

Exercise 3a

Creating a WSDL-first service

Prior Knowledge

Understand WSDL and Schema

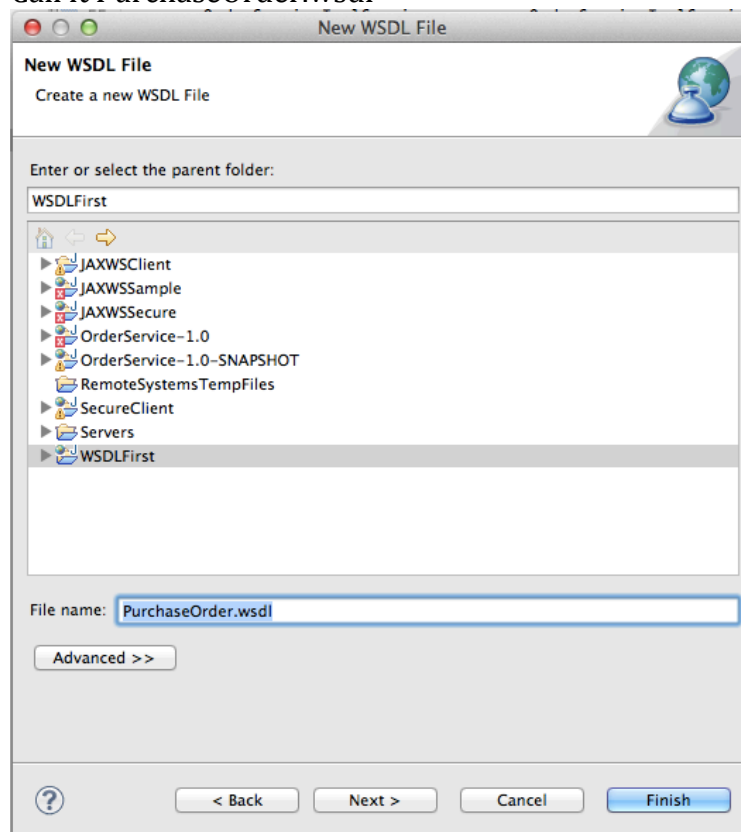
Objectives

Learn how to model services contract first and then implement.

Software Requirements

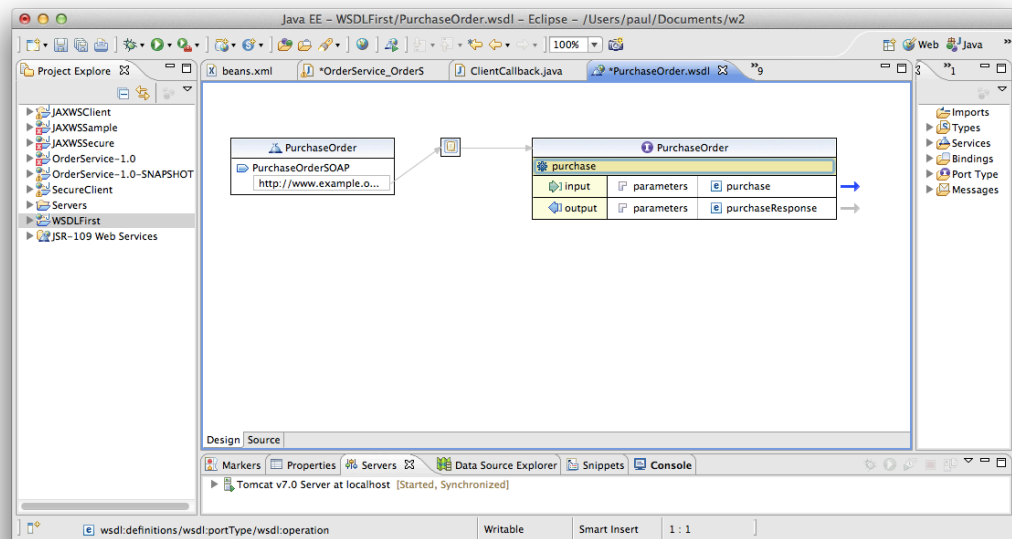
- Java Development Kit 7
- Eclipse JEE

1. In Eclipse create a new Dynamic Web Project
2. Create a WSDL file using File->New->Other->Web Services->WSDL
Call it PurchaseOrder.wsdl



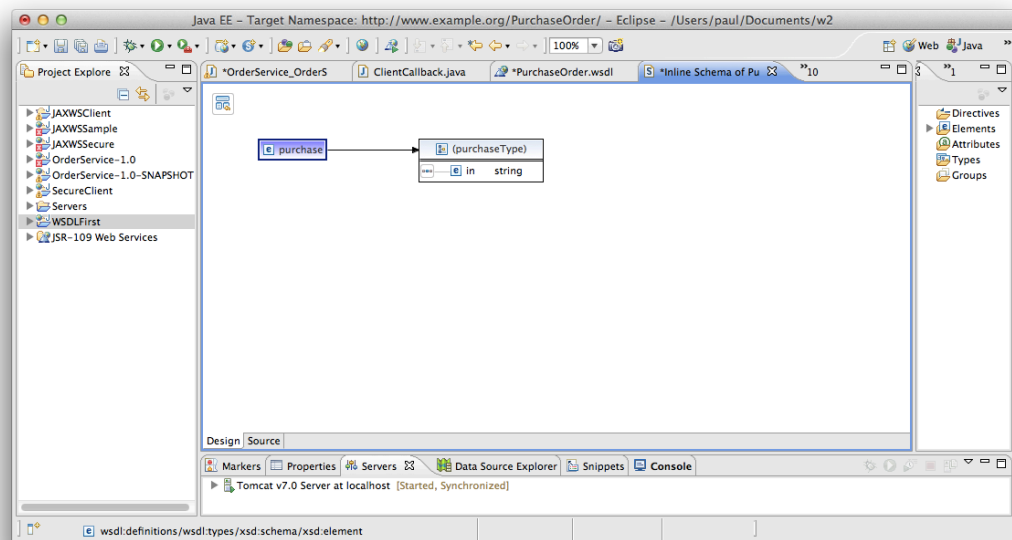


5. Now edit the Operation name to be purchase

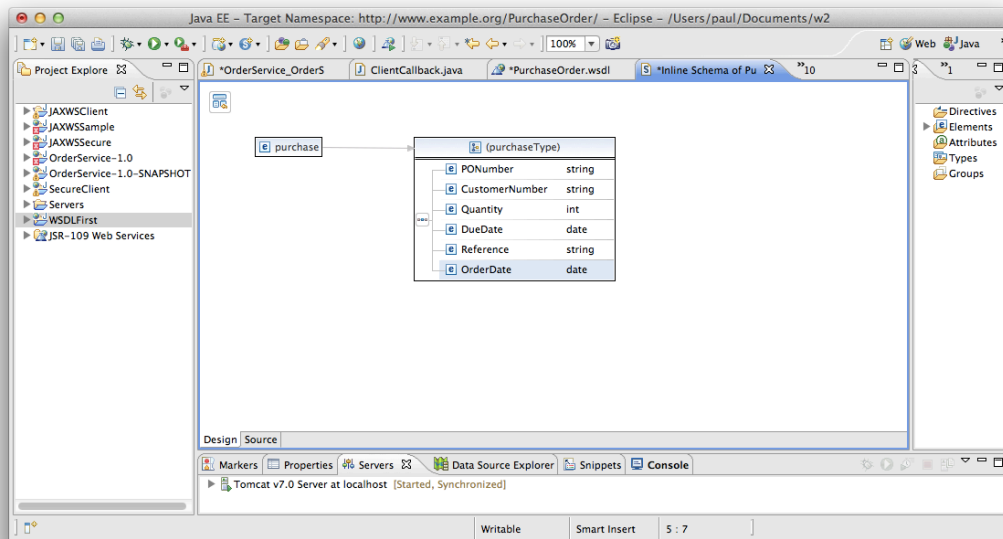


6. Now click on the → next to purchase

7. You will see some XML Schema

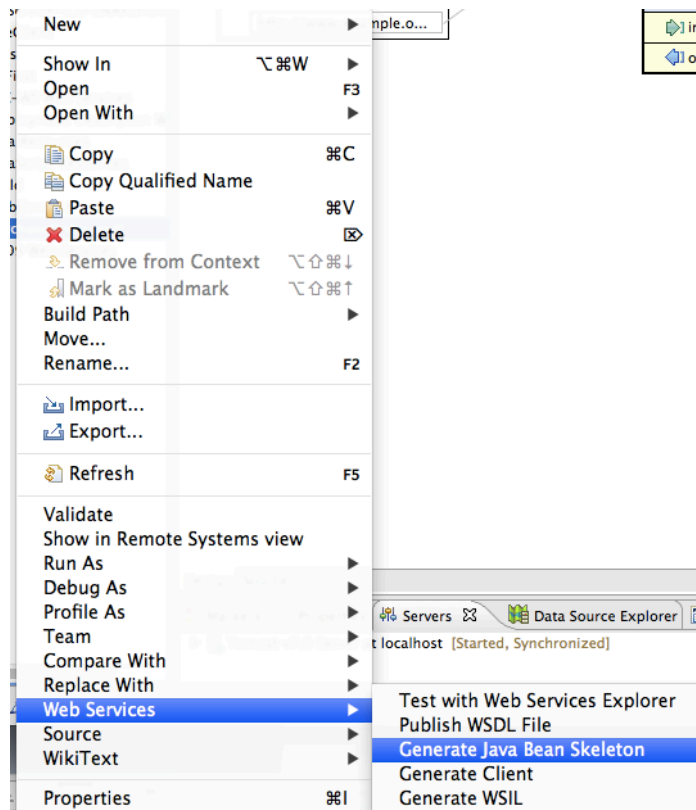


8. Replace the default single element that makes up the Purchase type with some useful elements. You should end up with something like:

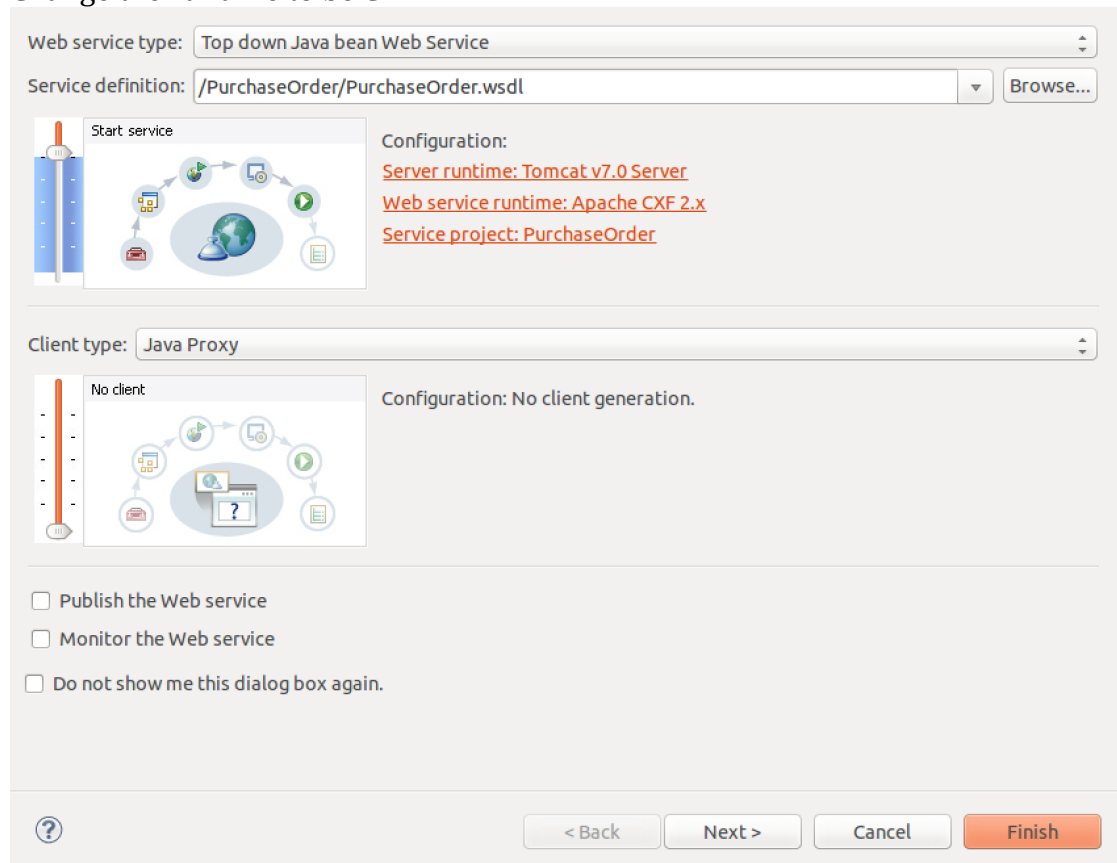


9. Save the schema (Ctrl-S)
10. Navigate back and do the same to make an intelligent response.
11. Right click on your WSDL file and select Validate. Check it all looks nice.
12. If you really want to, add some further operations, and schemas.
13. Now let's create some code. Right-click on your WSDL and select Web-Services->Generate Java Bean Skeleton.





14. Change the runtime to be CXF:



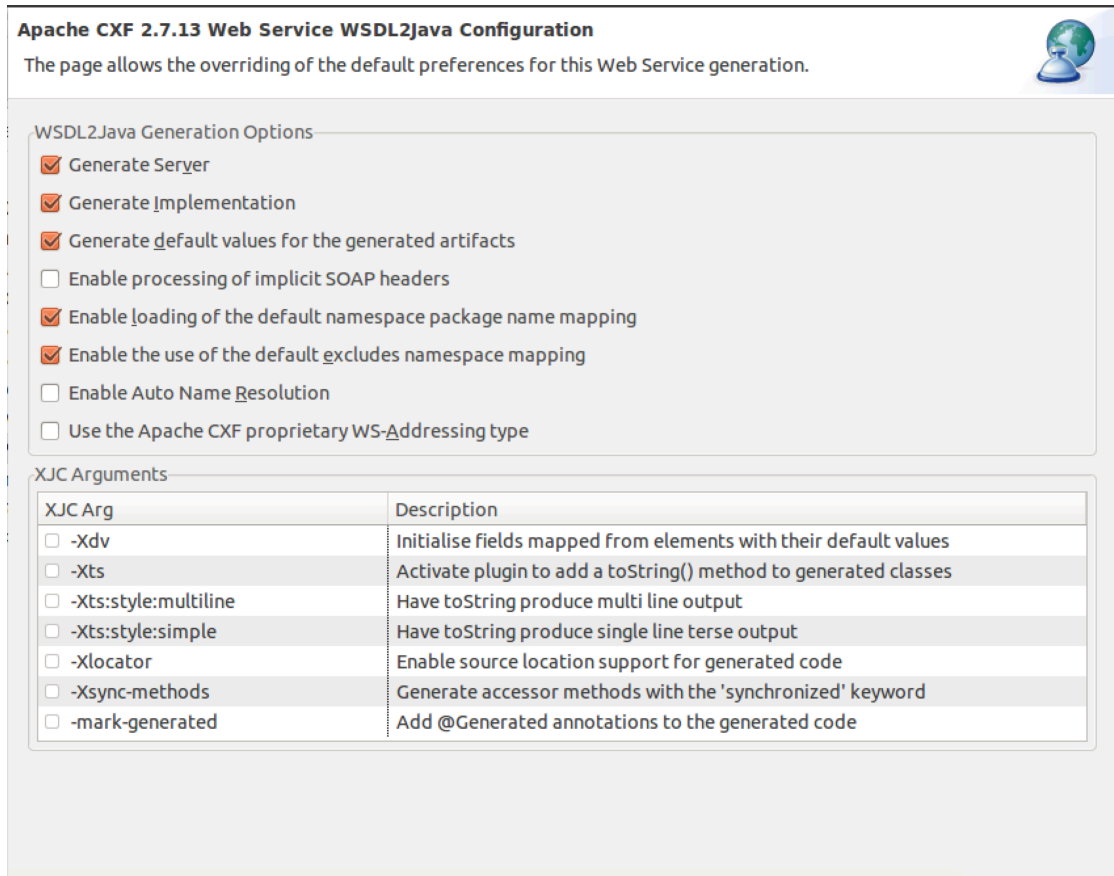
15. Click Next> and then make sure the Service Name is selected properly

The screenshot shows a configuration window with the following fields and controls:

- Output Directory:** A text box containing `/PurchaseOrder/src`.
- Package Name:** A text box containing `org.example.purchaseorder`.
- ☐ **Specify WSDL Namepsace to Package Name Mappings** (Note the typo 'Namepsace').
- Service Name:** A dropdown menu currently showing `PurchaseOrder`.
- Binding Files:** An empty text box with **Add...** and **Remove...** buttons to its right.
- Navigation:** At the bottom, there is a question mark icon, and buttons for **< Back**, **Next >**, **Cancel**, and **Finish** (highlighted in orange).



16. Click Next> and review the options. I suggest you check the same ones as shown below:



Apache CXF 2.7.13 Web Service WSDL2Java Configuration

The page allows the overriding of the default preferences for this Web Service generation.

WSDL2Java Generation Options

- ☒ Generate Server
- ☒ Generate Implementation
- ☒ Generate default values for the generated artifacts
- ☐ Enable processing of implicit SOAP headers
- ☒ Enable loading of the default namespace package name mapping
- ☒ Enable the use of the default excludes namespace mapping
- ☐ Enable Auto Name Resolution
- ☐ Use the Apache CXF proprietary WS-Addressing type

XJC Arguments

XJC Arg	Description
<input type="checkbox"/> -Xdv	Initialise fields mapped from elements with their default values
<input type="checkbox"/> -Xts	Activate plugin to add a toString() method to generated classes
<input type="checkbox"/> -Xts:style:multiline	Have toString produce multi line output
<input type="checkbox"/> -Xts:style:simple	Have toString produce single line terse output
<input type="checkbox"/> -Xlocator	Enable source location support for generated code
<input type="checkbox"/> -Xsync-methods	Generate accessor methods with the 'synchronized' keyword
<input type="checkbox"/> -mark-generated	Add @Generated annotations to the generated code

17. Click Next>
18. Don't do any UDDI stuff! Click Finish.
19. The server has generated a nice implementation for you.
20. To find the WSDL, you need to either use the WSDD stuff (which we will cover later!). Alternatively you need to put together the project name into a URL like this:
`http://localhost:8080/YourProjectNameGoesHere/services`
21. Replace/Improve the logic in PurchaseOrderImpl with your logic. Run the service and test it with SOAPUI.
22. Extension: Find a WSDL on the internet and see if you can implement a service that shares the same WSDL.



