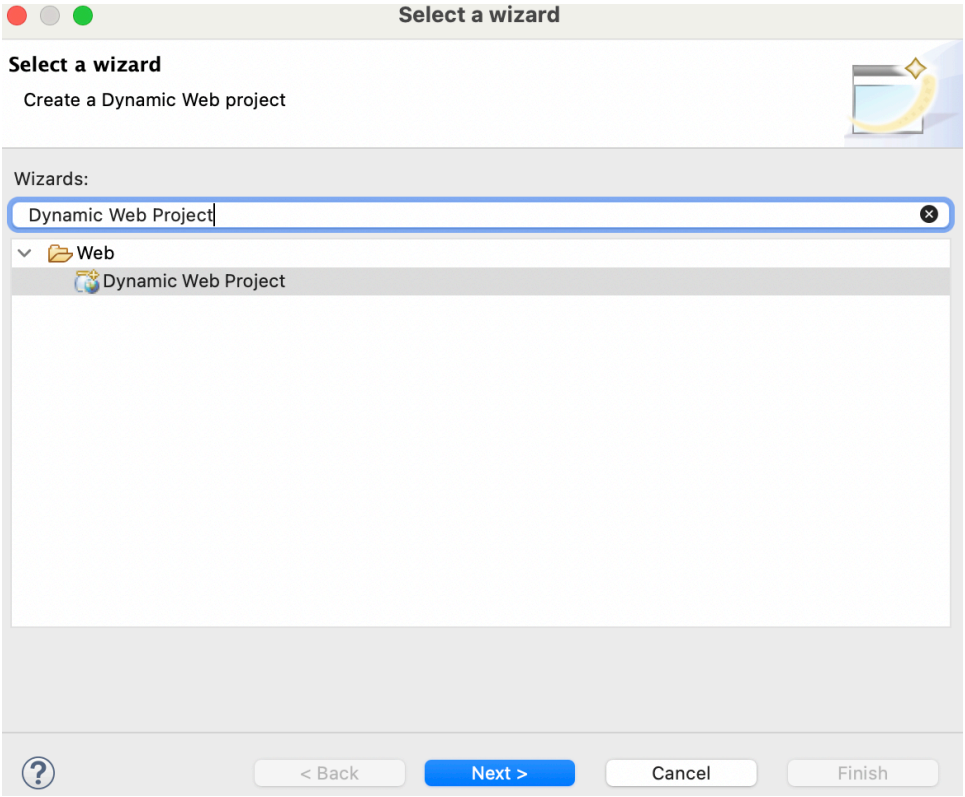
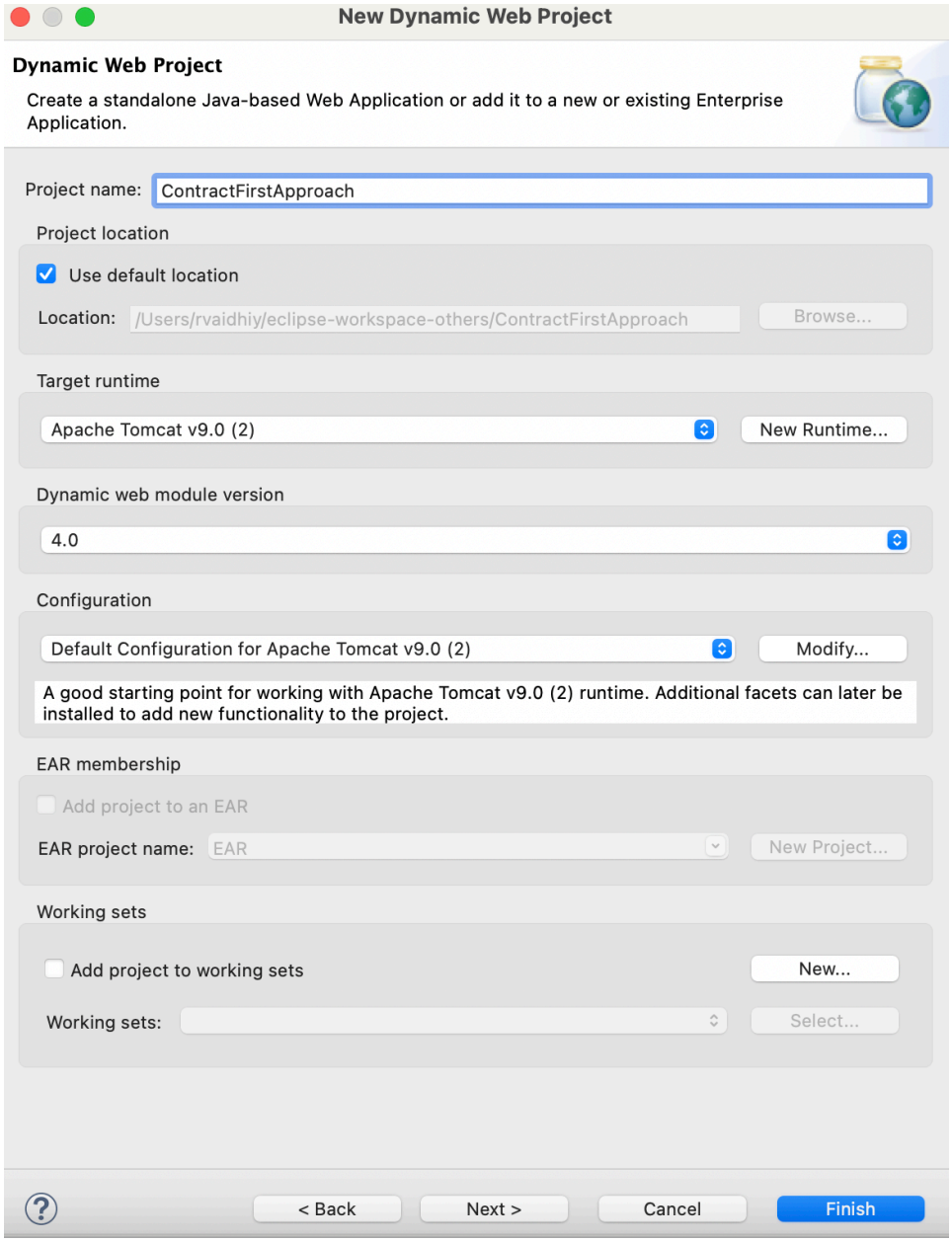
Java Version recommended: 1.8
IDE: Eclipse
Server: Tomcat (installed in Eclipse) version 9

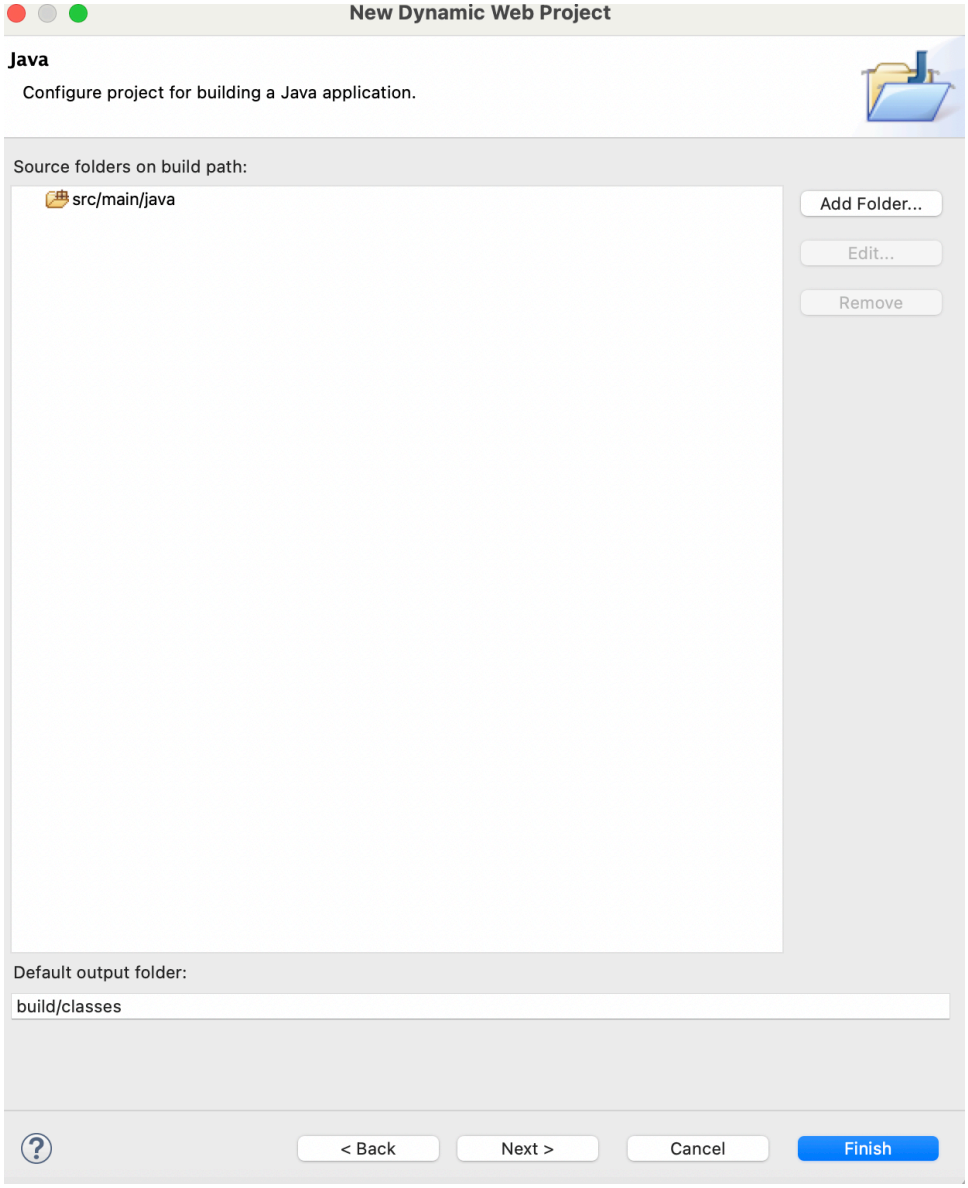
Creating the Project:

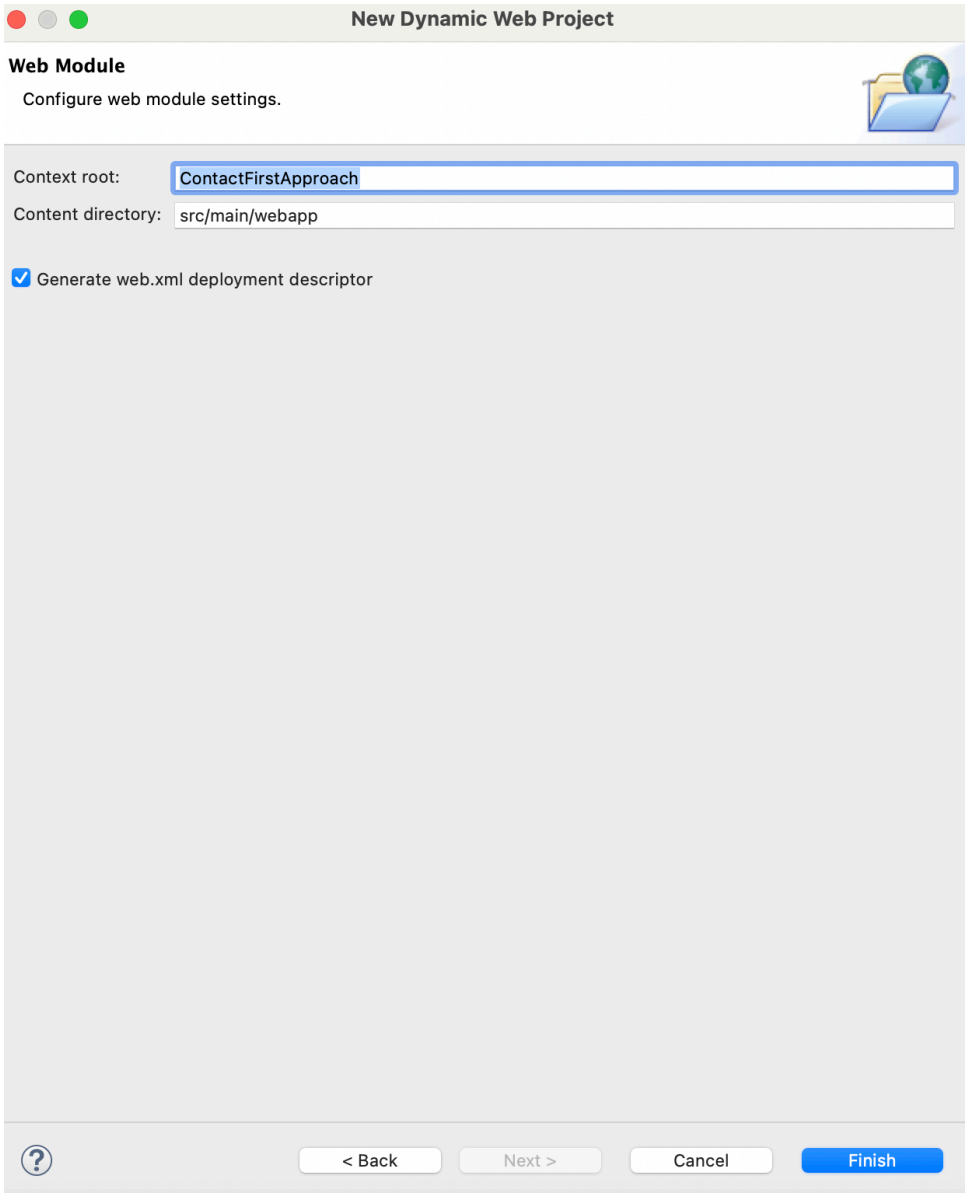
Create a new **Dynamic Web Project**



Provide a name for our project **ContractFirstApproach.**
Rest of the settings will be set to default (as long as the correct Tomcat version is set).


New Dynamic Web Project
Dynamic Web Project
Create a standalone Java-based Web Application or add it to a new or existing Enterprise Application.
Project name:
ContractFirstApproach
Project location
Use default location
Location:
/Users/rvaidhiy/eclipse-workspace-others/ContractFirstApproach
Browse...
Target runtime
Apache Tomcat v9.0 (2)
New Runtime...
Dynamic web module version
4.0
Configuration
Default Configuration for Apache Tomcat v9.0 (2)
Modify...
A good starting point for working with Apache Tomcat v9.0 (2) runtime. Additional facets can later be installed to add new functionality to the project.
EAR membership
Add project to an EAR
EAR project name:
EAR
New Project...
Working sets
Add project to working sets
New...
Working sets:
Select...
< Back
Next >
Cancel
Finish

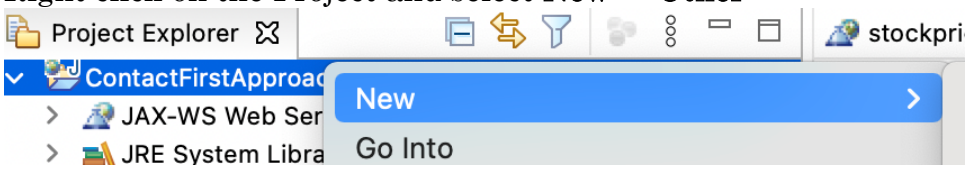

New Dynamic Web Project
Java
Configure project for building a Java application.
Source folders on build path:
src/main/java
Add Folder...
Edit...
Remove
Default output folder:
build/classes
< Back
Next >
Cancel
Finish



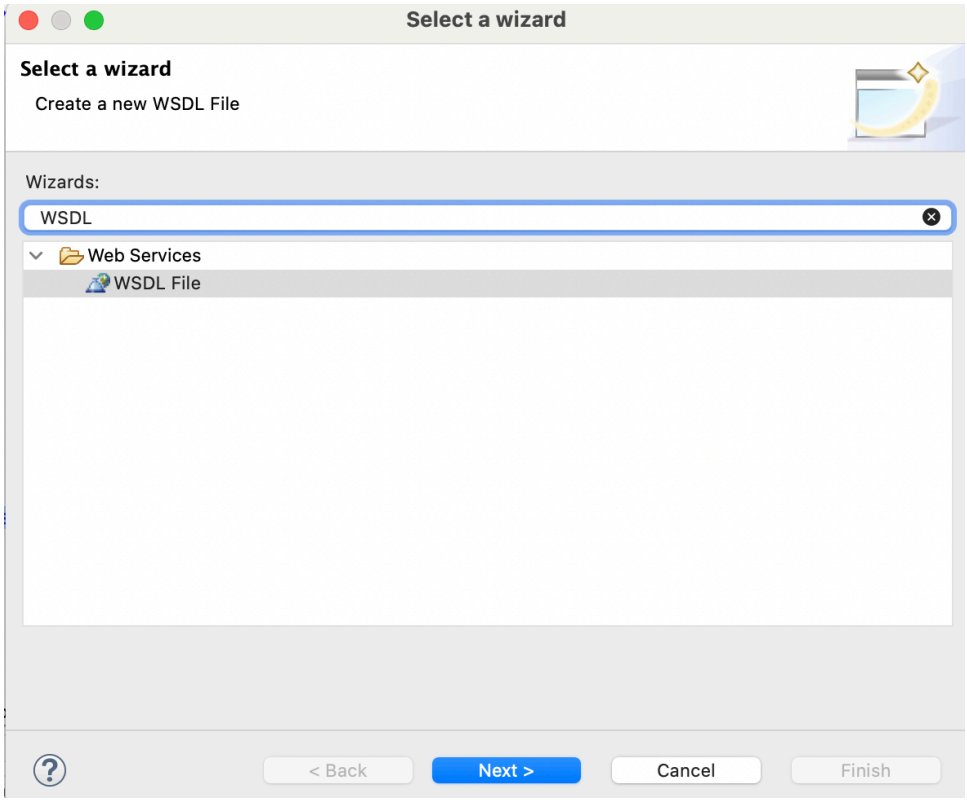
Click on **Finish**.

Creating the WSDL document:

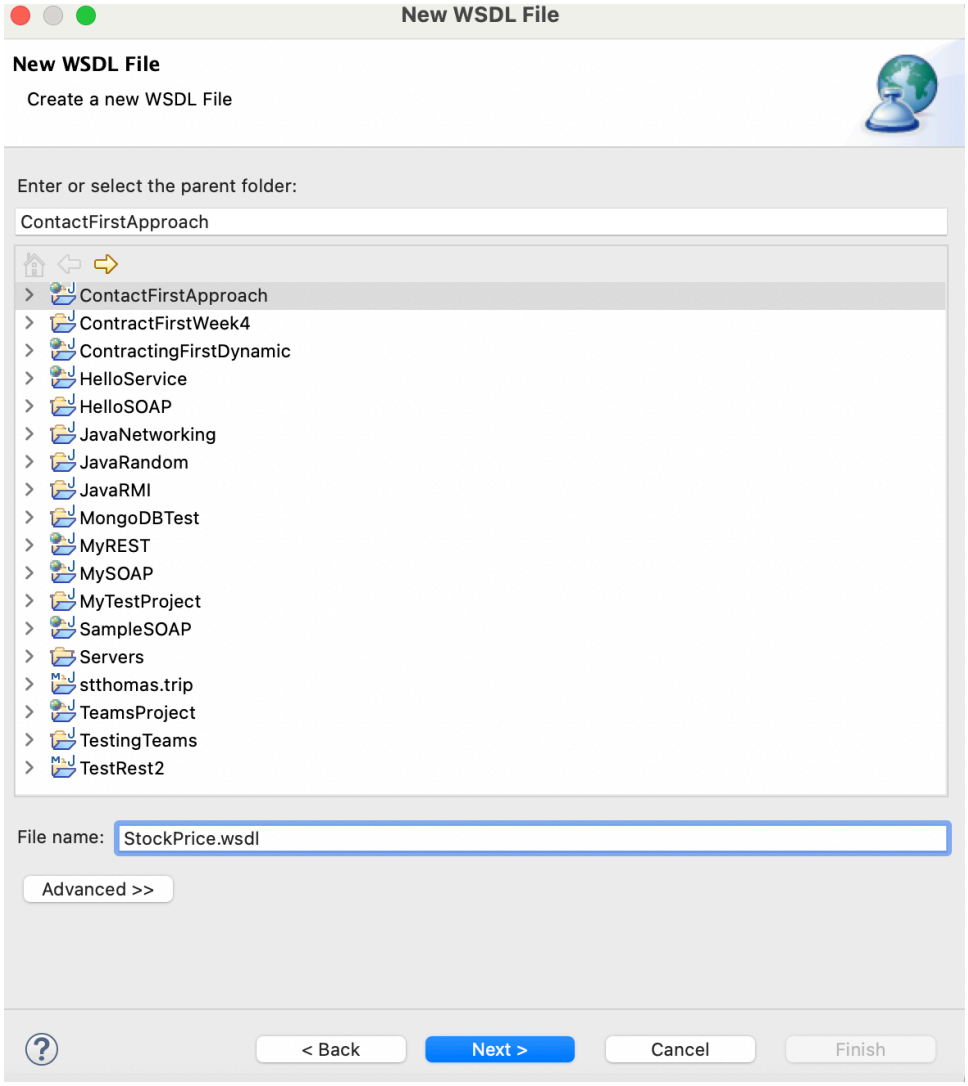
Right click on the Project and select **New -> Other**



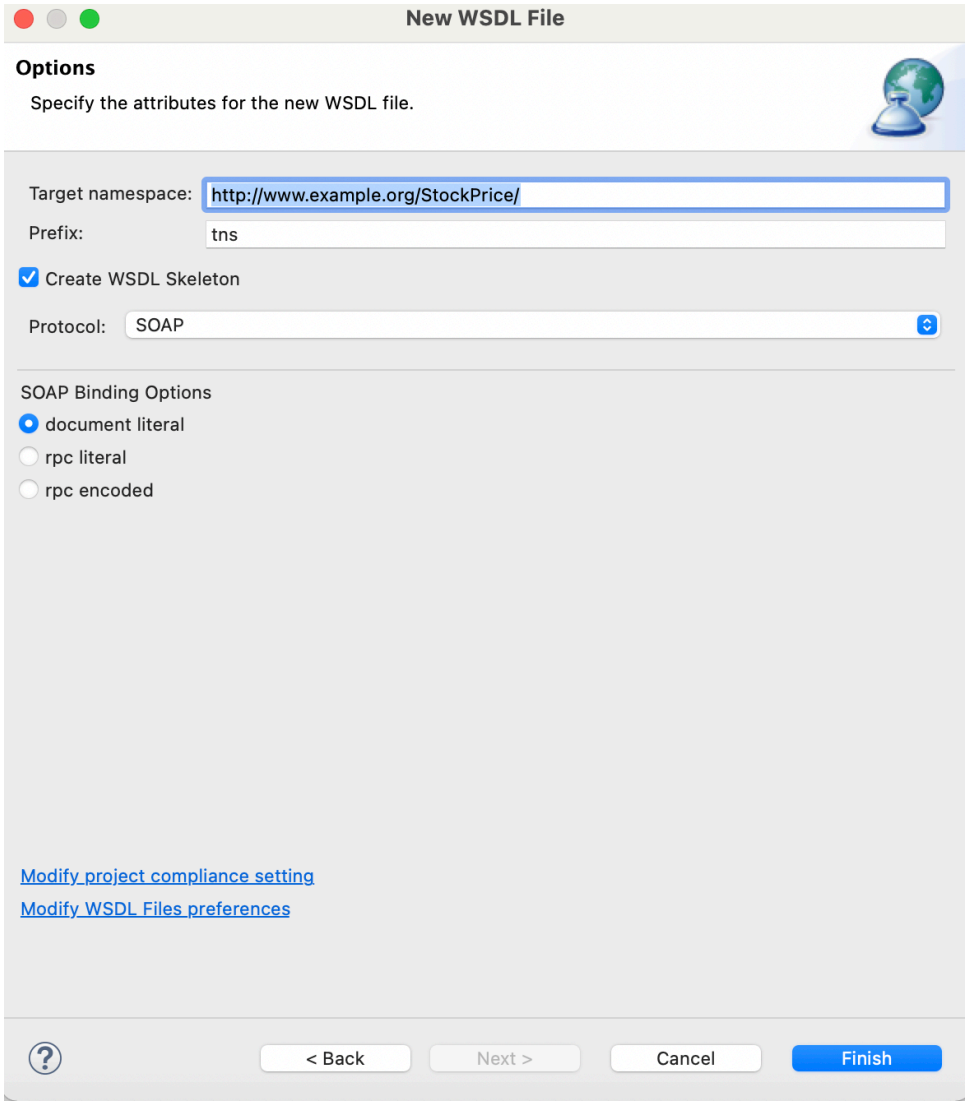
Search for **WSDL**.



Select the **WSDL File.**
Choose the correct project (ContractFirstApproach) and give a name to your WSDL file e.g. **StockPrice.wsdl**



You can choose to give your own namespace, if you want. Keep it as **document literal.** The protocol should be **SOAP** and click on **Finish**.



You will find a StockPrice.wsdl like this.

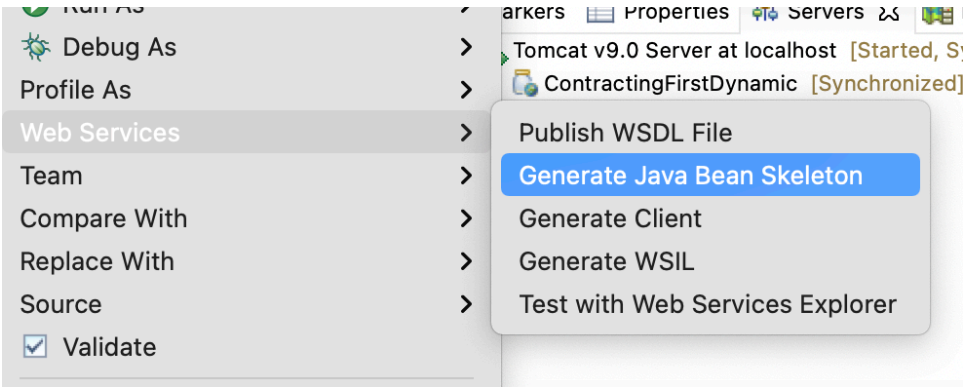
```
1  <?xml version="1.0" encoding="UTF-8" standalone="no"?>
2  <wsdl:definitions xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" xmlns:tns="http://www.exampi
3    <wsdl:types>
4      <xsd:schema targetNamespace="http://www.example.org/StockPrice/">
5        <xsd:element name="NewOperation">
6          <xsd:complexType>
7            <xsd:sequence>
8              <xsd:element name="in" type="xsd:string"/>
9            </xsd:sequence>
10         </xsd:complexType>
11       </xsd:element>
12       <xsd:element name="NewOperationResponse">
13         <xsd:complexType>
14           <xsd:sequence>
15             <xsd:element name="out" type="xsd:string"/>
16           </xsd:sequence>
17         </xsd:complexType>
18       </xsd:element>
19     </xsd:schema>
20   </wsdl:types>
21   <wsdl:message name="NewOperationRequest">
22     <wsdl:part element="tns:NewOperation" name="parameters"/>
23   </wsdl:message>
24   <wsdl:message name="NewOperationResponse">
25     <wsdl:part element="tns:NewOperationResponse" name="parameters"/>
26   </wsdl:message>
27   <wsdl:portType name="StockPrice">
28     <wsdl:operation name="NewOperation">
29       <wsdl:input message="tns:NewOperationRequest"/>
30       <wsdl:output message="tns:NewOperationResponse"/>
31     </wsdl:operation>
32   </wsdl:portType>
```

Replace all "**NewOperation**" keyword with "**StockTickerPrice**". You should get a WSDL file like this.

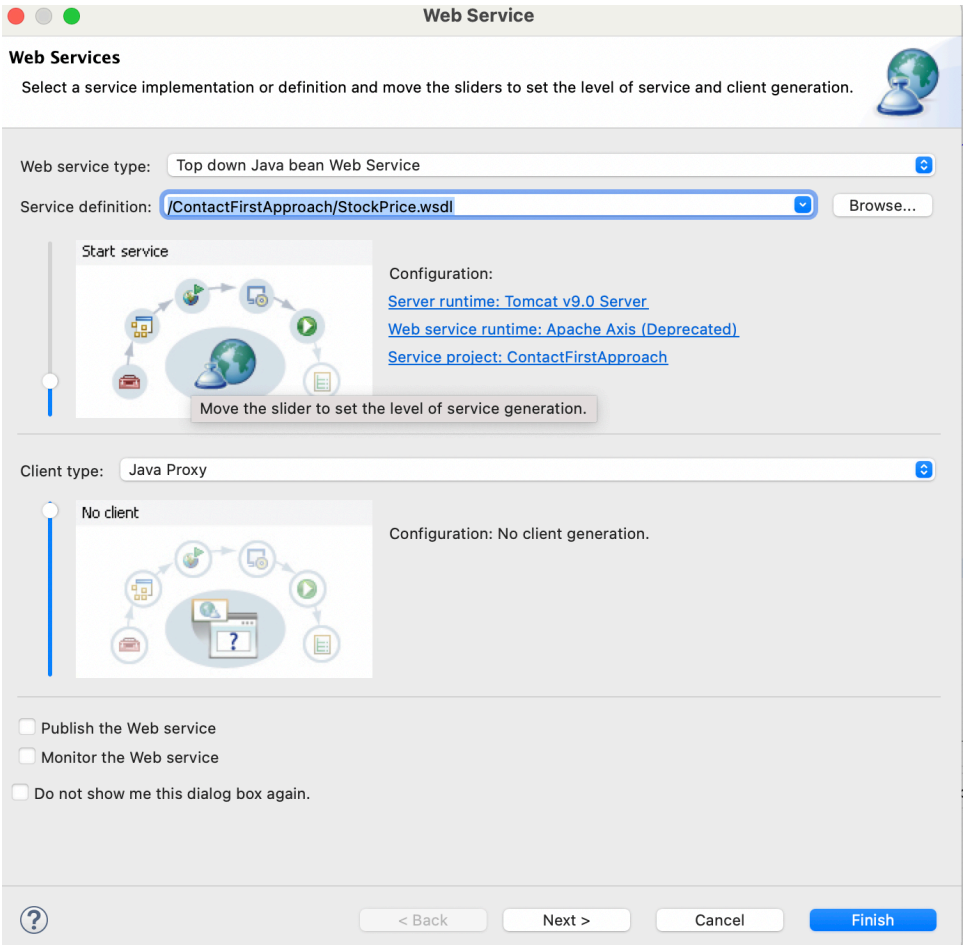
```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<wsdl:definitions xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:tns="http://www.example.org/StockPrice/"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" name="StockPrice"
targetNamespace="http://www.example.org/StockPrice/">
  <wsdl:types>
    <xsd:schema targetNamespace="http://www.example.org/StockPrice/">
      <xsd:element name="StockTickerPrice">
        <xsd:complexType>
          <xsd:sequence>
            <xsd:element name="in" type="xsd:string"/>
          </xsd:sequence>
        </xsd:complexType>
      </xsd:element>
      <xsd:element name="StockTickerPriceResponse">
        <xsd:complexType>
          <xsd:sequence>
            <xsd:element name="out" type="xsd:string"/>
          </xsd:sequence>
        </xsd:complexType>
      </xsd:element>
    </xsd:schema>
  </wsdl:types>
  <wsdl:message name="StockTickerPriceRequest">
    <wsdl:part element="tns:StockTickerPrice" name="parameters"/>
  </wsdl:message>
  <wsdl:message name="StockTickerPriceResponse">
    <wsdl:part element="tns:StockTickerPriceResponse" name="parameters"/>
  </wsdl:message>
  <wsdl:portType name="StockPrice">
    <wsdl:operation name="StockTickerPrice">
      <wsdl:input message="tns:StockTickerPriceRequest"/>
      <wsdl:output message="tns:StockTickerPriceResponse"/>
    </wsdl:operation>
  </wsdl:portType>
  <wsdl:binding name="StockPriceSOAP" type="tns:StockPrice">
    <soap:binding style="document"
transport="http://schemas.xmlsoap.org/soap/http"/>
    <wsdl:operation name="StockTickerPrice">
      <soap:operation
soapAction="http://www.example.org/StockPrice/StockTickerPrice"/>
      <wsdl:input>
        <soap:body use="literal"/>
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>
  </wsdl:binding>
  <wsdl:service name="StockPrice">
    <wsdl:port binding="tns:StockPriceSOAP" name="StockPriceSOAP">
      <soap:address location="http://www.example.org/"/>
    </wsdl:port>
  </wsdl:service>
</wsdl:definitions>
```

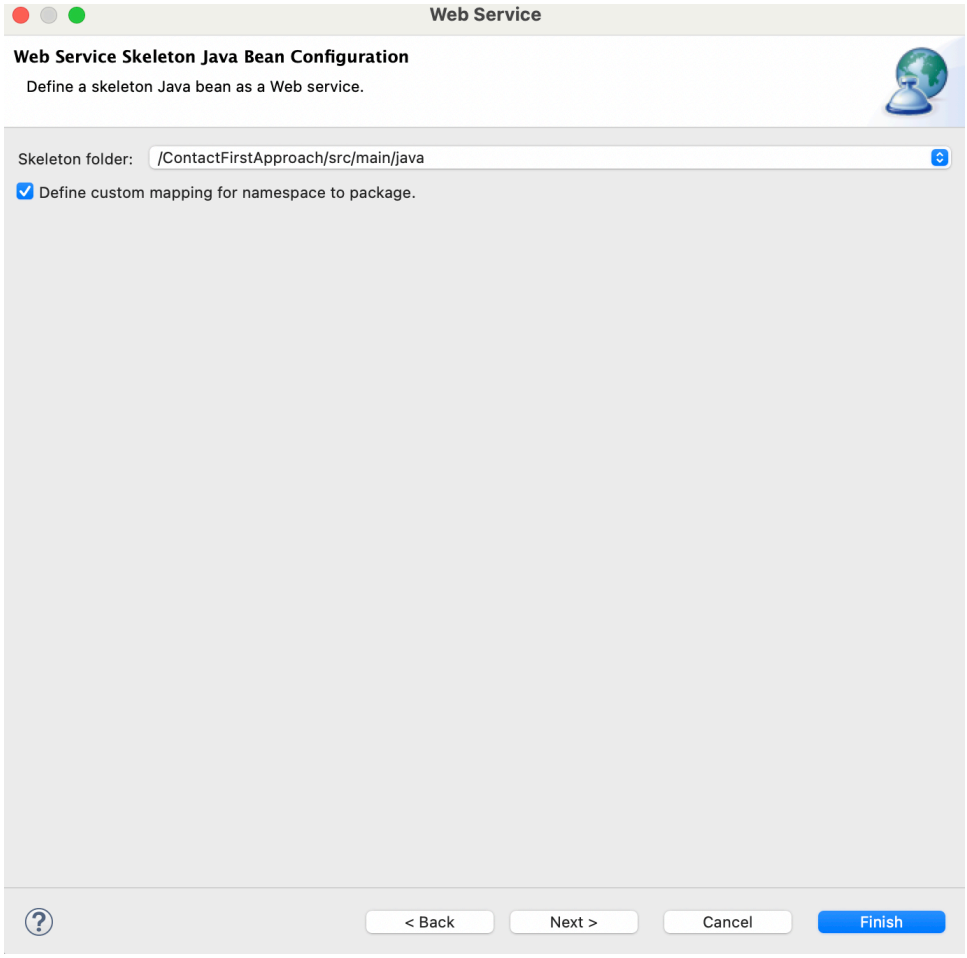
Generating the Service Classes:

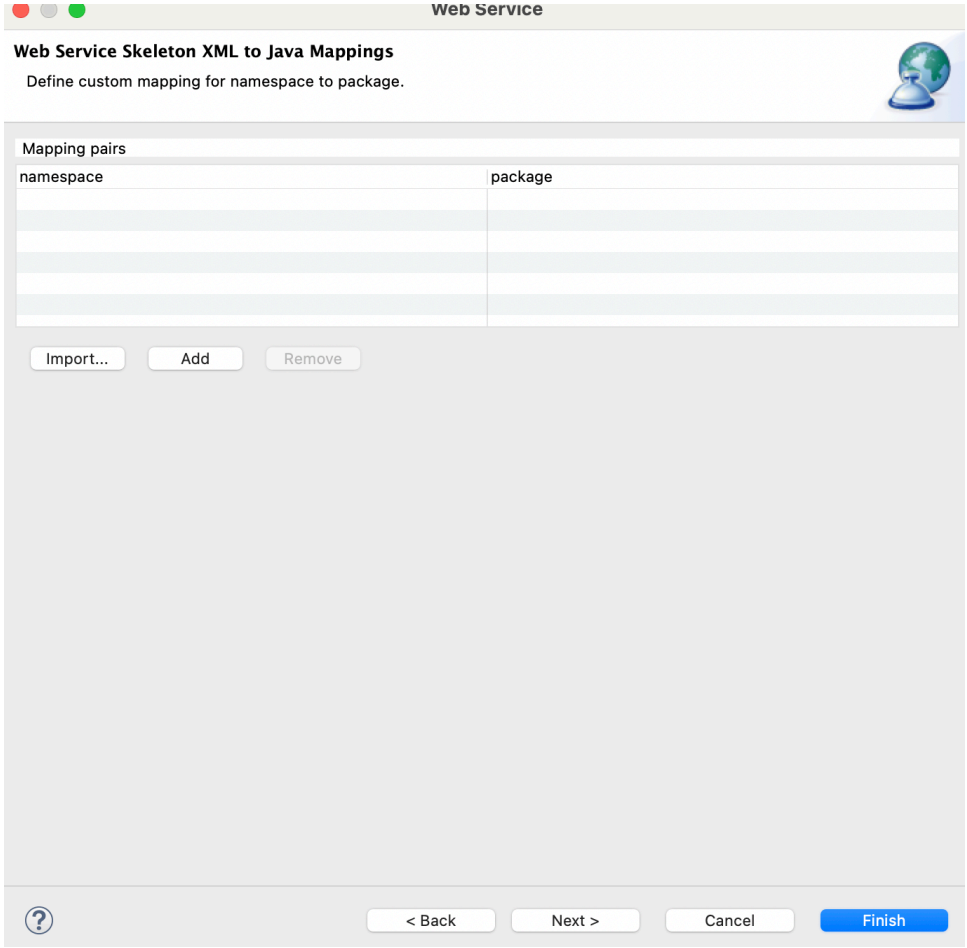
Right click on **StockPrice.wsdl** file and go to **Web Services -> Generate Java Bean Skeleton** option.



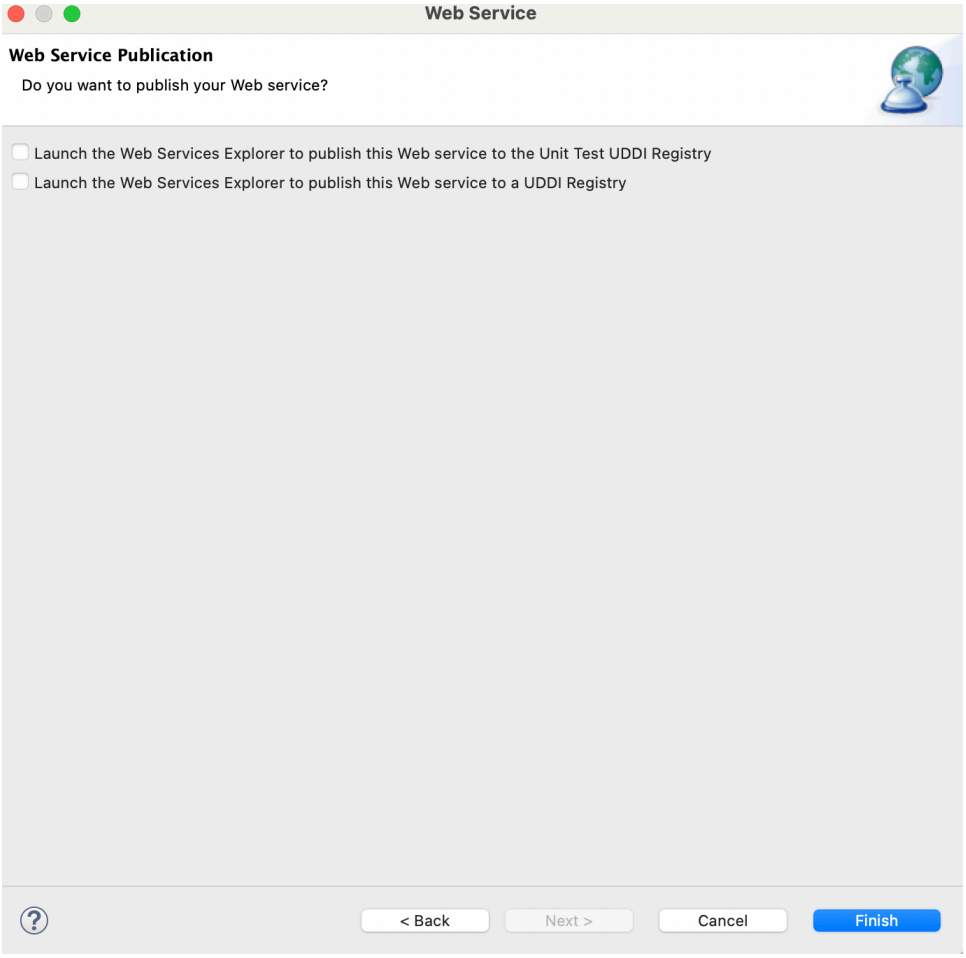
Use the defaults as shown below. We will be creating a Top Down Java Bean Web Service and we will let the IDE create and start the service for us. All we need to later is to implement the service. Press **Next**.




Web Service
Web Service Skeleton Java Bean Configuration
Define a skeleton Java bean as a Web service.
Skeleton folder:
/ContactFirstApproach/src/main/java
Define custom mapping for namespace to package.
< Back
Next >
Cancel
Finish



After this page, if the Tomcat Server is not up, it will ask you to **Start Server.** Once you do that you can click **Next** which will take you to the following page.



Press **Finish** (we are not publishing the service to UDDI).

Implementing the Service:

You will be provided with the editor for **StockPriceSOAPImpl.java**

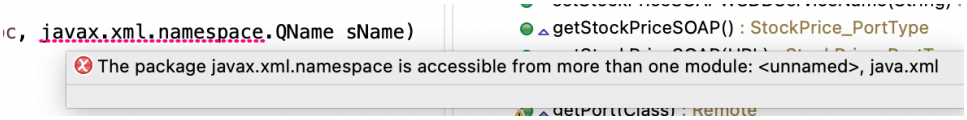
```
/**
 * StockPriceSOAPImpl.java
 *
 * This file was auto-generated from WSDL
 * by the Apache Axis 1.4 Apr 22, 2006 (06:55:48 PDT) WSDL2Java emitter.
 */

package org.example.www.StockPrice;

public class StockPriceSOAPImpl implements
org.example.www.StockPrice.StockPrice_PortType{
    public java.lang.String stockTickerPrice(java.lang.String in) throws
java.rmi.RemoteException {
        return null;
    }

}
```

Implement your code. Here is my sample implementation.

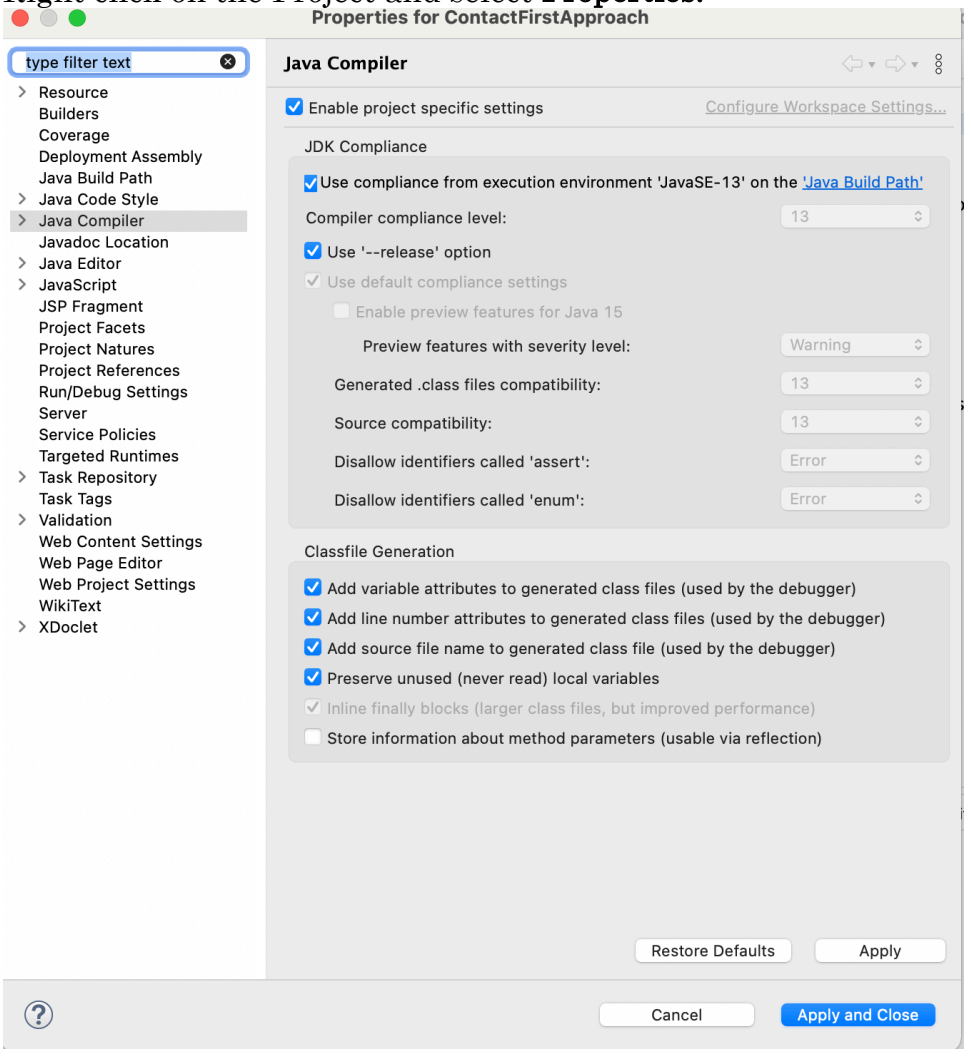
```
System.out.println("Input Parameters "+in);
String result = "";
    if(null!=in && !(in.isBlank()) && in.equalsIgnoreCase("ORCL"))
    {

    result="82.14";
    }

return result;
```

Namespace Error Handling (if any):

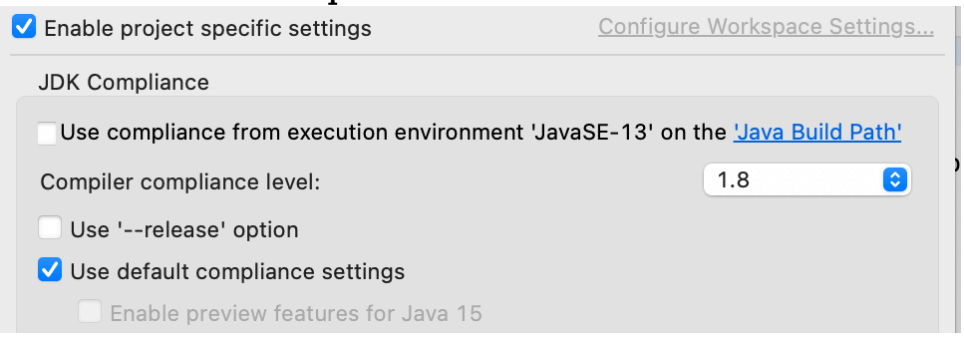
If you find the following error message in any of your Java Code, then this means that your project has been set to Java 11+.
We need to change it to Java 1.8.



Right click on the Project and select **Properties.**



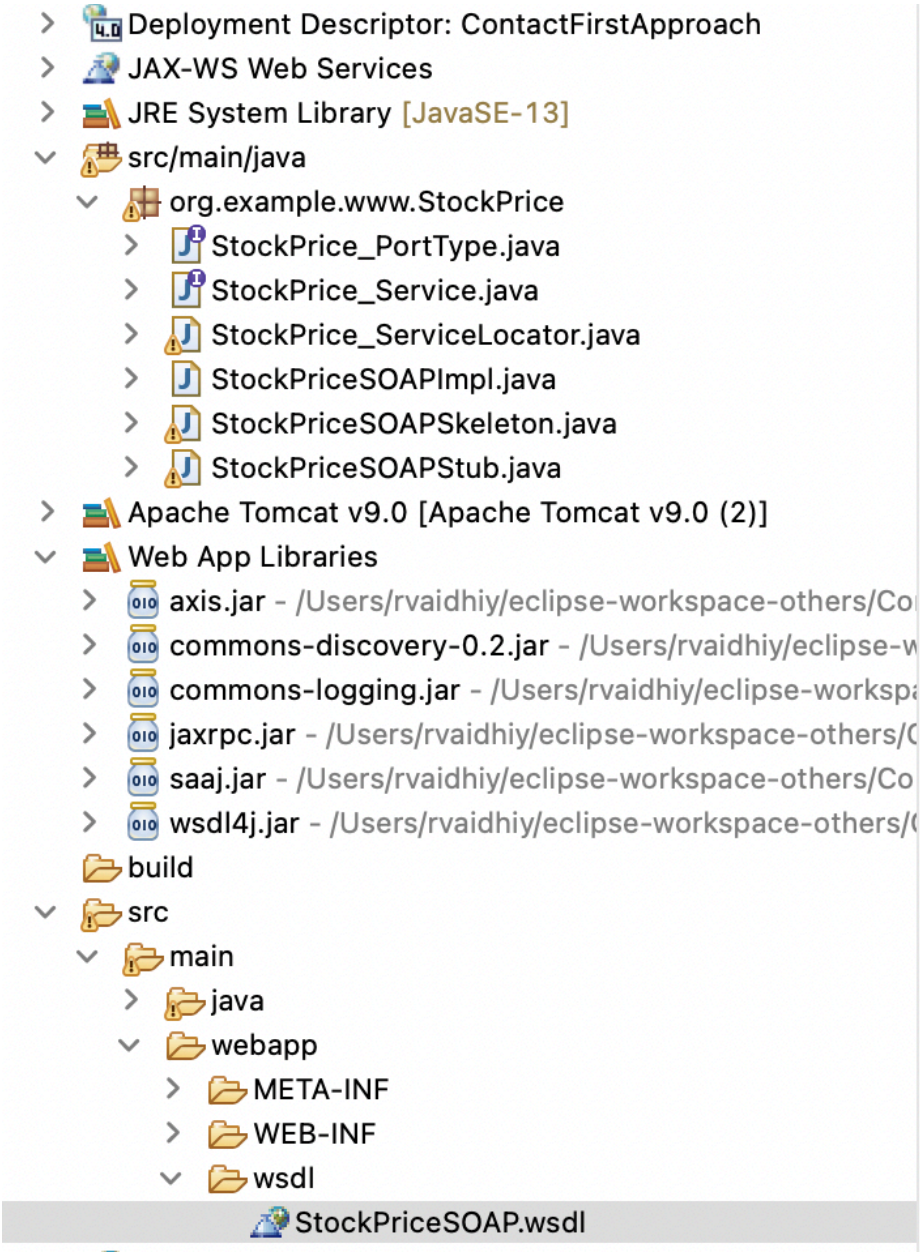
Uncheck the **Use Compliance** … and the **Use --release …**and set it to 1.8.



When you click **Apply and Close,** it will ask you if the code can be updated and compiled. Click Yes. You will notice that the errors are gone.

Testing the Service:

You can now right-click on the project and select **Run As->Run On Server.** You will find the updated WSDL in the **src->main->webapp->wsdl** folder.



Use this file to find your Endpoint address and the way to call your web service and test it using Postman.
The newly updated WSDL with all the endpoints updated looks like this.

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<wsdl:definitions xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:tns="http://www.example.org/StockPrice/"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" name="StockPrice"
targetNamespace="http://www.example.org/StockPrice/">

  <wsdl:types>

    <xsd:schema targetNamespace="http://www.example.org/StockPrice/">

      <xsd:element name="StockTickerPrice">

        <xsd:complexType>
```

```
          <xsd:sequence>
            <xsd:element name="in" type="xsd:string"/>
          </xsd:sequence>
        </xsd:complexType>
      </xsd:element>
      <xsd:element name="StockTickerPriceResponse">
        <xsd:complexType>
          <xsd:sequence>
            <xsd:element name="out" type="xsd:string"/>
          </xsd:sequence>
        </xsd:complexType>
      </xsd:element>
    </xsd:schema>
  </wsdl:types>
  <wsdl:message name="StockTickerPriceRequest">
    <wsdl:part element="tns:StockTickerPrice" name="parameters"/>
  </wsdl:message>
  <wsdl:message name="StockTickerPriceResponse">
    <wsdl:part element="tns:StockTickerPriceResponse" name="parameters"/>
  </wsdl:message>
  <wsdl:portType name="StockPrice">
    <wsdl:operation name="StockTickerPrice">
      <wsdl:input message="tns:StockTickerPriceRequest"/>
      <wsdl:output message="tns:StockTickerPriceResponse"/>
    </wsdl:operation>
  </wsdl:portType>
  <wsdl:binding name="StockPriceSOAP" type="tns:StockPrice">
    <soap:binding style="document"
transport="http://schemas.xmlsoap.org/soap/http"/>
    <wsdl:operation name="StockTickerPrice">
      <soap:operation
soapAction="http://www.example.org/StockPrice/StockTickerPrice"/>
      <wsdl:input>
        <soap:body use="literal"/>
```

```
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>
  </wsdl:binding>
  <wsdl:service name="StockPrice">
    <wsdl:port binding="tns:StockPriceSOAP" name="StockPriceSOAP">
      <soap:address
location="http://localhost:8080/ContactFirstApproach/services/StockPriceSOAP"/>
    </wsdl:port>
  </wsdl:service>
</wsdl:definitions>
```

My Postman request details are as follows:

URL for POST:
http://localhost:8080/ContactFirstApproach/services/StockPriceSOAP

Headers:
Content-Type: text/xml
soapAction:""

Body:

```
<soapenv:Envelope
      xmlns:mns="http://www.example.org/StockPrice/"
      xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
      <soapenv:Body>
                  <mns:StockTickerPrice>
                        <in>ORCL</in>
                        </mns:StockTickerPrice>
      </soapenv:Body>
</soapenv:Envelope>
```

Response Received:

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:x
sd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance">
    <soapenv:Body>
        <StockTickerPriceResponse xmlns="http://www.example.org/StockPrice/">
            <out xmlns="">82.14</out>
        </StockTickerPriceResponse>
    </soapenv:Body>
</soapenv:Envelope>
```

Question for you:

What is the key difference between the WSDL file you start off with (which you wrote manually specifying the operations) and the one you used finally (which is in a slightly different folder)? Why do you think the IDE had to modify the one you gave?