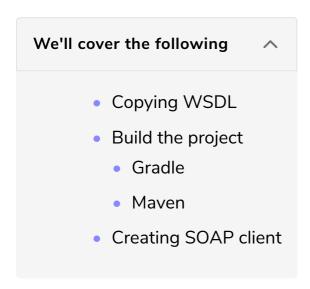
SOAP Client for Making Web Service Calls

In this lesson, we will see how to copy the WSDL document, build the project and create the SOAP client for making web service calls over HTTP.



This lesson is a continuation of *Creating SOAP Client Project* lesson.

Copying WSDL#

For creating the web service client, we need the WSDL file that holds the information about the web service request-response formats and where the service is hosted.

A sample WDSL file is already given in **SOAP Messages and WSDL** lesson.

Download the WSDL XML document for the web service under test and place it under src/main/resources. For the demonstration, let's assume the name of the WSDL file is students.wsdl.

Please ensure the WSDL file name and the location is mentioned correctly in **build.gradle** in the case of **Gradle** or **pom.xml** in the case of **Maven**.

Build the project

Gradle #

Please ensure the current folder structure looks like the following:

```
build.gradle
src
main
sources
students.wsdl
test
java
resources
resources
```

After adding Gradle to classpath and creating build.gradle with the above content, run the following command to build the project.

```
gradle clean build
```

After running the above command, all the request-response classes in accordance to the src/main/resources/students.wsdl will be generated under build/generated-sources/jaxb.

Maven

Please ensure the current folder structure looks like the following:

```
- pom.xml
- src
- main
- java
- resources
- students.wsdl
- test
- java
- resources
```

After adding Maven to classpath and creating pom.xml with the above content, run the following command to build the project.

```
mvn clean compile
```

After running the above command, all the request-response classes in accordance to the src/main/resources/students.wsdl will be generated under target/generated-sources/jaxb.

Creating SOAP client

Create a package io.educative.soap under src/main/java, create

WebServiceClient.java under io.educative.soap and copy the following contents into it.

```
package io.educative.soap;
import javax.annotation.PostConstruct;
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import org.springframework.oxm.jaxb.Jaxb2Marshaller;
import org.springframework.util.ClassUtils;
import org.springframework.ws.client.core.WebServiceTemplate;
import io.educative.soap_automation.GetStudentsRequest;
@Configuration
public class WebServiceClient {
    private Jaxb2Marshaller marshaller = new Jaxb2Marshaller();
    @PostConstruct
    public void init() throws Exception {
       marshaller.setPackagesToScan(ClassUtils.getPackageName(GetStudentsRequ
est.class));
       marshaller.afterPropertiesSet();
    }
    public WebServiceTemplate getWebServiceTemplate() {
        return new WebServiceTemplate(marshaller);
```

}

The import statements and GetStudentsRequest.class might change depending upon the SOAP request class.

Jaxb2Marshaller instance is needed to convert the response XML to Java objects.

WebServiceTemplate is used to make web service calls, which we will learn in the next lesson.

In this lesson, we learned how to create a test project, get the WSDL file, build the project and create the web service client. In the next lesson, we will learn how to use this web service client to make web service calls over HTTP.