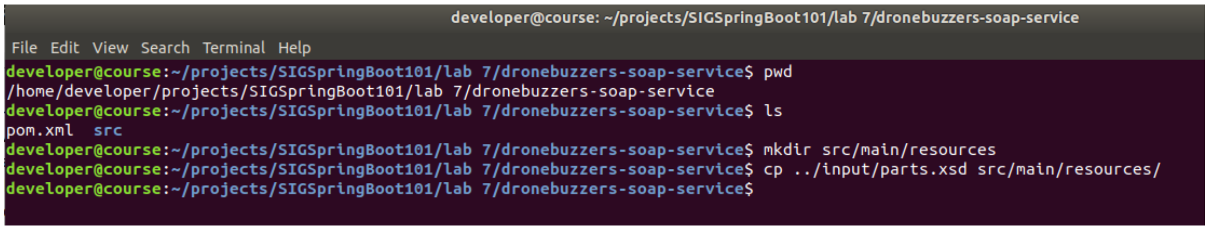
3. XML schema for messages

For a SOAP web service, we need an XML schema file that defines the business objects and the request and response messages that are used in the SOAP web service.

The XSD is available:

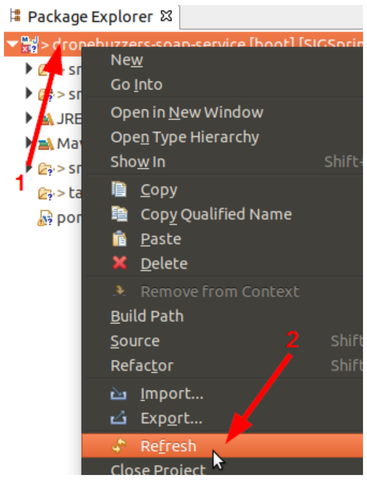
`<workspace>/SIGSpringBoot101/lab 7/input/parts.xsd`

You need to import it into the Eclipse STS project in the `resources` directory:

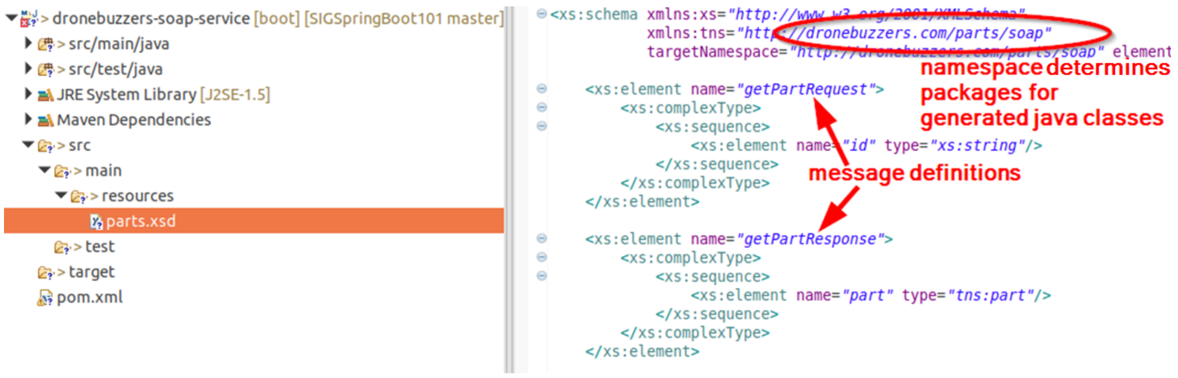


```
developer@course: ~/projects/SIGSpringBoot101/lab 7/dronebuzzers-soap-service
File  Edit  View  Search  Terminal  Help
developer@course:~/projects/SIGSpringBoot101/lab 7/dronebuzzers-soap-service$ pwd
/home/developer/projects/SIGSpringBoot101/lab 7/dronebuzzers-soap-service
developer@course:~/projects/SIGSpringBoot101/lab 7/dronebuzzers-soap-service$ ls
pom.xml  src
developer@course:~/projects/SIGSpringBoot101/lab 7/dronebuzzers-soap-service$ mkdir src/main/resources
developer@course:~/projects/SIGSpringBoot101/lab 7/dronebuzzers-soap-service$ cp ../input/parts.xsd src/main/resources/
developer@course:~/projects/SIGSpringBoot101/lab 7/dronebuzzers-soap-service$
```

Refresh the project in Eclipse STS by right-clicking it and click Refresh:

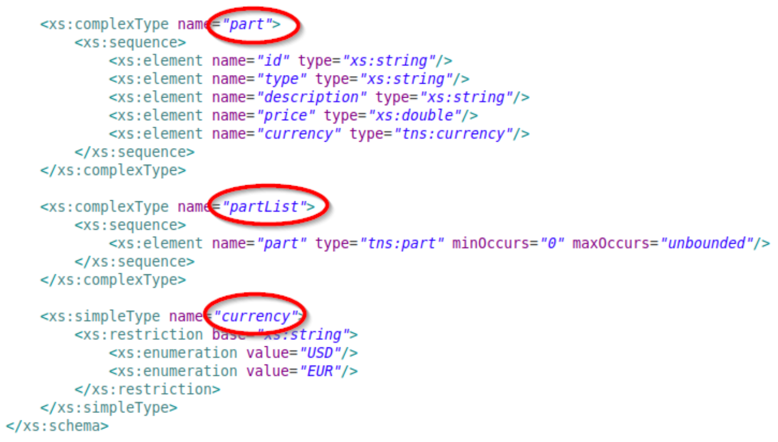


Now, open the `parts.xsd` so you can examine the message definitions:


dronebuzzers-soap-service [boot] [SIGSpringBoot101 master]
src/main/java
src/test/java
JRE System Library [J2SE-1.5]
Maven Dependencies
src
main
resources
parts.xsd
test
target
pom.xml
namespace determines packages for generated java classes
message definitions

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
           xmlns:tns="http://dronebuzzers.com/parts/soap"
           targetNamespace="http://dronebuzzers.com/parts/soap" element

    <xs:element name="getPartRequest">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="id" type="xs:string"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>

    <xs:element name="getPartResponse">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="part" type="tns:part"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
```

… and a bit further down the file the business object definitions:

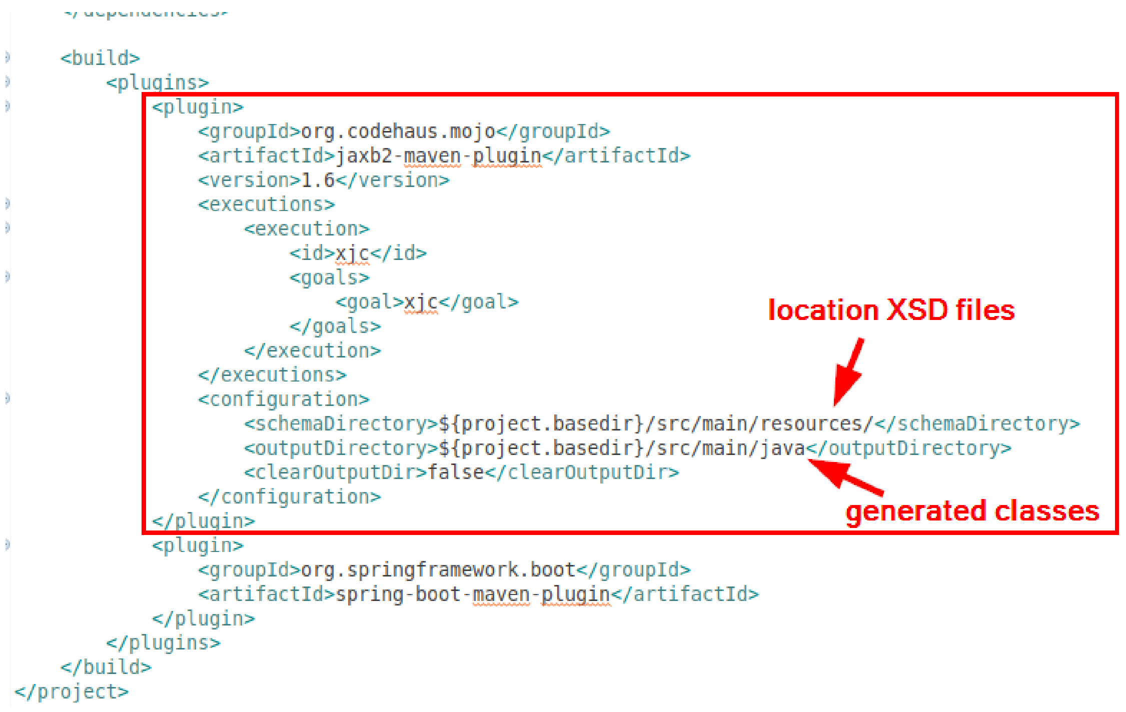


```
    <xs:complexType name="part">
        <xs:sequence>
            <xs:element name="id" type="xs:string"/>
            <xs:element name="type" type="xs:string"/>
            <xs:element name="description" type="xs:string"/>
            <xs:element name="price" type="xs:double"/>
            <xs:element name="currency" type="tns:currency"/>
        </xs:sequence>
    </xs:complexType>

    <xs:complexType name="partList">
        <xs:sequence>
            <xs:element name="part" type="tns:part" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>

    <xs:simpleType name="currency">
        <xs:restriction base="xs:string">
            <xs:enumeration value="USD"/>
            <xs:enumeration value="EUR"/>
        </xs:restriction>
    </xs:simpleType>
</xs:schema>
```

4. JAXB: Java classes for parts.xsd

In this section, we will use JAXB to create the Java classes for the messages and business objects defined in `parts.xsd`.

First, add the JAXB maven plugin in the pom.xml:

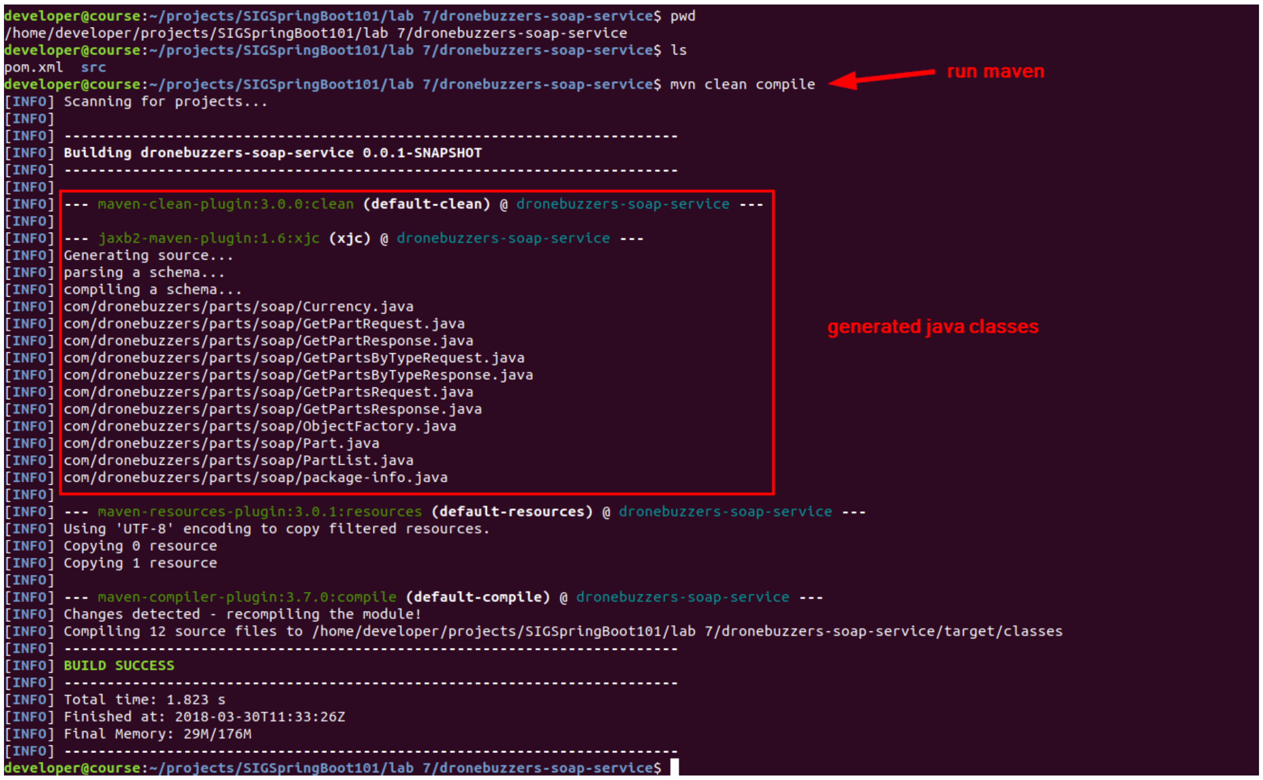


```
    <build>
        <plugins>
            <plugin>
                <groupId>org.codehaus.mojo</groupId>
                <artifactId>jaxb2-maven-plugin</artifactId>
                <version>1.6</version>
                <executions>
                    <execution>
                        <id>xjc</id>
                        <goals>
                            <goal>xjc</goal>
                        </goals>
                    </execution>
                </executions>
                <configuration>
                    <schemaDirectory>${project.basedir}/src/main/resources/</schemaDirectory>
                    <outputDirectory>${project.basedir}/src/main/java</outputDirectory>
                    <clearOutputDir>false</clearOutputDir>
                </configuration>
            </plugin>
            <plugin>
                <groupId>org.springframework.boot</groupId>
                <artifactId>spring-boot-maven-plugin</artifactId>
            </plugin>
        </plugins>
    </build>
</project>
```

Either type it in by hand… or copy it from the file:

`<workspace>/SIGSpringBoot101/lab 7/input/jaxb-plugin.xml`

Next, generate the domain classes with the JAXB plugin. Don't forget to save the updated pom.xml first! The command to run the JAXB plugin is `mvn clean compile` :



```
developer@course:~/projects/SIGSpringBoot101/lab 7/dronebuzzers-soap-service$ pwd
/home/developer/projects/SIGSpringBoot101/lab 7/dronebuzzers-soap-service
developer@course:~/projects/SIGSpringBoot101/lab 7/dronebuzzers-soap-service$ ls
pom.xml  src
developer@course:~/projects/SIGSpringBoot101/lab 7/dronebuzzers-soap-service$ mvn clean compile
[INFO] Scanning for projects...
[INFO]
[INFO] ------------------------------------------------------------------------
[INFO] Building dronebuzzers-soap-service 0.0.1-SNAPSHOT
[INFO] ------------------------------------------------------------------------
[INFO]
[INFO] --- maven-clean-plugin:3.0.0:clean (default-clean) @ dronebuzzers-soap-service ---
[INFO]
[INFO] --- jaxb2-maven-plugin:1.6:xjc (xjc) @ dronebuzzers-soap-service ---
[INFO] Generating source...
[INFO] parsing a schema...
[INFO] compiling a schema...
[INFO] com/dronebuzzers/parts/soap/Currency.java
[INFO] com/dronebuzzers/parts/soap/GetPartRequest.java
[INFO] com/dronebuzzers/parts/soap/GetPartResponse.java
[INFO] com/dronebuzzers/parts/soap/GetPartsByTypeRequest.java
[INFO] com/dronebuzzers/parts/soap/GetPartsByTypeResponse.java
[INFO] com/dronebuzzers/parts/soap/GetPartsRequest.java
[INFO] com/dronebuzzers/parts/soap/GetPartsResponse.java
[INFO] com/dronebuzzers/parts/soap/ObjectFactory.java
[INFO] com/dronebuzzers/parts/soap/Part.java
[INFO] com/dronebuzzers/parts/soap/PartList.java
[INFO] com/dronebuzzers/parts/soap/package-info.java
[INFO]
[INFO] --- maven-resources-plugin:3.0.1:resources (default-resources) @ dronebuzzers-soap-service ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] Copying 0 resource
[INFO] Copying 1 resource
[INFO]
[INFO] --- maven-compiler-plugin:3.7.0:compile (default-compile) @ dronebuzzers-soap-service ---
[INFO] Changes detected - recompiling the module!
[INFO] Compiling 12 source files to /home/developer/projects/SIGSpringBoot101/lab 7/dronebuzzers-soap-service/target/classes
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time: 1.823 s
[INFO] Finished at: 2018-03-30T11:33:26Z
[INFO] Final Memory: 29M/176M
[INFO] ------------------------------------------------------------------------
developer@course:~/projects/SIGSpringBoot101/lab 7/dronebuzzers-soap-service$
```

Note that the location-/-packages for the generated sources reflect the namespace. The packages for the namespace 'http://dronebuzzers.com/parts/soap' end up in package 'com.dronesbuzzers.parts.soap'.

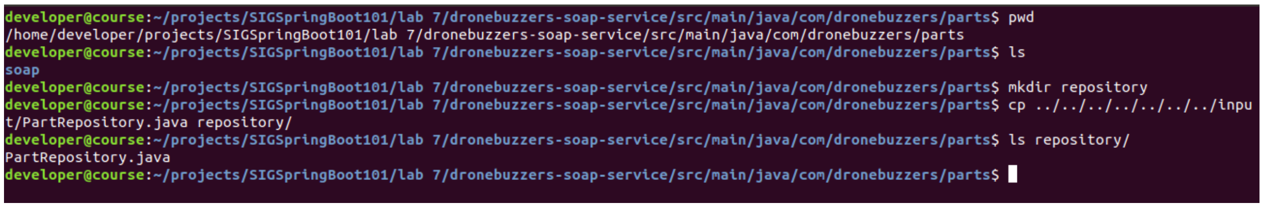
So far, all pretty straightforward JAXB stuff.

5. Parts repository

As the parts web service also needs access to parts data, we need to have a parts repository. For this lab, we created a simple java class that will be used as a repository. Nothing fancy, just copy it from:

`<workspace>/SIGSpringBoot101/lab 7/input/PartRepository.java`

Mind to put it in the right directory: `repository`



```
developer@course:~/projects/SIGSpringBoot101/lab 7/dronebuzzers-soap-service/src/main/java/com/dronebuzzers/parts$ pwd
/home/developer/projects/SIGSpringBoot101/lab 7/dronebuzzers-soap-service/src/main/java/com/dronebuzzers/parts
developer@course:~/projects/SIGSpringBoot101/lab 7/dronebuzzers-soap-service/src/main/java/com/dronebuzzers/parts$ ls
soap
developer@course:~/projects/SIGSpringBoot101/lab 7/dronebuzzers-soap-service/src/main/java/com/dronebuzzers/parts$ mkdir repository
developer@course:~/projects/SIGSpringBoot101/lab 7/dronebuzzers-soap-service/src/main/java/com/dronebuzzers/parts$ cp ../../../../../../../inpu
t/PartRepository.java repository/
developer@course:~/projects/SIGSpringBoot101/lab 7/dronebuzzers-soap-service/src/main/java/com/dronebuzzers/parts$ ls repository/
PartRepository.java
developer@course:~/projects/SIGSpringBoot101/lab 7/dronebuzzers-soap-service/src/main/java/com/dronebuzzers/parts$ 
```

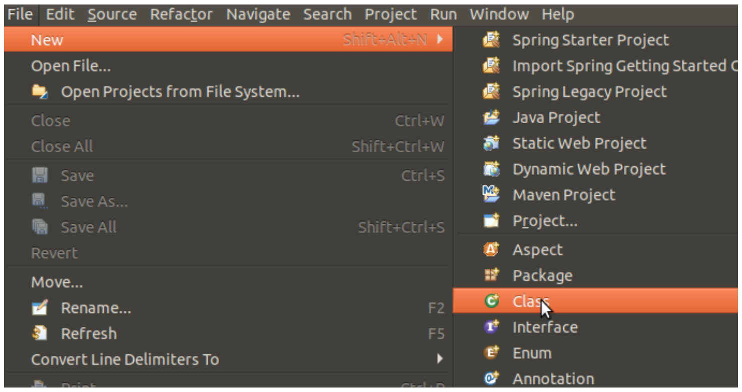
The PartRepository class has hard-coded drone parts and some methods for retrieving information about them.

Refresh the project!

6. Web service endpoint

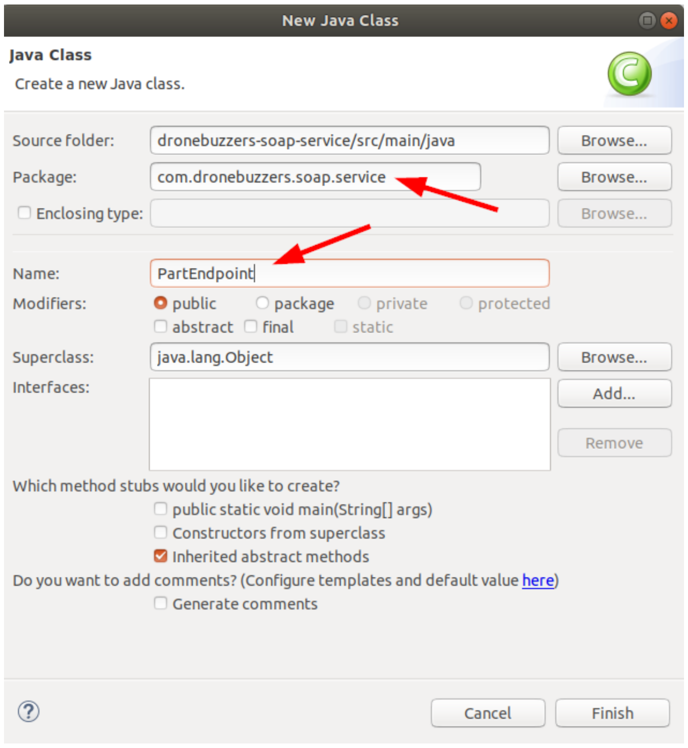
A SOAP web service has an endpoint. It is a simple java class with some Spring Boot annotations.

Create a new Java class by right-clicking the project, select New and then Class:



Complete like shown below:

- Source folder: dronebuzzers-soap-service/src/main/java
- Package: com.dronebuzzers.soap.service
- Name: PartEndpoint



New Java Class

Java Class
Create a new Java class.

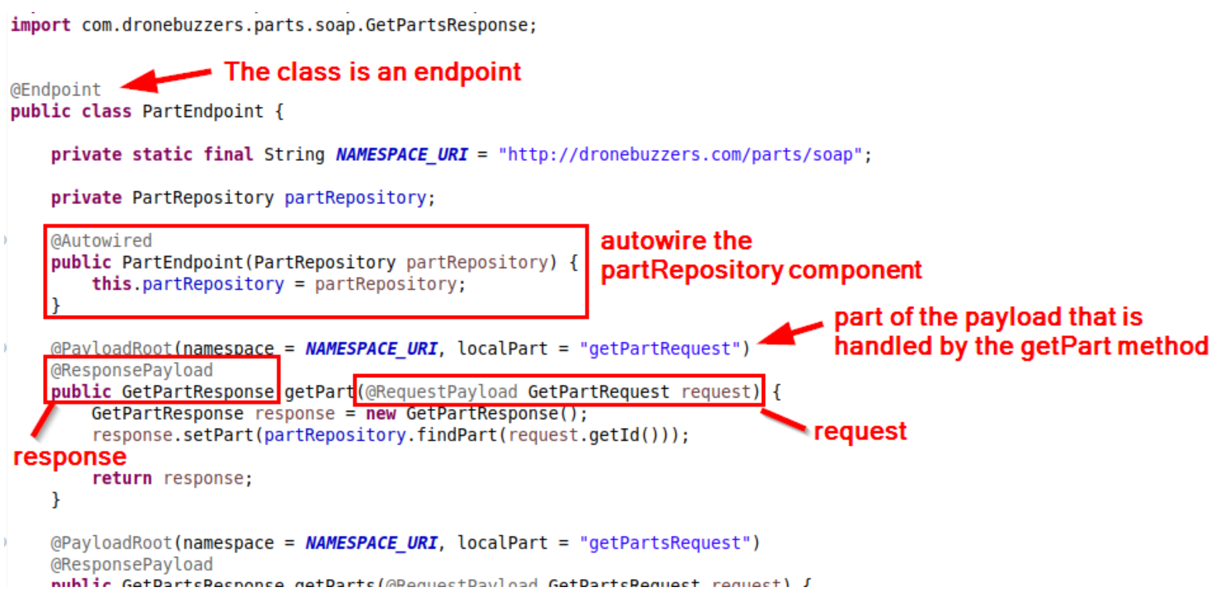
Source folder: dronebuzzers-soap-service/src/main/java
Package: com.dronebuzzers.soap.service
Enclosing type:
Name: PartEndpoint
Modifiers: [✓ public] package private protected
abstract final static
Superclass: java.lang.Object
Interfaces:
Which method stubs would you like to create?
public static void main(String[] args)
Constructors from superclass
[✓ Inherited abstract methods]
Do you want to add comments? (Configure templates and default value here)
Generate comments

Click Finish

In the resulting file, copy the contents of the file:

`<workspace>/SIGSpringBoot101/lab 7/input/PartEndpoint.java`

And examine the – annotations in – the code:



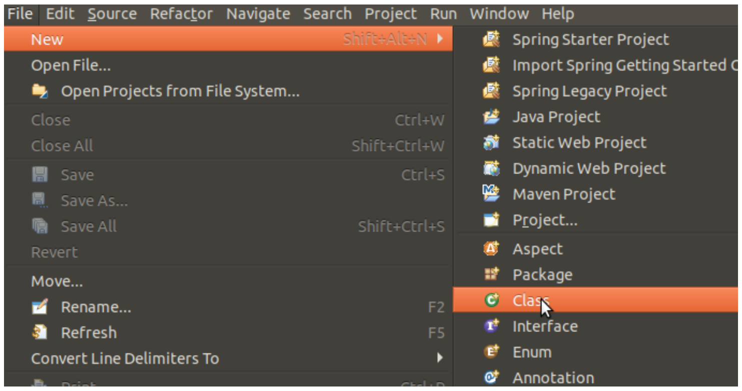
```
import com.dronebuzzers.parts.soap.GetPartsResponse;

@Endpoint
public class PartEndpoint {

    private static final String NAMESPACE_URI = "http://dronebuzzers.com/parts/soap";

    private PartRepository partRepository;

    @Autowired
    public PartEndpoint(PartRepository partRepository) {
        this.partRepository = partRepository;
    }

    @PayloadRoot(namespace = NAMESPACE_URI, localPart = "getPartRequest")
    @ResponsePayload
    public GetPartResponse getPart(@RequestPayload GetPartRequest request) {
        GetPartResponse response = new GetPartResponse();
        response.setPart(partRepository.findPart(request.getId()));

        return response;
    }

    @PayloadRoot(namespace = NAMESPACE_URI, localPart = "getPartsRequest")
    @ResponsePayload
```

The class is an endpoint
autowire the partRepository component
part of the payload that is handled by the getPart method
request
response

7. Web service configuration

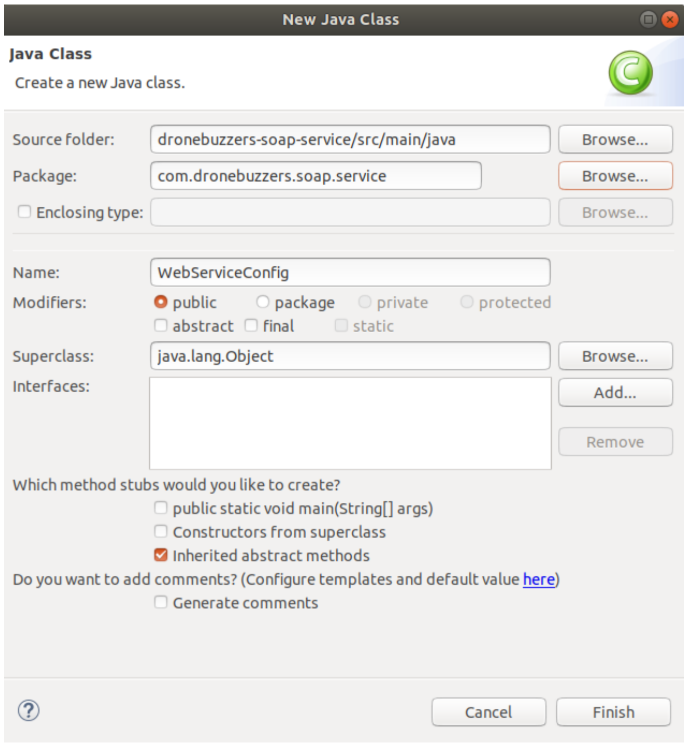
The previous section defined what the web service is doing on its endpoint. Now, we configure the web service endpoint itself.

Create a new java class by right-clicking the project, select New and then Class:



Complete like shown below:

- Source folder: dronebuzzers-soap-service/src/main/java
- Package: com.dronebuzzers.soap.service
- Name: WebServiceConfig



Java Class

Create a new Java class.

Source folder: dronebuzzers-soap-service/src/main/java

Package: com.dronebuzzers.soap.service

Enclosing type: [ ]

Name: WebServiceConfig

Modifiers: [✓ public] package private protected

[ ] abstract [ ] final [ ] static

Superclass: java.lang.Object

Interfaces:

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)

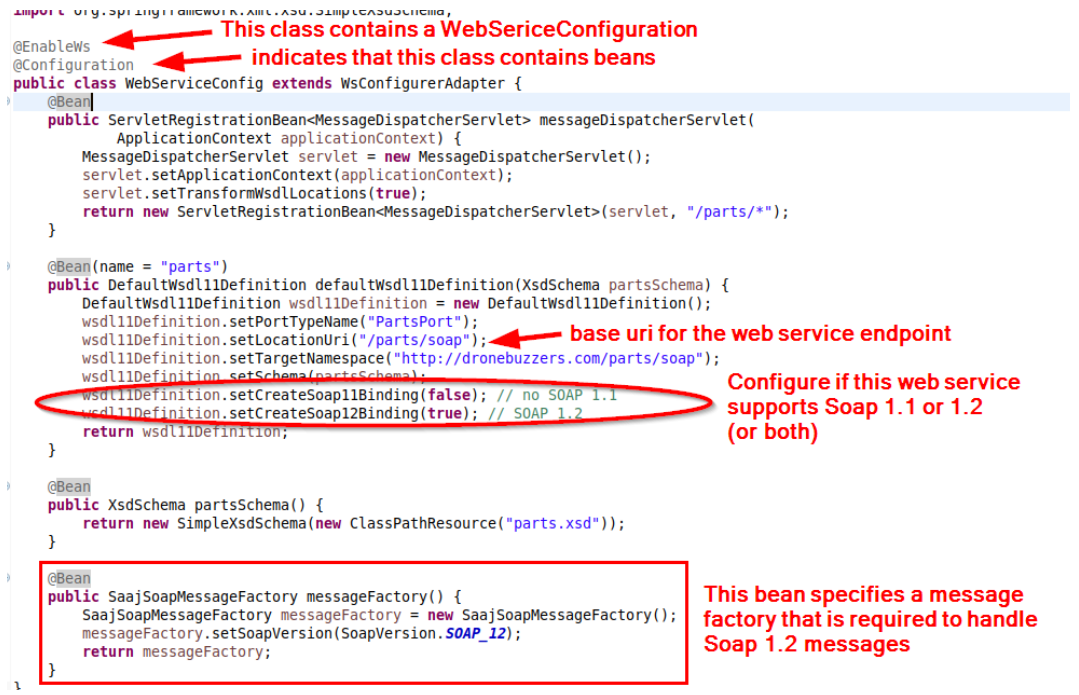
[ ] Generate comments

Click Finish

In the resulting file, copy the contents of the file:

```
<workspace>/SIGSpringBoot101/lab 7/input/WebServiceConfig.java
```

And examine the annotations in the code:



```
import org.springframework.xml.xsd.SimpleXsdSchema;

@EnableWs
@Configuration
public class WebServiceConfig extends WsConfigurerAdapter {
    @Bean
    public ServletRegistrationBean<MessageDispatcherServlet> messageDispatcherServlet(
            ApplicationContext applicationContext) {
        MessageDispatcherServlet servlet = new MessageDispatcherServlet();
        servlet.setApplicationContext(applicationContext);
        servlet.setTransformWsdlLocations(true);
        return new ServletRegistrationBean<MessageDispatcherServlet>(servlet, "/parts/*");
    }

    @Bean(name = "parts")
    public DefaultWsdl11Definition defaultWsdl11Definition(XsdSchema partsSchema) {
        DefaultWsdl11Definition wsdl11Definition = new DefaultWsdl11Definition();
        wsdl11Definition.setPortTypeName("PartsPort");
        wsdl11Definition.setLocationUri("/parts/soap");
        wsdl11Definition.setTargetNamespace("http://dronebuzzers.com/parts/soap");
        wsdl11Definition.setSchema(partsSchema);
        wsdl11Definition.setCreateSoap11Binding(false); // no SOAP 1.1
        wsdl11Definition.setCreateSoap12Binding(true); // SOAP 1.2
        return wsdl11Definition;
    }

    @Bean
    public XsdSchema partsSchema() {
        return new SimpleXsdSchema(new ClassPathResource("parts.xsd"));
    }

    @Bean
    public SaajSoapMessageFactory messageFactory() {
        SaajSoapMessageFactory messageFactory = new SaajSoapMessageFactory();
        messageFactory.setSoapVersion(SoapVersion.SOAP_12);
        return messageFactory;
    }
}
```

This class contains a WebSericeConfiguration

indicates that this class contains beans

base uri for the web service endpoint

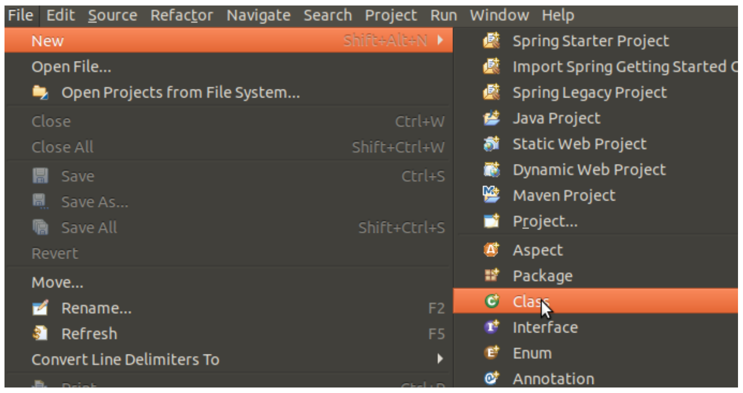
Configure if this web service supports Soap 1.1 or 1.2 (or both)

This bean specifies a message factory that is required to handle Soap 1.2 messages

8. Runnable application

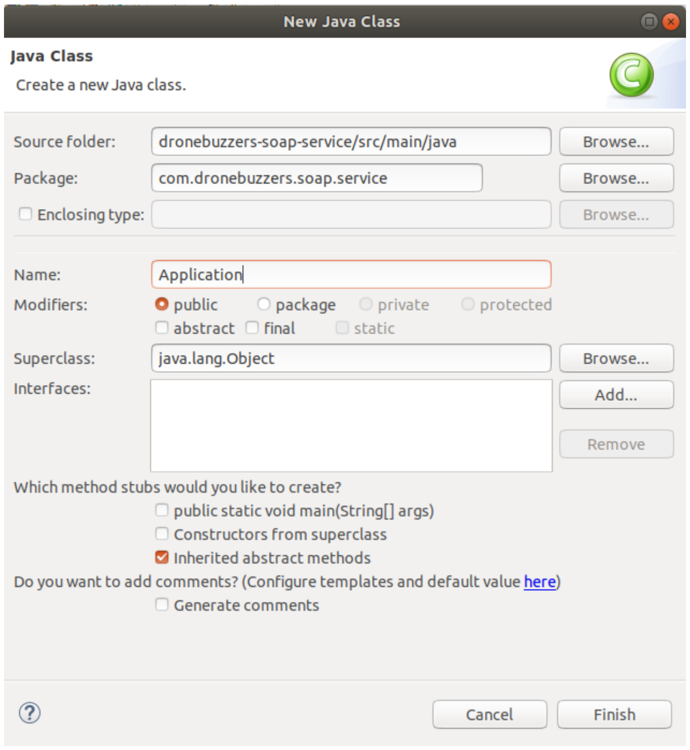
In order to turn this project into a runnable application, create the class Application.

Create a new java class by right-clicking the project, select New and then Class:



Complete like shown below:

- Source folder: dronebuzzers-soap-service/src/main/java
- Package: com.dronebuzzers.soap.service
- Name: Application



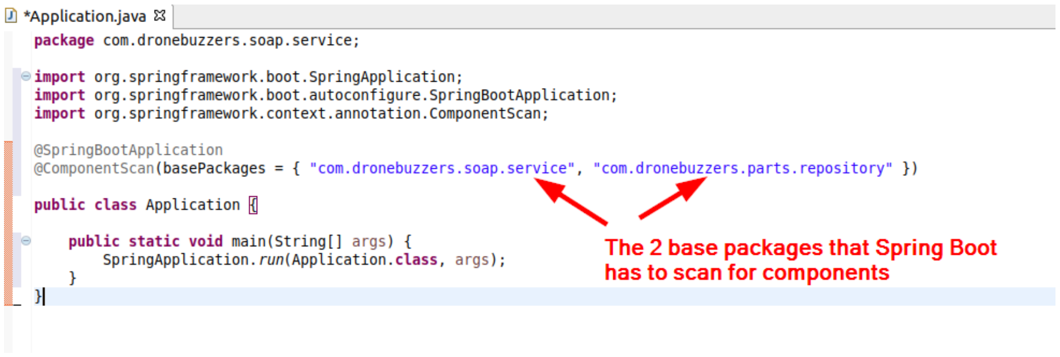
Source folder: dronebuzzers-soap-service/src/main/java
Package: com.dronebuzzers.soap.service
Enclosing type:
Name: Application
Modifiers: [✓ public] package private protected
abstract final static
Superclass: java.lang.Object
Interfaces:

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

In the resulting file, copy the contents of the file:

`<workspace>/SIGSpringBoot101/lab 7/input/Application.java`

And examine the annotations in the code:



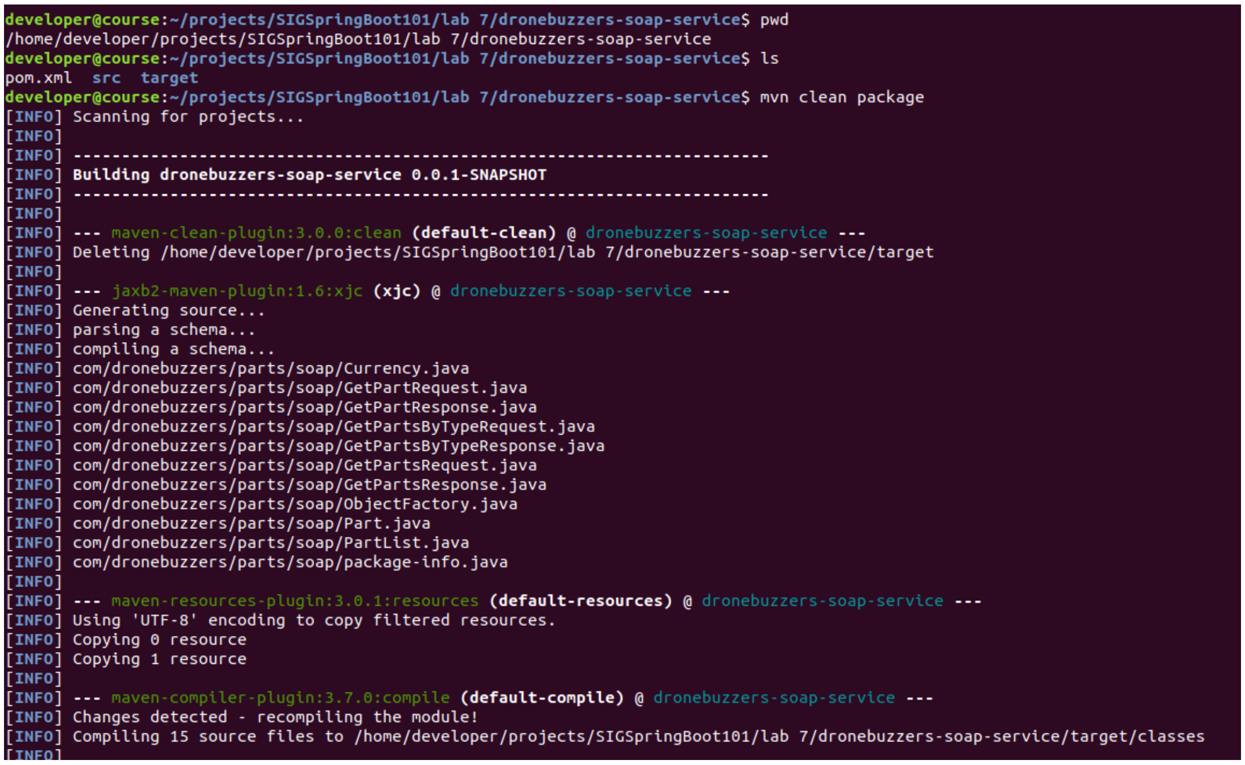
```
package com.dronebuzzers.soap.service;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.context.annotation.ComponentScan;

@SpringBootApplication
@ComponentScan(basePackages = { "com.dronebuzzers.soap.service", "com.dronebuzzers.parts.repository" })

public class Application {

    public static void main(String[] args) {
        SpringApplication.run(Application.class, args);
    }
}
```

The 2 base packages that Spring Boot has to scan for components

After saving all files in Eclipse STS, the implementation of the SOAP web service is finished.

9. Run and Test the Soap web service

Let’s start the web service from the command line, using `mvn clean package`[1]:

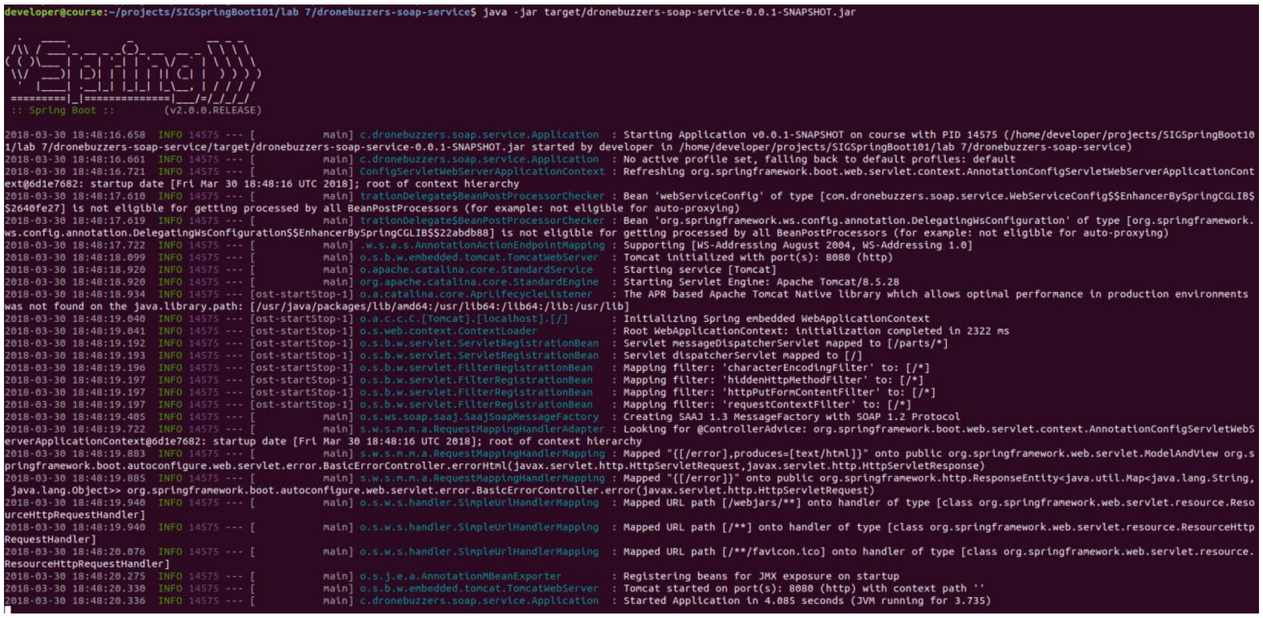


```
developer@course:~/projects/SIGSpringBoot101/lab 7/dronebuzzers-soap-service$ pwd
/home/developer/projects/SIGSpringBoot101/lab 7/dronebuzzers-soap-service
developer@course:~/projects/SIGSpringBoot101/lab 7/dronebuzzers-soap-service$ ls
pom.xml  src  target
developer@course:~/projects/SIGSpringBoot101/lab 7/dronebuzzers-soap-service$ mvn clean package
[INFO] Scanning for projects...
[INFO]
[INFO] ------------------------------------------------------------------------
[INFO] Building dronebuzzers-soap-service 0.0.1-SNAPSHOT
[INFO] ------------------------------------------------------------------------
[INFO]
[INFO] --- maven-clean-plugin:3.0.0:clean (default-clean) @ dronebuzzers-soap-service ---
[INFO] Deleting /home/developer/projects/SIGSpringBoot101/lab 7/dronebuzzers-soap-service/target
[INFO]
[INFO] --- jaxb2-maven-plugin:1.6:xjc (xjc) @ dronebuzzers-soap-service ---
[INFO] Generating source...
[INFO] parsing a schema...
[INFO] compiling a schema...
[INFO] com/dronebuzzers/parts/soap/Currency.java
[INFO] com/dronebuzzers/parts/soap/GetPartRequest.java
[INFO] com/dronebuzzers/parts/soap/GetPartResponse.java
[INFO] com/dronebuzzers/parts/soap/GetPartsByTypeRequest.java
[INFO] com/dronebuzzers/parts/soap/GetPartsByTypeResponse.java
[INFO] com/dronebuzzers/parts/soap/GetPartsRequest.java
[INFO] com/dronebuzzers/parts/soap/GetPartsResponse.java
[INFO] com/dronebuzzers/parts/soap/ObjectFactory.java
[INFO] com/dronebuzzers/parts/soap/Part.java
[INFO] com/dronebuzzers/parts/soap/PartList.java
[INFO] com/dronebuzzers/parts/soap/package-info.java
[INFO]
[INFO] --- maven-resources-plugin:3.0.1:resources (default-resources) @ dronebuzzers-soap-service ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] Copying 0 resource
[INFO] Copying 1 resource
[INFO]
[INFO] --- maven-compiler-plugin:3.7.0:compile (default-compile) @ dronebuzzers-soap-service ---
[INFO] Changes detected - recompiling the module!
[INFO] Compiling 15 source files to /home/developer/projects/SIGSpringBoot101/lab 7/dronebuzzers-soap-service/target/classes
[INFO]
```

Next, the web service can be started with the command:

```
java -jar target/dronebuzzers-soap-service-0.0.1-SNAPSHOT.jar
```

[1] Here, we run the service from the command line. In (most?) other labs, we run the services from Eclipse STS. Here we chose to run from command line for no particular reason. You can also run the web service from within Eclipse STS



```
developer@course:~/projects/SIGSpringBoot101/lab 7/dronebuzzers-soap-service$ java -jar target/dronebuzzers-soap-service-0.0.1-SNAPSHOT.jar

  .   ____          _            __ _ _
 /\\ / ___'_ __ _ _(_)_ __  __ _ \ \ \ \
( ( )\___ | '_ | '_| | '_ \/ _` | \ \ \ \
 \\/  ___)| |_)| | | | | || (_| |  ) ) ) )
  '  |____| .__|_| |_|_| |_\__, | / / / /
 =========|_|==============|___/=/_/_/_/
 :: Spring Boot ::        (v2.0.0.RELEASE)

2018-03-30 18:48:16.658  INFO 14575 --- [           main] c.dronebuzzers.soap.service.Application  : Starting Application v0.0.1-SNAPSHOT on course with PID 14575 (/home/developer/projects/SIGSpringBoot10
1/lab 7/dronebuzzers-soap-service/target/dronebuzzers-soap-service-0.0.1-SNAPSHOT.jar started by developer in /home/developer/projects/SIGSpringBoot101/lab 7/dronebuzzers-soap-service)
2018-03-30 18:48:16.661  INFO 14575 --- [           main] c.dronebuzzers.soap.service.Application  : No active profile set, falling back to default profiles: default
2018-03-30 18:48:16.721  INFO 14575 --- [           main] ConfigServletWebServerApplicationContext : Refreshing org.springframework.boot.web.servlet.context.AnnotationConfigServletWebServerApplicationCont
ext@6d1e7682: startup date [Fri Mar 30 18:48:16 UTC 2018]; root of context hierarchy
2018-03-30 18:48:17.610  INFO 14575 --- [           main] trationDelegate$BeanPostProcessorChecker : Bean 'webServiceConfig' of type [com.dronebuzzers.soap.service.WebServiceConfig$$EnhancerBySpringCGLIB$
$2640fe27] is not eligible for getting processed by all BeanPostProcessors (for example: not eligible for auto-proxying)
2018-03-30 18:48:17.619  INFO 14575 --- [           main] trationDelegate$BeanPostProcessorChecker : Bean 'org.springframework.ws.config.annotation.DelegatingWsConfiguration' of type [org.springframework.
ws.config.annotation.DelegatingWsConfiguration$$EnhancerBySpringCGLIB$$22abdb88] is not eligible for getting processed by all BeanPostProcessors (for example: not eligible for auto-proxying)
2018-03-30 18:48:17.722  INFO 14575 --- [           main] .w.s.a.s.AnnotationActionEndpointMapping : Supporting [WS-Addressing August 2004, WS-Addressing 1.0]
2018-03-30 18:48:18.099  INFO 14575 --- [           main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat initialized with port(s): 8080 (http)
2018-03-30 18:48:18.920  INFO 14575 --- [           main] o.apache.catalina.core.StandardService   : Starting service [Tomcat]
2018-03-30 18:48:18.920  INFO 14575 --- [           main] org.apache.catalina.core.StandardEngine  : Starting Servlet Engine: Apache Tomcat/8.5.28
2018-03-30 18:48:18.934  INFO 14575 --- [ost-startStop-1] o.a.catalina.core.AprLifecycleListener   : The APR based Apache Tomcat Native library which allows optimal performance in production environments
was not found on the java.library.path: [/usr/java/packages/lib/amd64:/usr/lib64:/lib64:/lib:/usr/lib]
2018-03-30 18:48:19.040  INFO 14575 --- [ost-startStop-1] o.a.c.c.C.[Tomcat].[localhost].[/]       : Initializing Spring embedded WebApplicationContext
2018-03-30 18:48:19.041  INFO 14575 --- [ost-startStop-1] o.s.web.context.ContextLoader            : Root WebApplicationContext: initialization completed in 2322 ms
2018-03-30 18:48:19.192  INFO 14575 --- [ost-startStop-1] o.s.b.w.servlet.ServletRegistrationBean  : Servlet messageDispatcherServlet mapped to [/parts/*]
2018-03-30 18:48:19.193  INFO 14575 --- [ost-startStop-1] o.s.b.w.servlet.ServletRegistrationBean  : Servlet dispatcherServlet mapped to [/]
2018-03-30 18:48:19.196  INFO 14575 --- [ost-startStop-1] o.s.b.w.servlet.FilterRegistrationBean   : Mapping filter: 'characterEncodingFilter' to: [/*]
2018-03-30 18:48:19.197  INFO 14575 --- [ost-startStop-1] o.s.b.w.servlet.FilterRegistrationBean   : Mapping filter: 'hiddenHttpMethodFilter' to: [/*]
2018-03-30 18:48:19.197  INFO 14575 --- [ost-startStop-1] o.s.b.w.servlet.FilterRegistrationBean   : Mapping filter: 'httpPutFormContentFilter' to: [/*]
2018-03-30 18:48:19.197  INFO 14575 --- [ost-startStop-1] o.s.b.w.servlet.FilterRegistrationBean   : Mapping filter: 'requestContextFilter' to: [/*]
2018-03-30 18:48:19.405  INFO 14575 --- [           main] o.s.ws.soap.saaj.SaajSoapMessageFactory  : Creating SAAJ 1.3 MessageFactory with SOAP 1.2 Protocol
2018-03-30 18:48:19.722  INFO 14575 --- [           main] s.w.s.m.m.a.RequestMappingHandlerAdapter : Looking for @ControllerAdvice: org.springframework.boot.web.servlet.context.AnnotationConfigServletWebS
erverApplicationContext@6d1e7682: startup date [Fri Mar 30 18:48:16 UTC 2018]; root of context hierarchy
2018-03-30 18:48:19.883  INFO 14575 --- [           main] s.w.s.m.m.a.RequestMappingHandlerMapping : Mapped "{[/error],produces=[text/html]}" onto public org.springframework.web.servlet.ModelAndView org.s
pringframework.boot.autoconfigure.web.servlet.error.BasicErrorController.errorHtml(javax.servlet.http.HttpServletRequest,javax.servlet.http.HttpServletResponse)
2018-03-30 18:48:19.885  INFO 14575 --- [           main] s.w.s.m.m.a.RequestMappingHandlerMapping : Mapped "{[/error]}" onto public org.springframework.http.ResponseEntity<java.util.Map<java.lang.String,
 java.lang.Object>> org.springframework.boot.autoconfigure.web.servlet.error.BasicErrorController.error(javax.servlet.http.HttpServletRequest)
2018-03-30 18:48:19.940  INFO 14575 --- [           main] o.s.w.s.handler.SimpleUrlHandlerMapping  : Mapped URL path [/webjars/**] onto handler of type [class org.springframework.web.servlet.resource.Reso
urceHttpRequestHandler]
2018-03-30 18:48:19.940  INFO 14575 --- [           main] o.s.w.s.handler.SimpleUrlHandlerMapping  : Mapped URL path [/**] onto handler of type [class org.springframework.web.servlet.resource.ResourceHttp
RequestHandler]
2018-03-30 18:48:20.076  INFO 14575 --- [           main] o.s.w.s.handler.SimpleUrlHandlerMapping  : Mapped URL path [/**/favicon.ico] onto handler of type [class org.springframework.web.servlet.resource.
ResourceHttpRequestHandler]
2018-03-30 18:48:20.275  INFO 14575 --- [           main] o.s.j.e.a.AnnotationMBeanExporter        : Registering beans for JMX exposure on startup
2018-03-30 18:48:20.330  INFO 14575 --- [           main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat started on port(s): 8080 (http) with context path ''
2018-03-30 18:48:20.336  INFO 14575 --- [           main] c.dronebuzzers.soap.service.Application  : Started Application in 4.085 seconds (JVM running for 3.735)
```

With the service up and running, you can have a look at the WSDL by

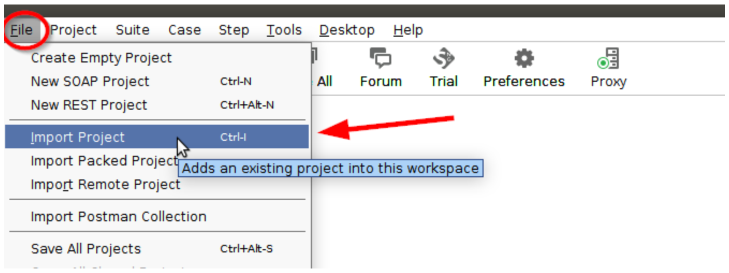
http://localhost:8080/parts/soap/parts.wsdl

The testing work can be done with SoapUI. Start SoapUI by clicking on

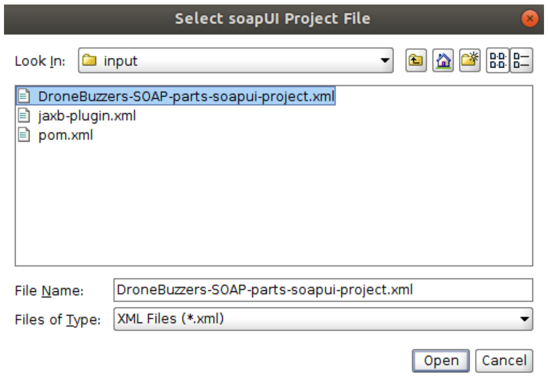
SoapUI will start. Next, import the sample project:

```
<workspace>/SIGSpringBoot101/lab 7/input/DroneBuzzers-SOAP-parts-soapui-project.xml
```

In SoapUI, click File – Import Project:



Select the sample project:



Select soapUI Project File

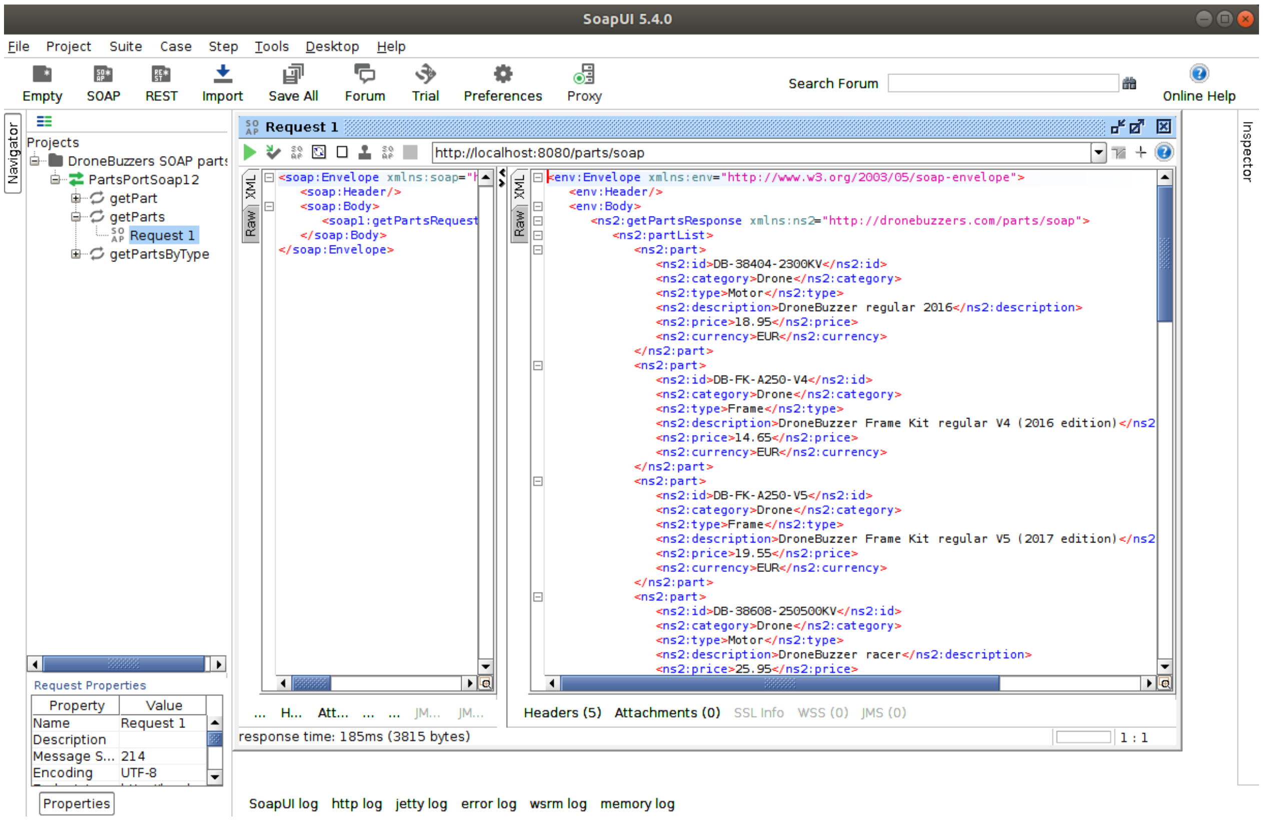
Look In: input

DroneBuzzers-SOAP-parts-soapui-project.xml
jaxb-plugin.xml
pom.xml

File Name: DroneBuzzers-SOAP-parts-soapui-project.xml

Files of Type: XML Files (*.xml)

And open it. Next, the 3 operations can be used for testing the parts Soap web service:



SoapUI 5.4.0

File Project Suite Case Step Tools Desktop Help
Empty SOAP REST Import Save All Forum Trial Preferences Proxy
Search Forum
Online Help
Navigator
Inspector

Projects
- DroneBuzzers SOAP parts
  - PartsPortSoap12
    - getPart
    - getParts
      - Request 1
    - getPartsByType

Request 1

http://localhost:8080/parts/soap

```
<soap:Envelope xmlns:soap="h
   <soap:Header/>
   <soap:Body>
      <soap1:getPartsRequest
   </soap:Body>
</soap:Envelope>
```

```
<env:Envelope xmlns:env="http://www.w3.org/2003/05/soap-envelope">
   <env:Header/>
   <env:Body>
      <ns2:getPartsResponse xmlns:ns2="http://dronebuzzers.com/parts/soap">
         <ns2:partList>
            <ns2:part>
               <ns2:id>DB-38404-2300KV</ns2:id>
               <ns2:category>Drone</ns2:category>
               <ns2:type>Motor</ns2:type>
               <ns2:description>DroneBuzzer regular 2016</ns2:description>
               <ns2:price>18.95</ns2:price>
               <ns2:currency>EUR</ns2:currency>
            </ns2:part>
            <ns2:part>
               <ns2:id>DB-FK-A250-V4</ns2:id>
               <ns2:category>Drone</ns2:category>
               <ns2:type>Frame</ns2:type>
               <ns2:description>DroneBuzzer Frame Kit regular V4 (2016 edition)</ns2
               <ns2:price>14.65</ns2:price>
               <ns2:currency>EUR</ns2:currency>
            </ns2:part>
            <ns2:part>
               <ns2:id>DB-FK-A250-V5</ns2:id>
               <ns2:category>Drone</ns2:category>
               <ns2:type>Frame</ns2:type>
               <ns2:description>DroneBuzzer Frame Kit regular V5 (2017 edition)</ns2
               <ns2:price>19.55</ns2:price>
               <ns2:currency>EUR</ns2:currency>
            </ns2:part>
            <ns2:part>
               <ns2:id>DB-38608-250500KV</ns2:id>
               <ns2:category>Drone</ns2:category>
               <ns2:type>Motor</ns2:type>
               <ns2:description>DroneBuzzer racer</ns2:description>
               <ns2:price>25.95</ns2:price>
```

... H... Att... ... ... JM... JM...
Headers (5) Attachments (0) SSL Info WSS (0) JMS (0)

response time: 185ms (3815 bytes)

1 : 1

Request Properties

Property	Value
Name	Request 1
Description	
Message S...	214
Encoding	UTF-8