ΕΛΛΗΝΙΚΗ ΔΗΜΟΚΡΑΤΙΑ
ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΡΗΤΗΣ

Εισαγωγή στα Δίκτυα Υπηρεσιών

Assisting Lecture 9b - Top Down SOAP Web Services and Php Clients)

Μύρων Παπαδάκης
Τμήμα Επιστήμης Υπολογιστών

Introduction to Service Networks
CS-592 – Spring 2015

Assisting Lecture 12: Top- Down WS with Apache Tomcat, Axis2, Eclipse – Part II

Myron Papadakis

Outline

- Previous Lab: Tools for Top-Down Web Services and AreaService Example with clients

- Top-Down Web Services (cont'd)
  - Math Web Service
  - Loan Web Service

- Web Services Clients: Soap and Php

Eclipse Axis2 Lab Example

Lab Example Description

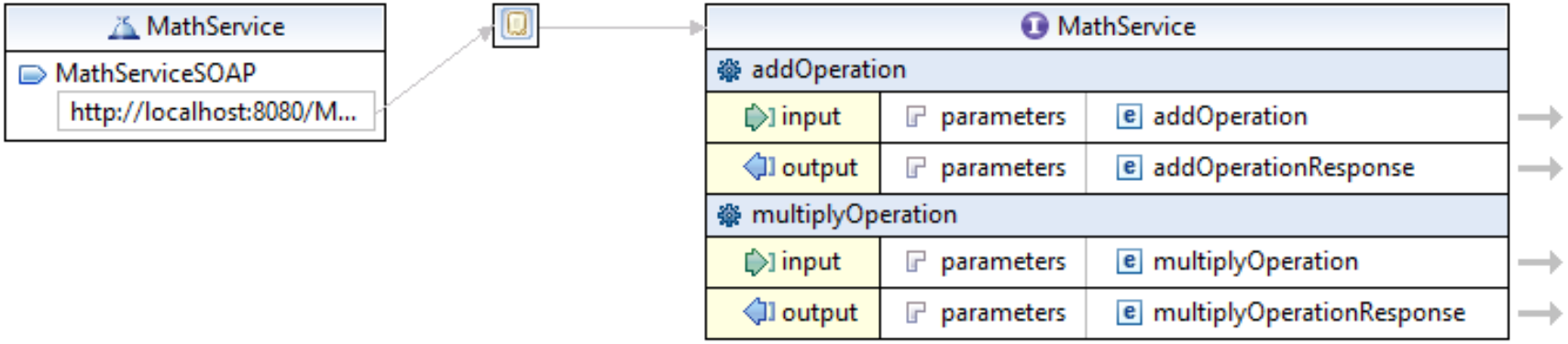
- Create a web service:
  - takes as input two integers
  - Returns the sum of the two integers
  - Returns the multiplication of the integers
- The Web Service will also have to be developed in Eclipse (as a Dynamic Web Project) using the top-down approach.

Lab Assignment

- Step 1: Interface of the Web Service
  - Create the WSDL file for the Math Service
  - Add 2 operations in the WSDL
    - Add: operation that adds two integers
    - Multiply: operation that multiplies two integers
- Step 2: Implementation of the service operations
  - Use Axis2 and create a top-down web service from the WSDL
  - Implement the operations defined in the WSDL file in Java
  - Test the Web service…

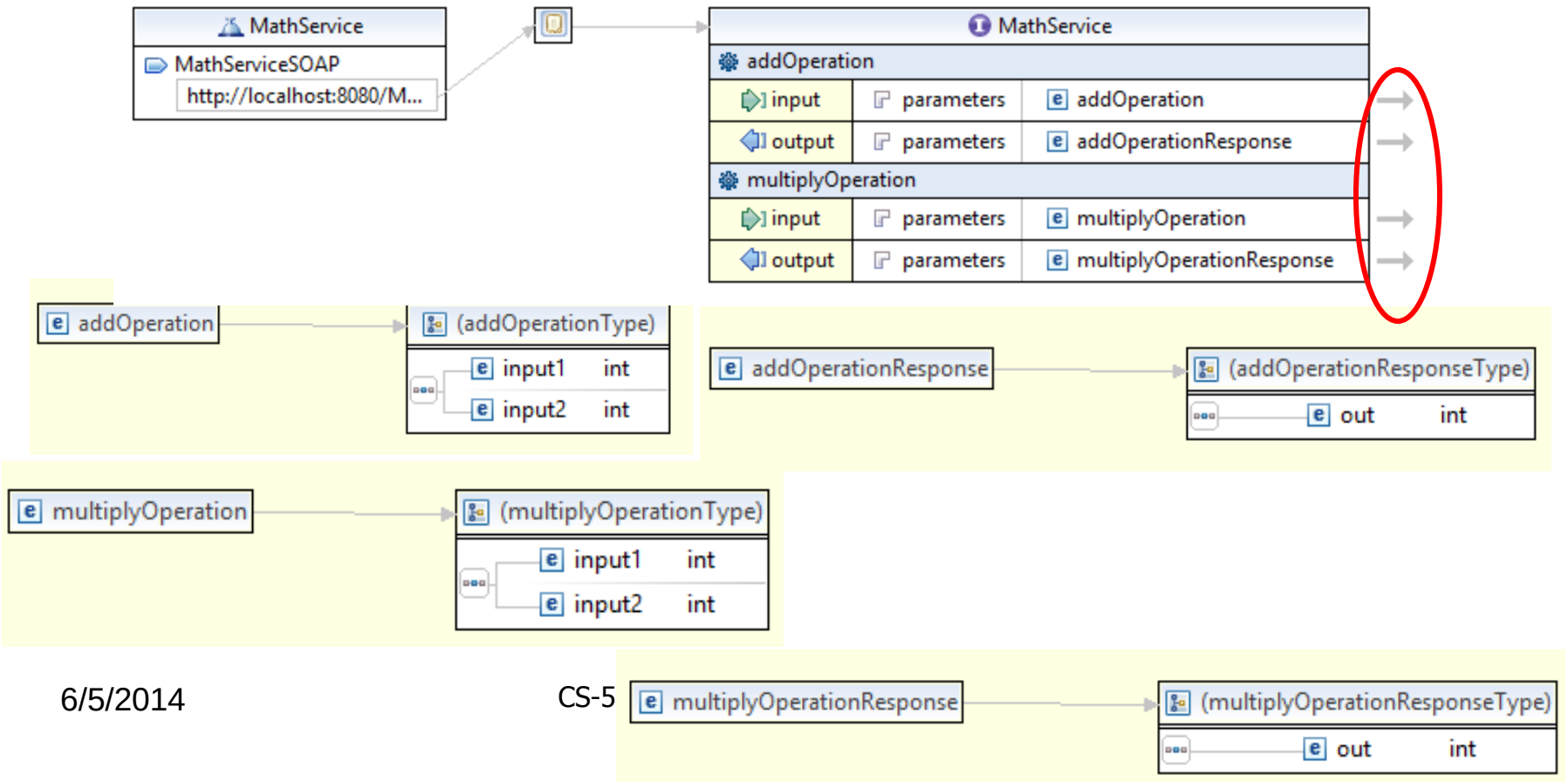
Developing the Math Web Service

- Create a new Dynamic Web Project called “Math”
- Create a folder WSDL on the Math project.
- Right click on this folder and then New→ Other → Web Services →WSDL and name this file MathService.wsdl.
- Add the operations to the WSDL File

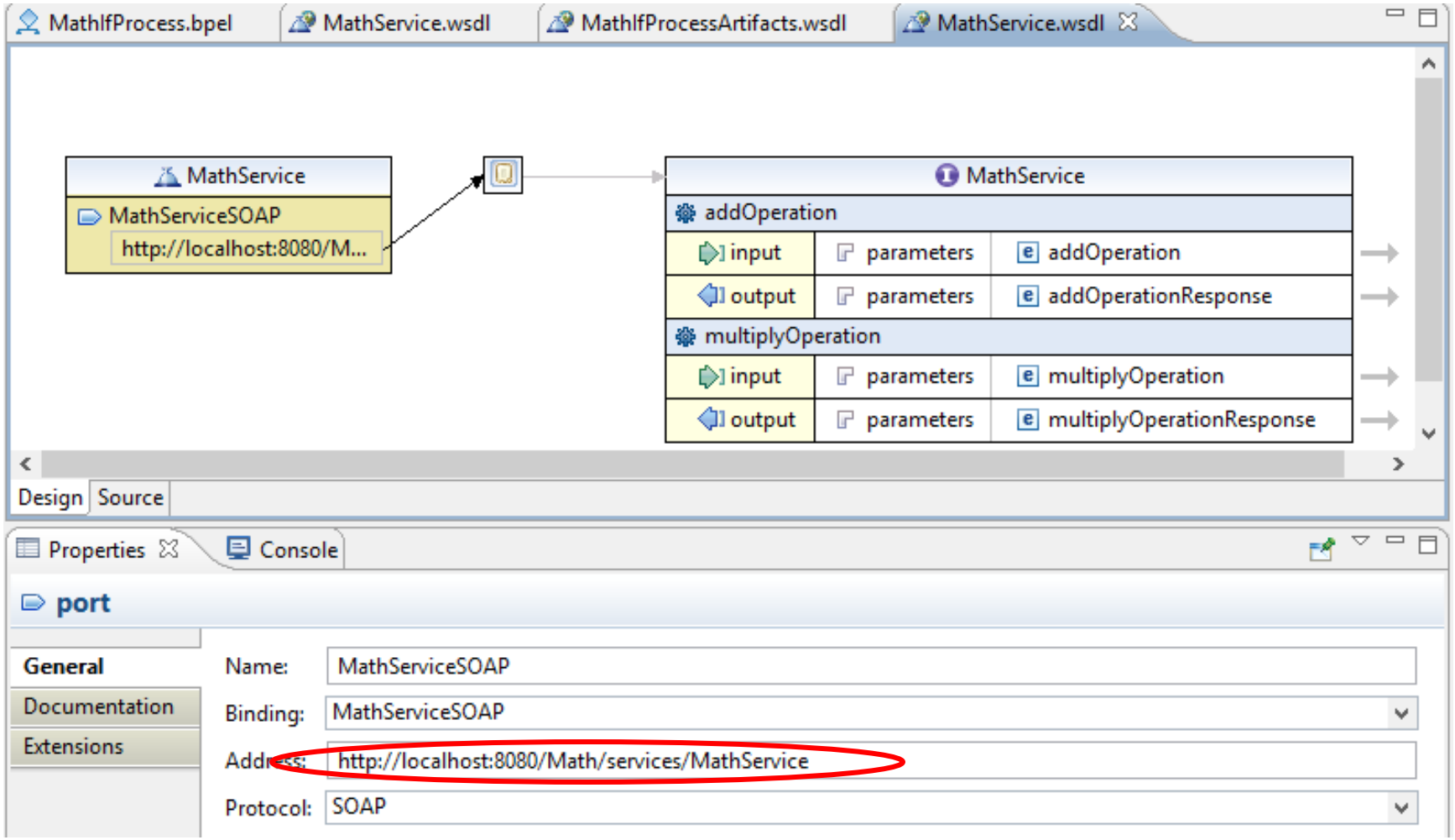


Developing the Math Web Service

- Both of these operations take as input 2 integers and return an integer as a results.



Developing the Math Web Service



Implementing the operations

- Implement the multiply operation

MathIfProcess.bpel | MathIfProcessArtifac | MathService.wsdl | MathServiceSkeleton.

```
 */
public class MathServiceSkeleton{


    /**
     * Auto generated method signature
     *
     * @param multiplyOperation
     * @return multiplyOperationResponse
     */

    public org.example.www.mathservice.MultiplyOperationResponse multiplyOperation
    (
            org.example.www.mathservice.MultiplyOperation multiplyOperation
    )
    {
        MultiplyOperationResponse r = new MultiplyOperationResponse();
        r.setOut(multiplyOperation.getInput1()*multiplyOperation.getInput2());
        return r;
    }
```

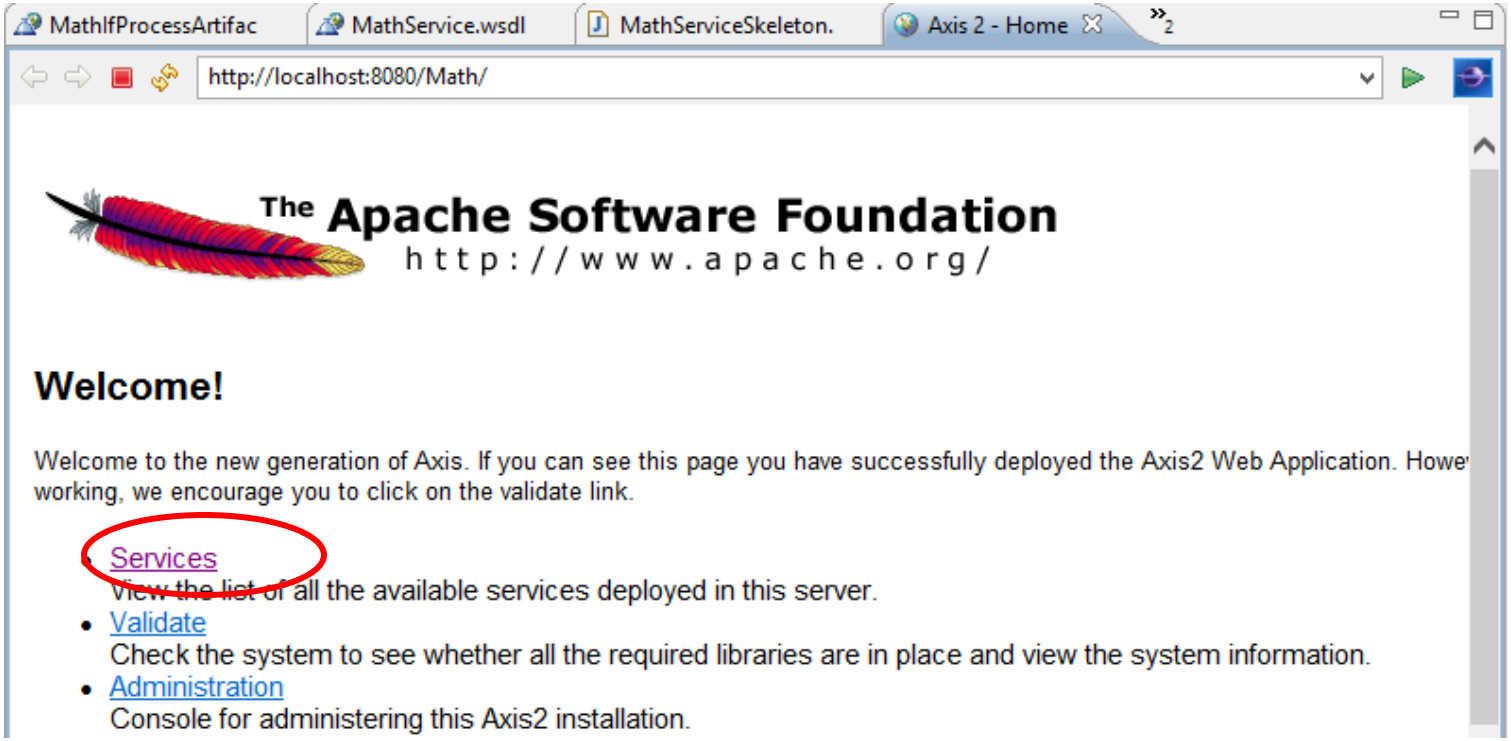
Implementing the operations

- Similarly implement the add operation in the MathServiceSkeleton.java

```
public org.example.www.mathservice.AddOperationResponse addOperation
(
        org.example.www.mathservice.AddOperation addOperation
)
{
    AddOperationResponse r = new AddOperationResponse();
    r.setOut(addOperation.getInput1()+addOperation.getInput2());
    return r;
}
```

Running the Math Project

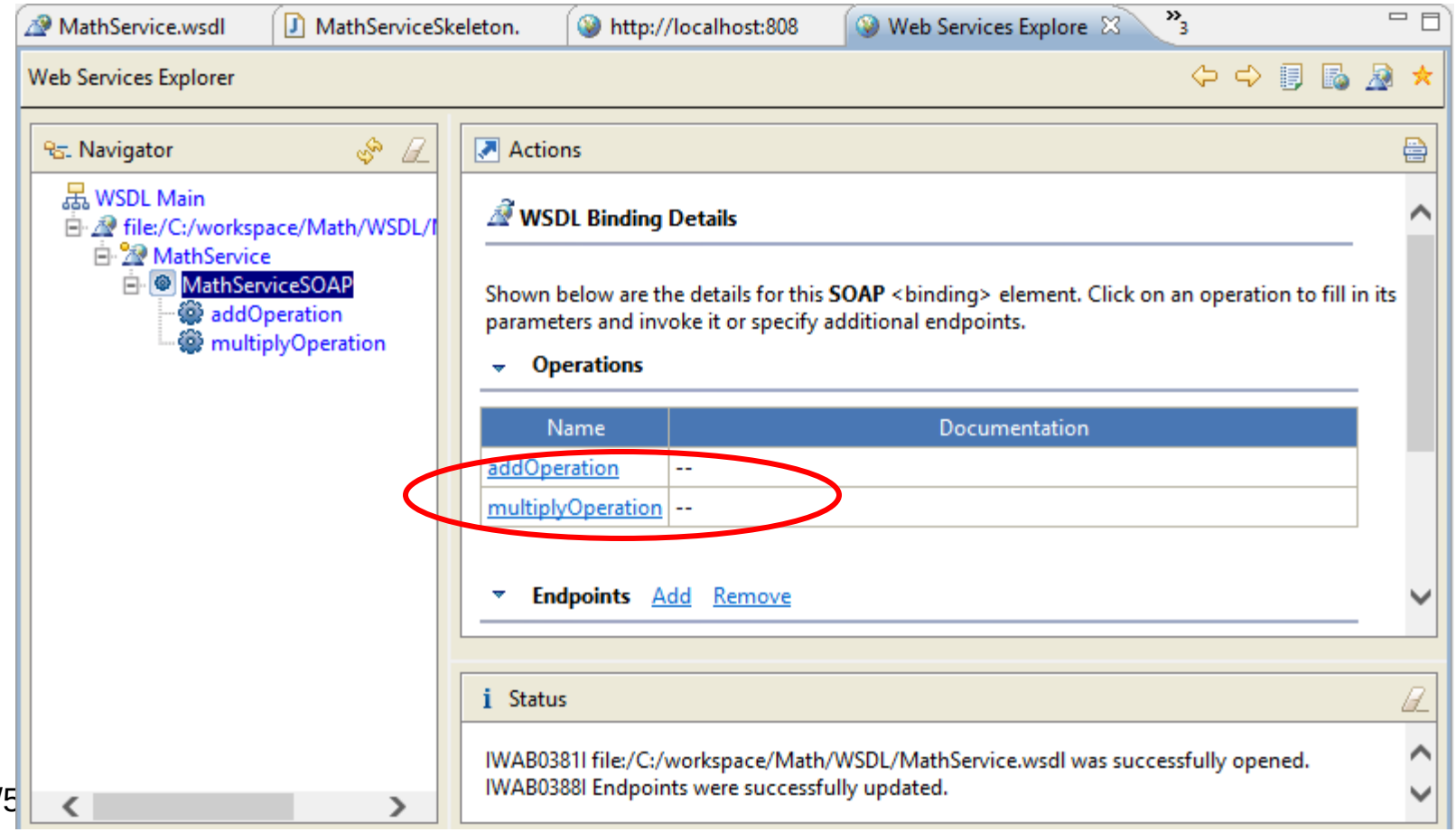
- Right Click the Project and select Run As → Run on Server
- Restart the server if prompted to do so



Running the Math Project

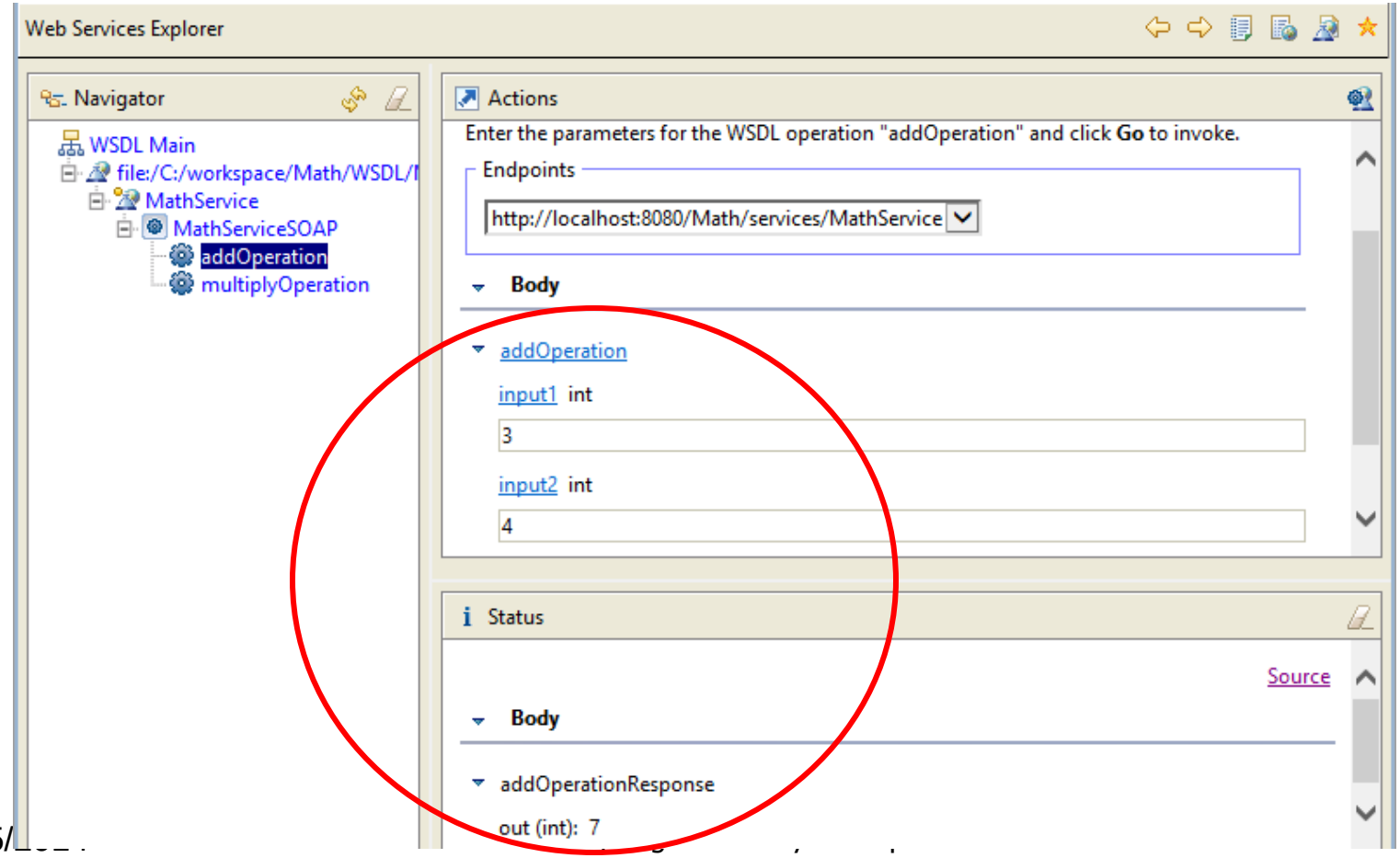
Running the Math Project > Test Web Service

- Check if service is running



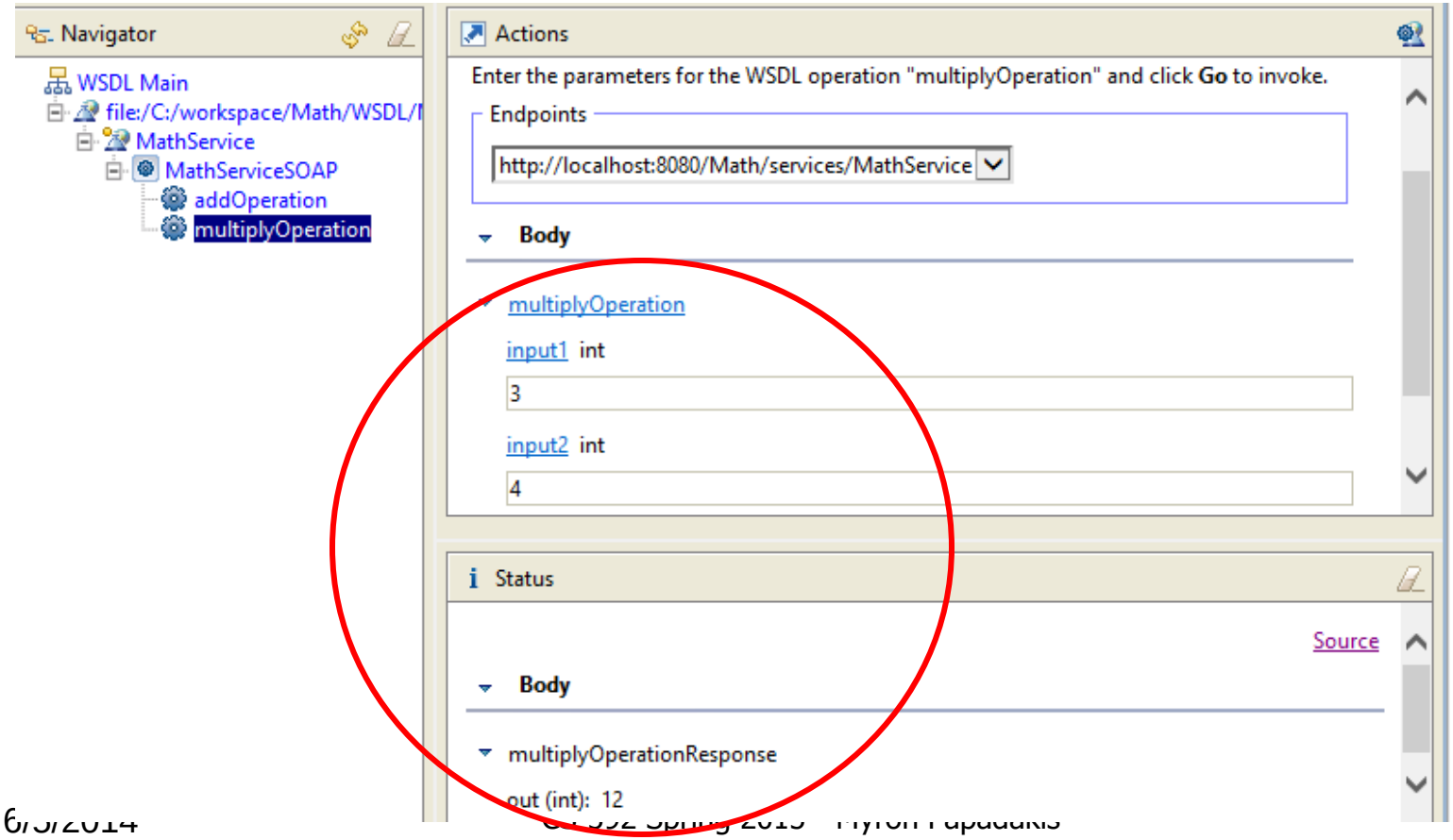
Running the Math Project > Test Web Service Operations

- Test the addOperation



Running the Math Project > Test Web Service Operations

- Test the multiplyOperation



More Axis2 Web Services – Example 3

Loan Example > Scenario

- Suppose that a client wants to request a loan from a bank
- The client sends the loan request and receives a response afterwards whether the loan was approved or rejected.
- The client sends the following details when asking for a loan
  - Name
  - Address
  - Age
  - Annual Salary
  - Amount Requested
- The bank returns a response to the client. If the loan is rejected for some reason the bank also states the reason of the rejection

Loan Example > Scenario

- The bank examines the loan request and either approves the loan or not.
- **Age**: If the applicant is under 18 years old or above 65 years old the application is rejected. The response contains a message that informs the applicant about the reason
- **Annual Salary**: If the annual salary is less than 20000 then the application is rejected
- **Amount Requested**: If the amount that the client requests is greater than the amount of loan based on years to pay off loan then the loan is not appoved.
  - limit = annualSalary * yearsToRepay * 0.5
  - yearsToRepay = AVERAGE_LIFE_EXPECTANCY - applicantAge;
- In any other case the application is approved.

Steps to create the WS

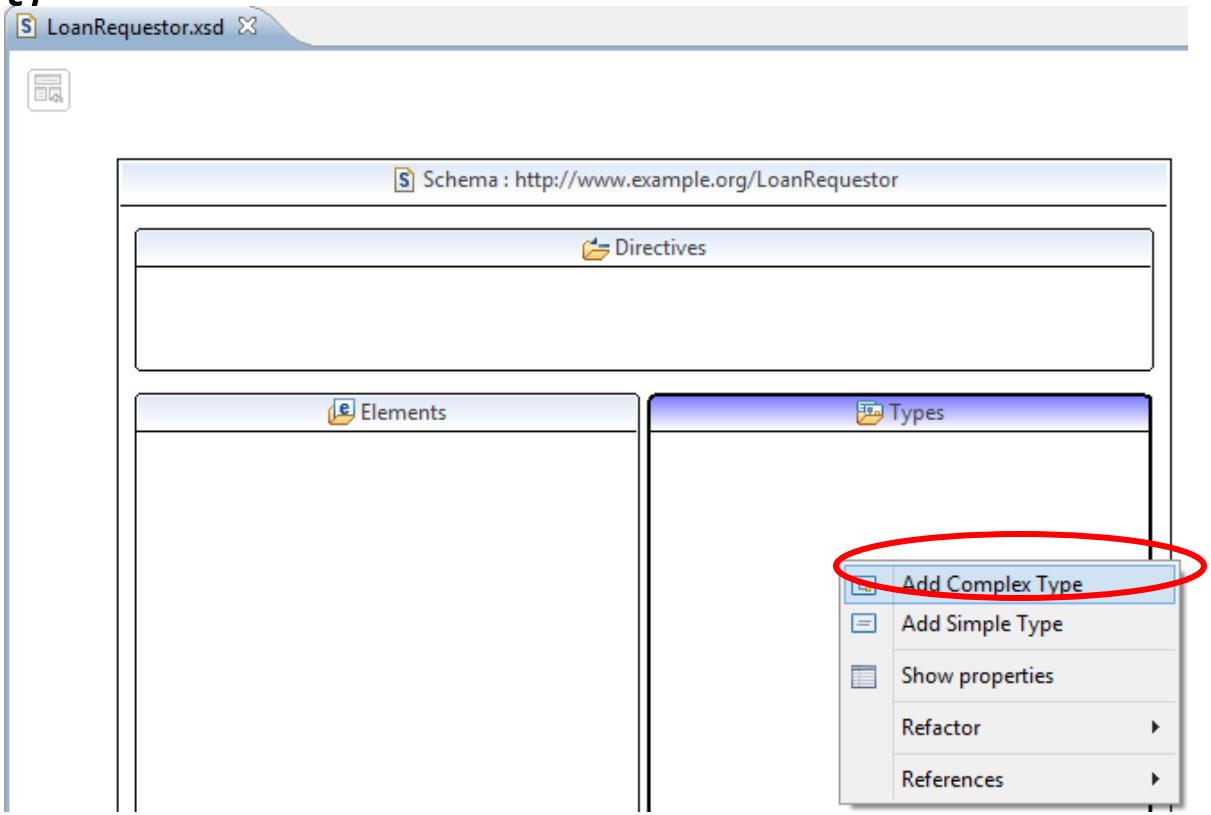
- Create the XSD Schema (loan request and loan response)
- Create the WSDL Document for the Web Service and provide the operations of the Web Service
  - Set the types of the request and the responses of the Web Service operations according to the XML Schema
- Create the Web Service from the WSDL using Axis2 (top-down approach)
- Implement the Web Service operations

Eclipse Dynamic Web Project

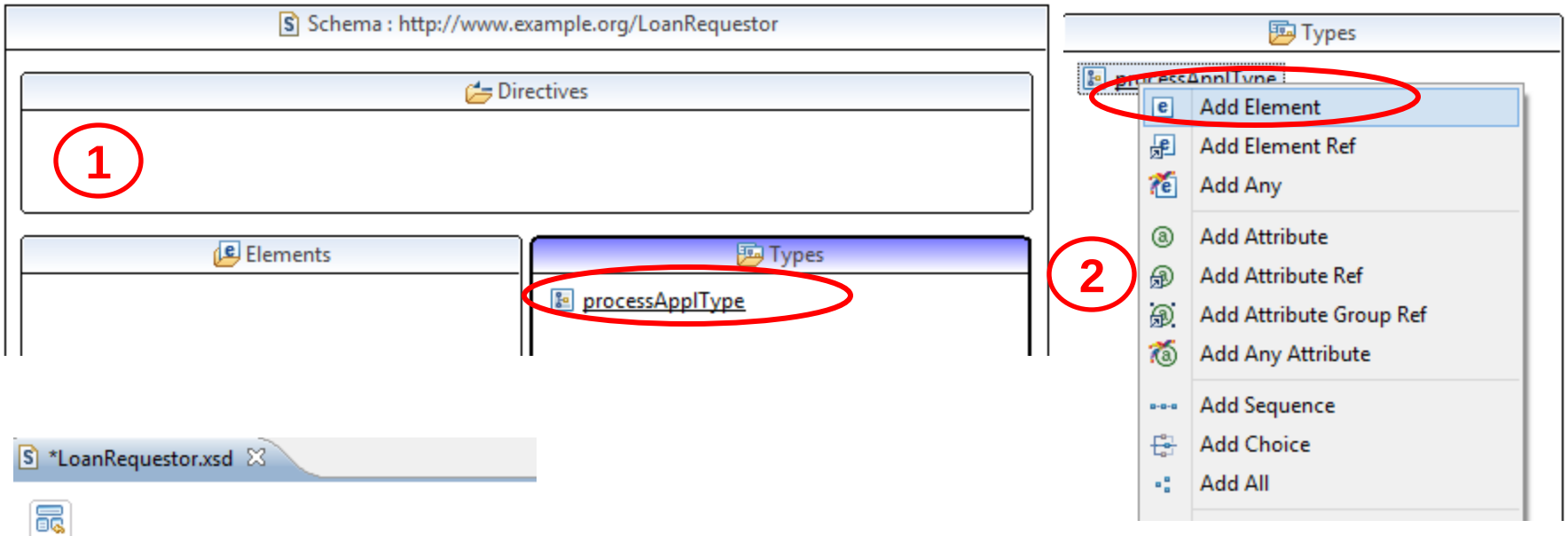
- Create a new Dynamic Web Project in Eclipse
- Create a folder “WSDL” in the project
  - In this folder we will place the XML schema and the WSDL document itself.
- Create a new XSD document (LoanRequestor.xsd)
  - File→New→Other→XML→XML Schema

XSD Schema (1/6)

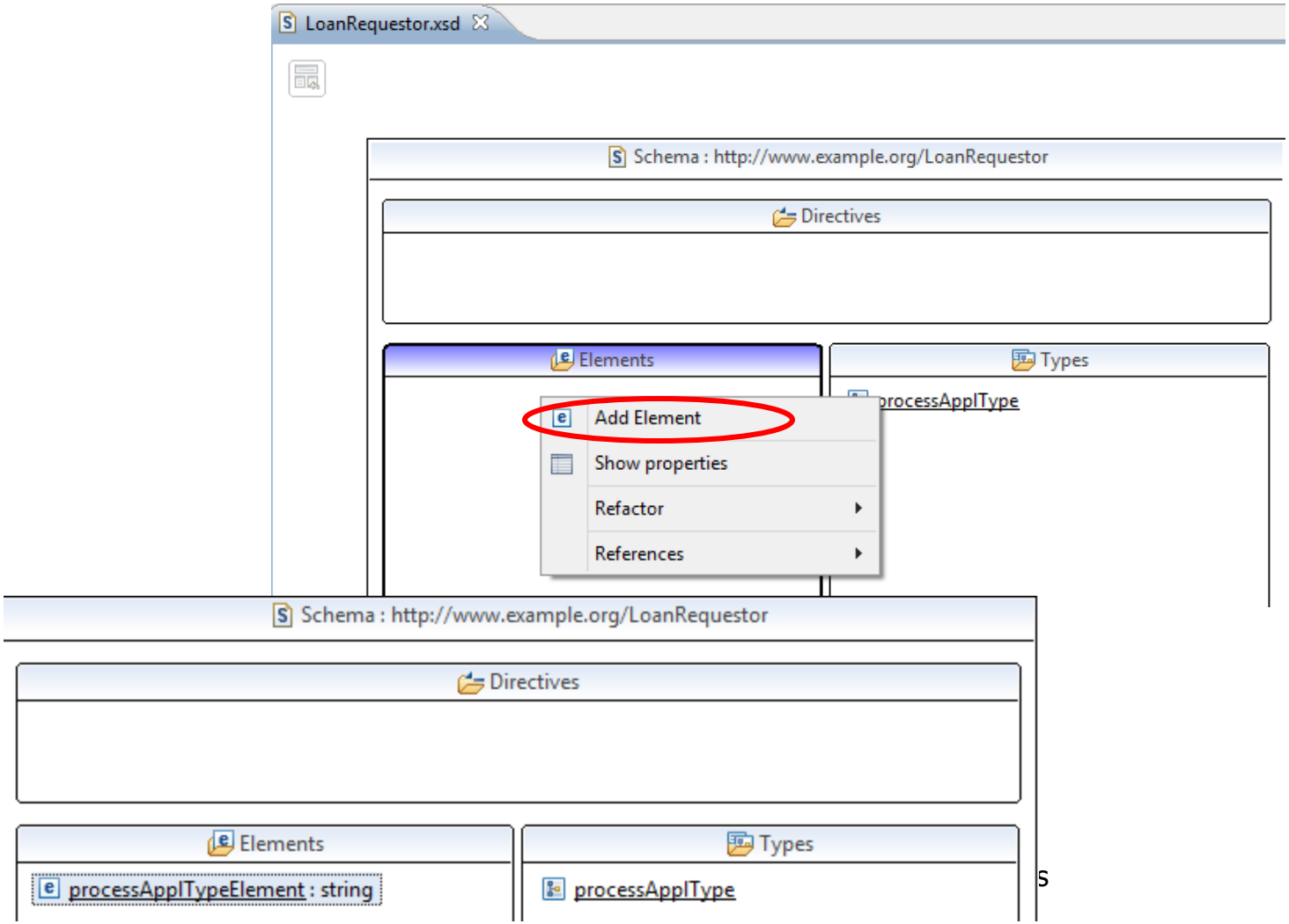
- Right click in the Types area and select “Add Complex Type” and name it “processApplType” (will be used for the loan request)



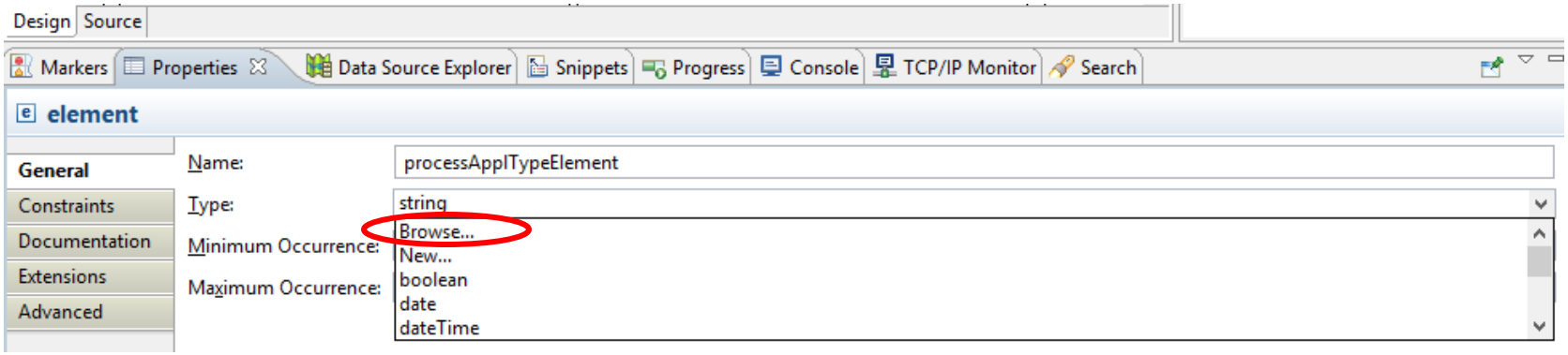
XSD Schema (2/6)

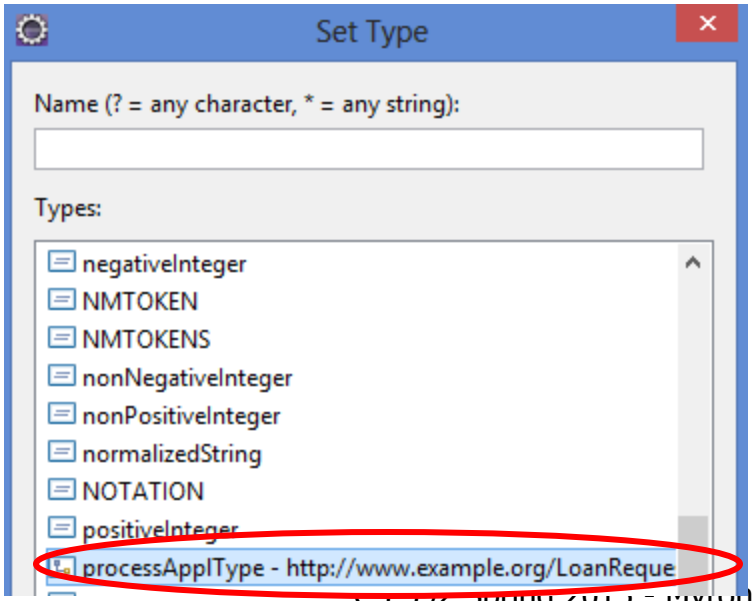


XSD Schema (3/6)



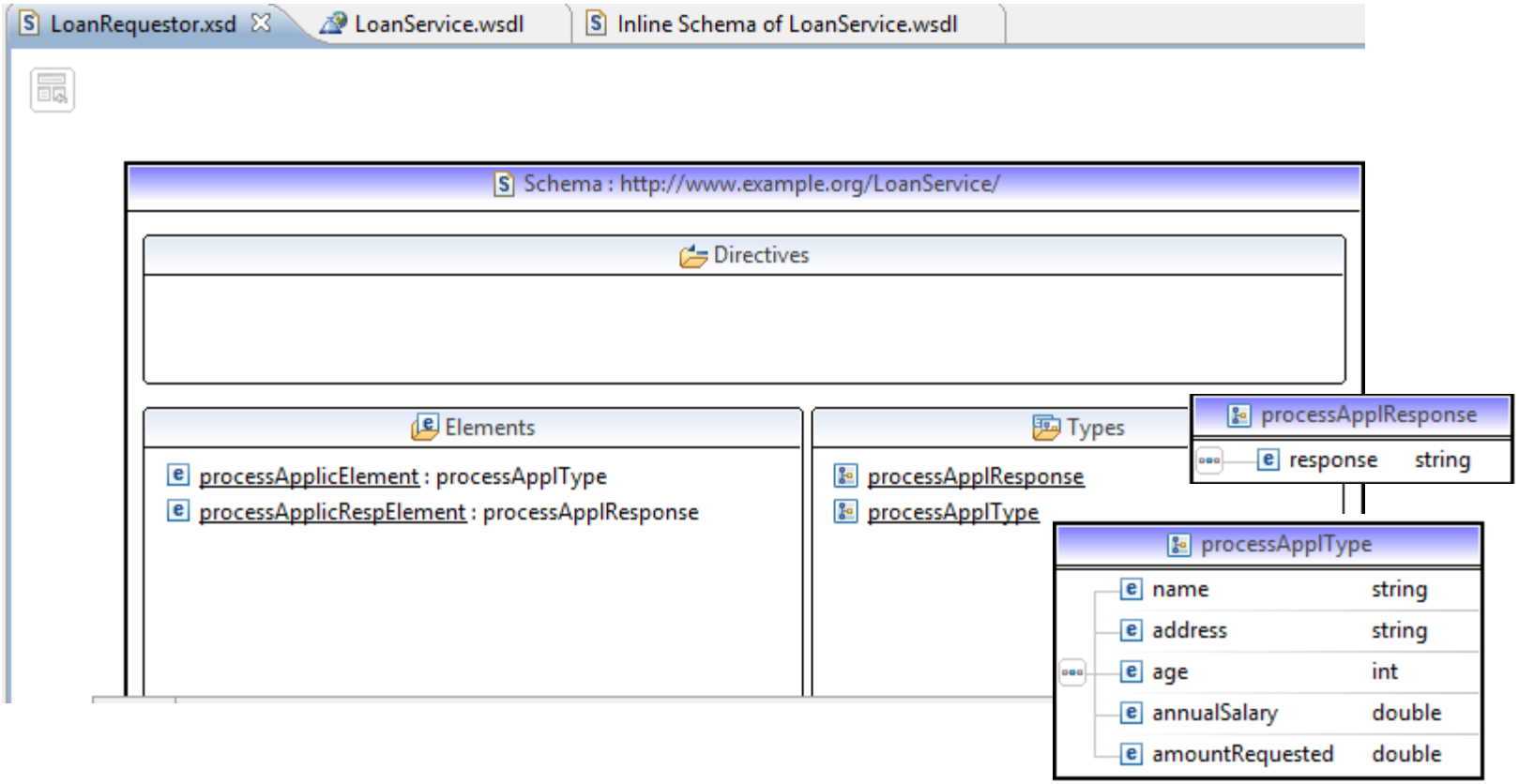
XSD Schema (4/6)



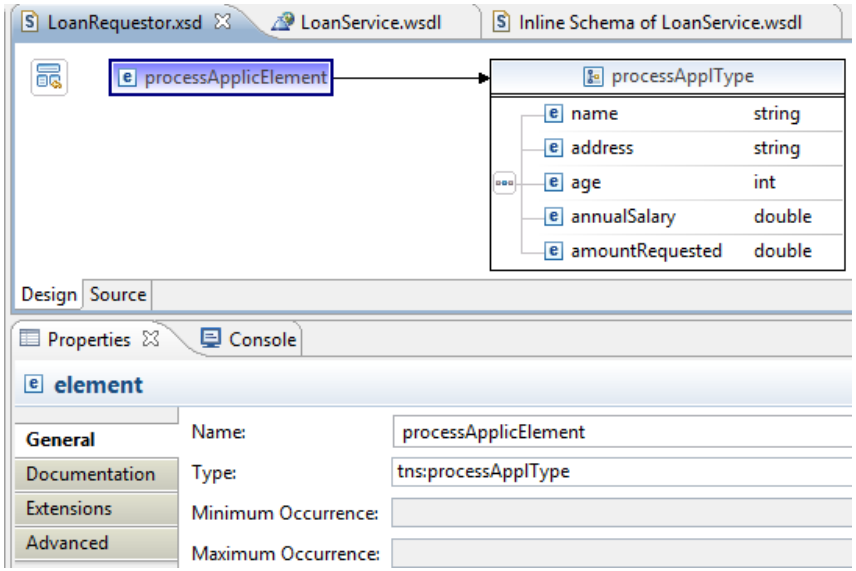


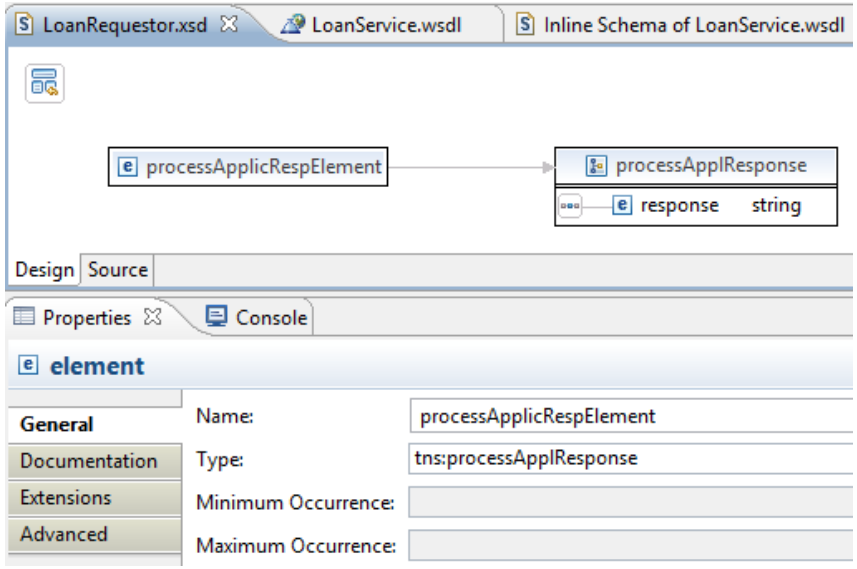
Follow similar steps for the response element..

XSD Schema (5/6)



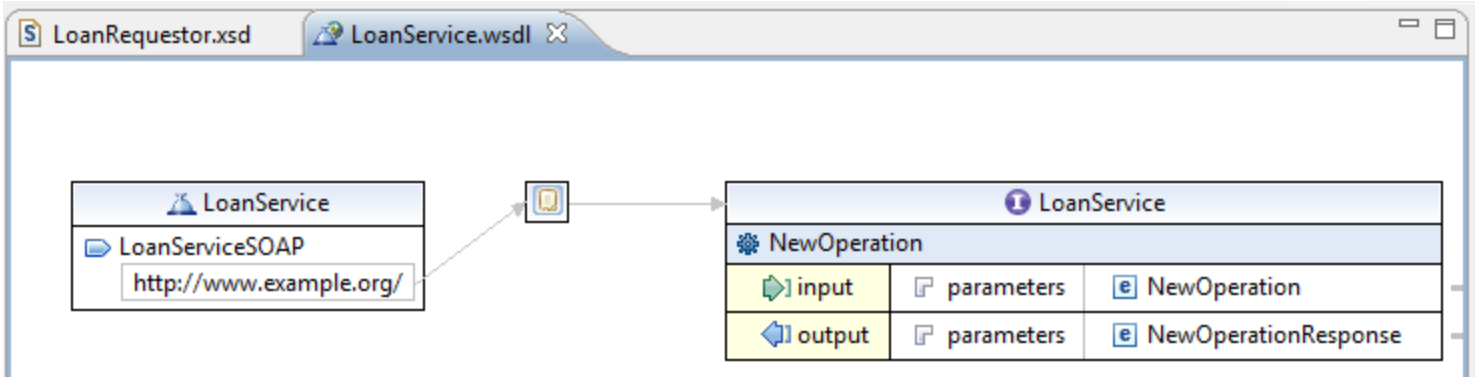
XSD Schema (6/6)





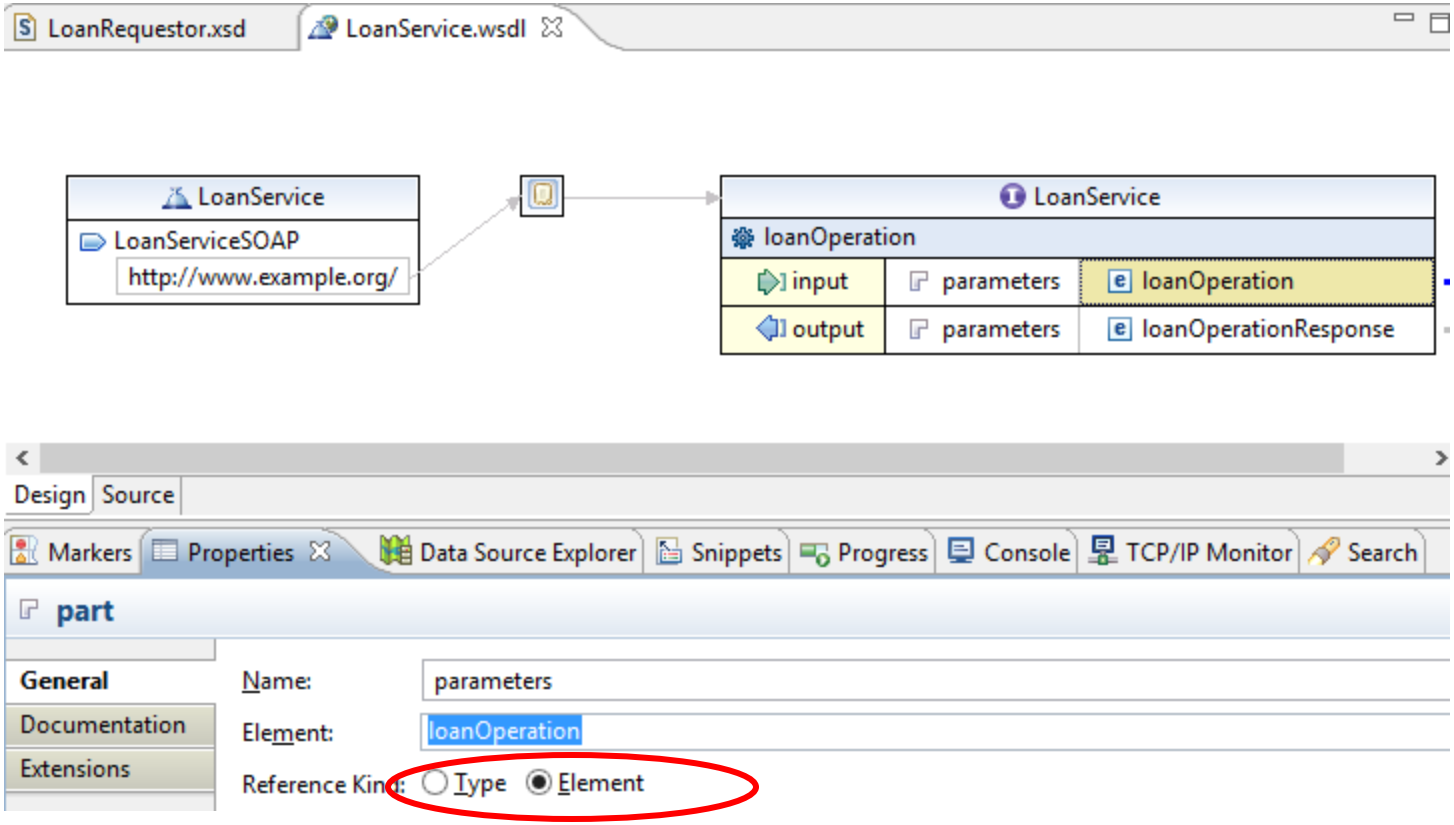
Create the WSDL

- Create the WSDL file for the Loan Web Service “LoanService.wsdl”
  - Right Click → File → New → Other → WSDL



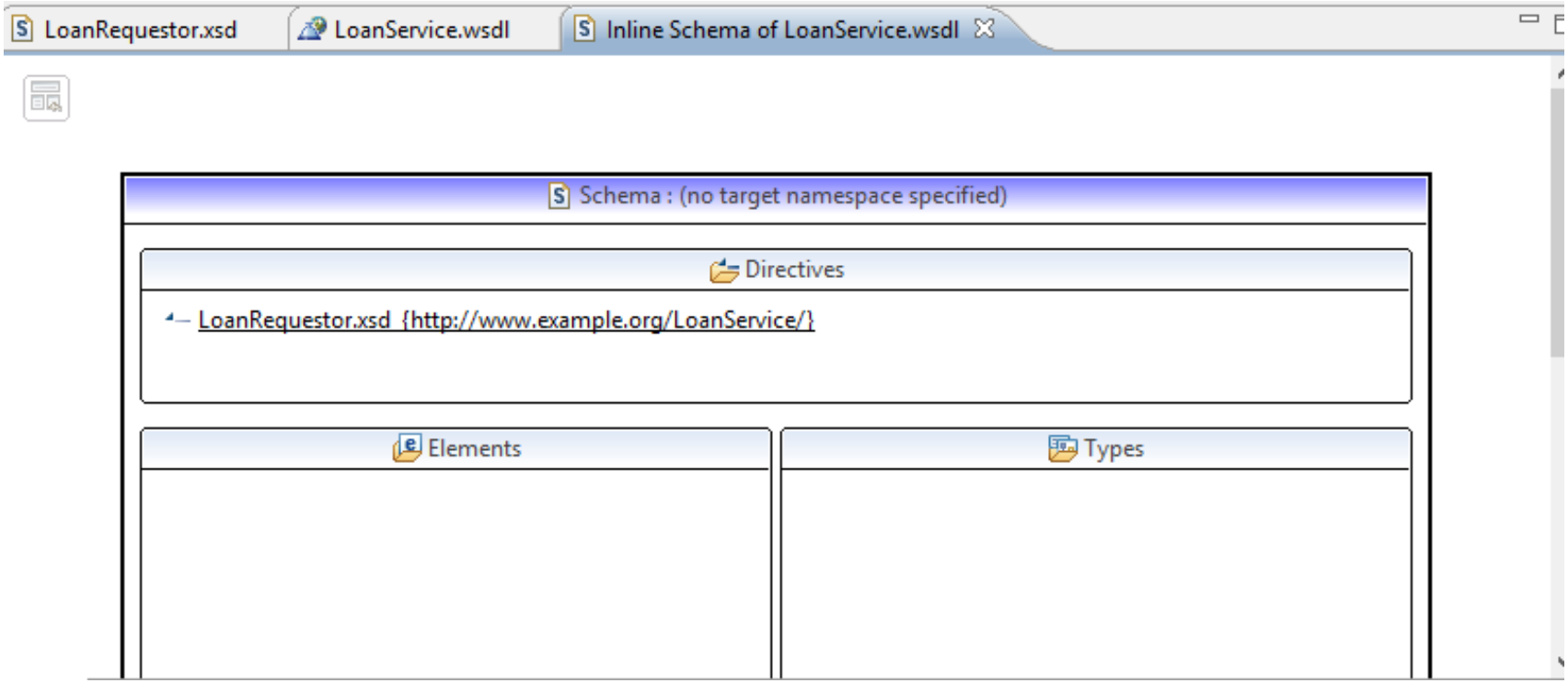
- Add a single loanOperation (change the default operation)

Create the WSDL



- Must import the XSD file we created earlier on in order to use the request and response elements…

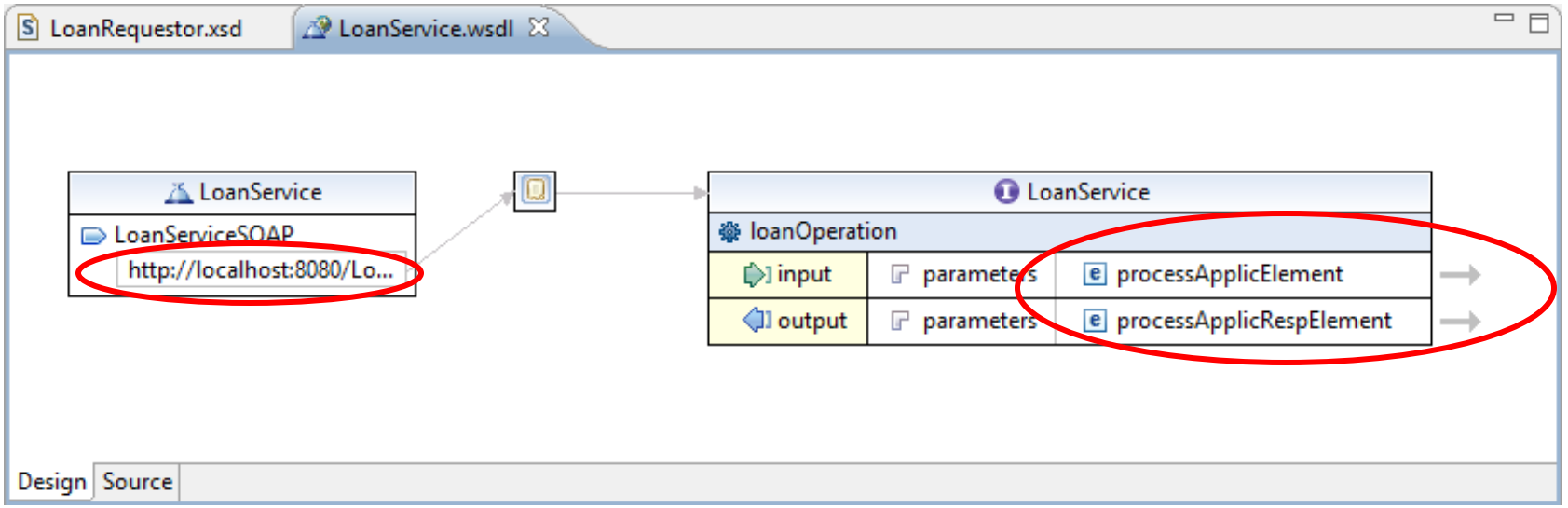
WSDL File and XSD



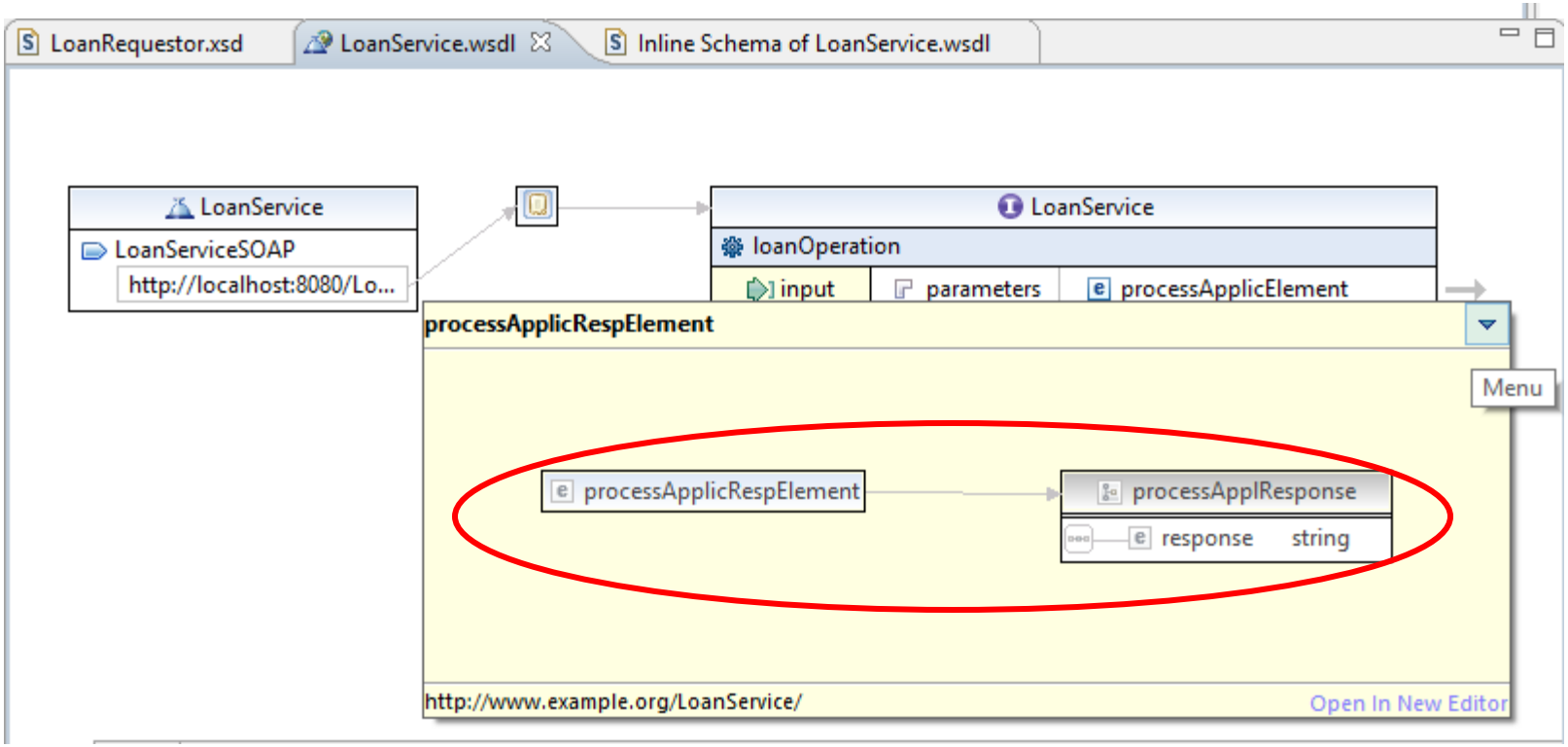

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<wsdl:definitions xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" xmlns:tns="http://www.example.org/LoanService/" xmlns:w
  <wsdl:types>
    <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
        <xsd:import schemaLocation="LoanRequestor.xsd" namespace="http://www.example.org/LoanService/"></xsd:import>
    </xsd:schema>
  </wsdl:types>
  <wsdl:message name="loanOperationRequest">
```

WSDL File

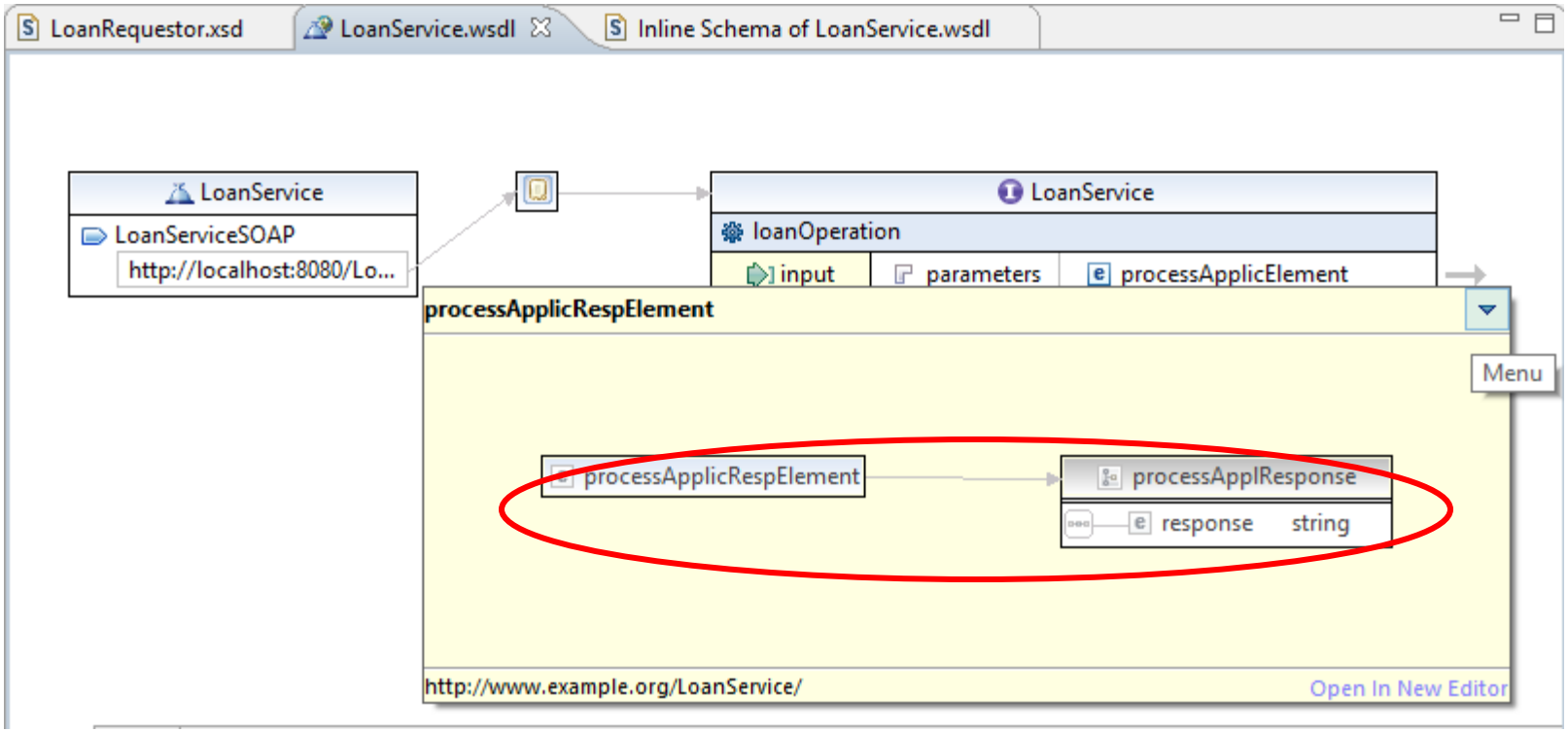
- **Use the XSD messages that you created earlier for the request and the response accordingly**.
- Do not forget to change the address of the LoanService (Properties View)
  - http://localhost:8080/LoanWS/services/LoanService
  - LoanWS: the name of the Eclipse Project



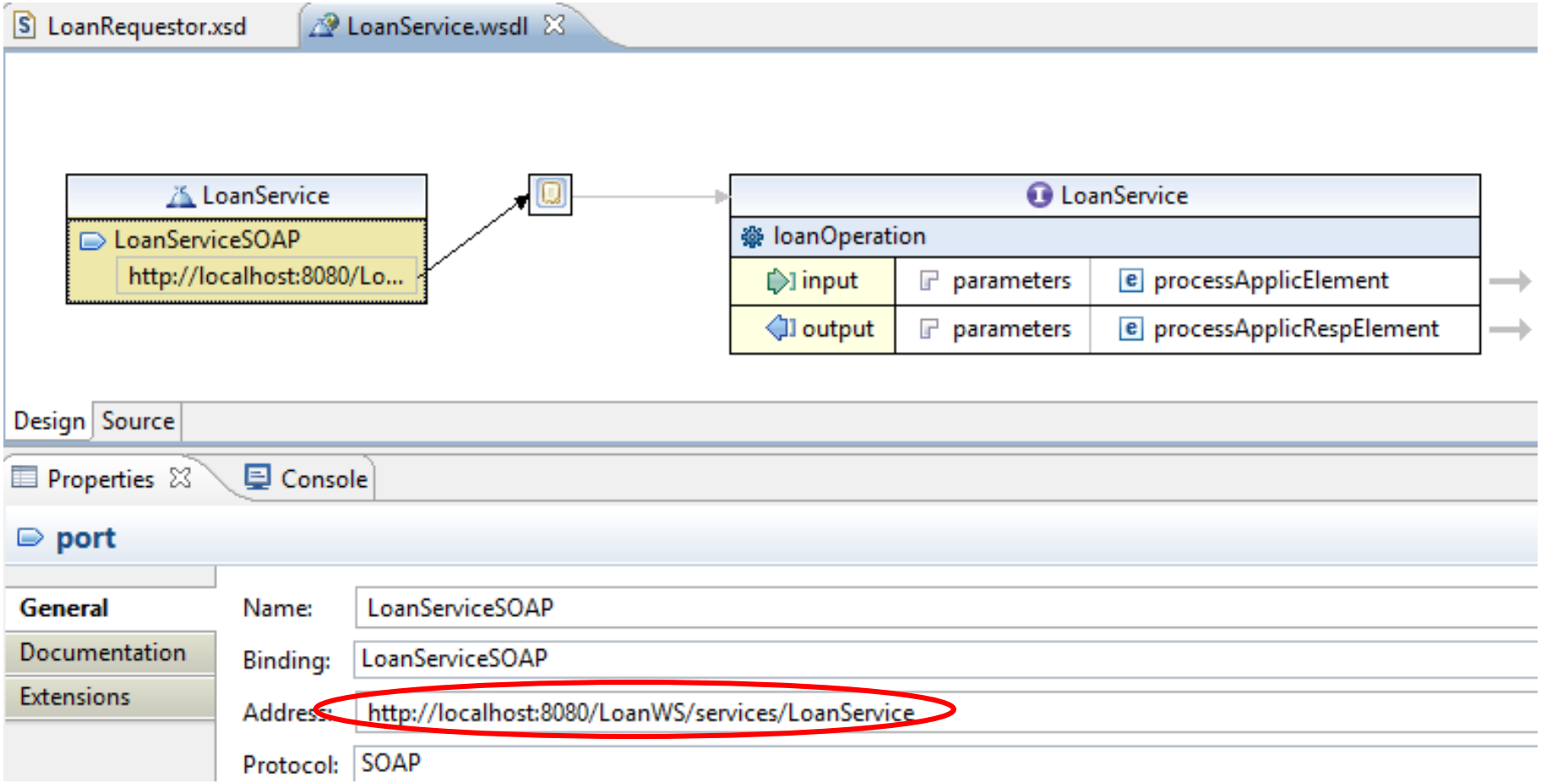
WSDL File


LoanRequestor.xsd
LoanService.wsdl
Inline Schema of LoanService.wsdl
LoanService
LoanServiceSOAP
http://localhost:8080/Lo...
LoanService
loanOperation
input
parameters
processApplicElement
processApplicRespElement
Menu
processApplicRespElement
processApplResponse
response
string
http://www.example.org/LoanService/
Open In New Editor

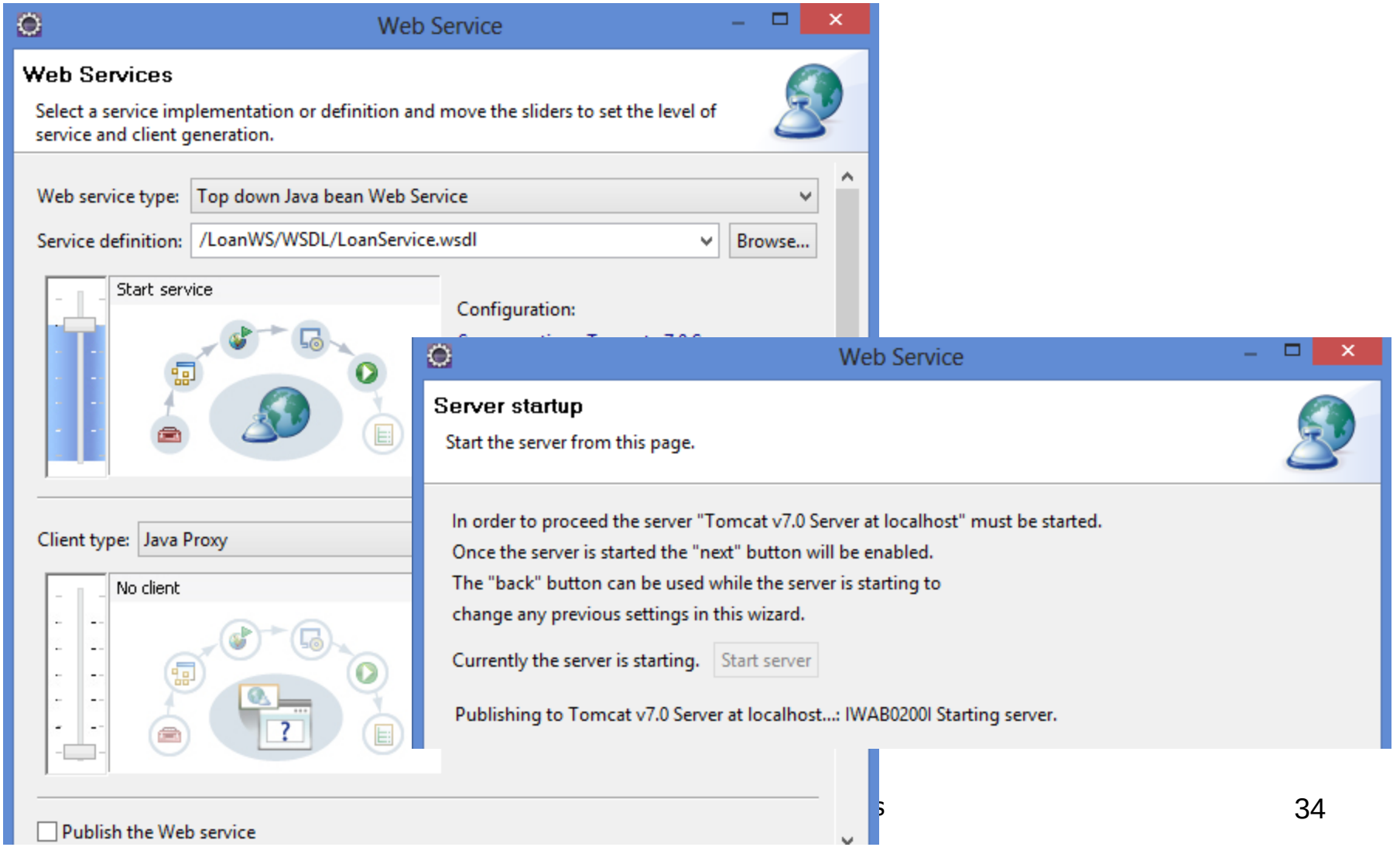
WSDL File


LoanRequestor.xsd
LoanService.wsdl
Inline Schema of LoanService.wsdl
LoanService
LoanServiceSOAP
http://localhost:8080/Lo...
LoanService
loanOperation
input
parameters
processApplicElement
processApplicRespElement
Menu
processApplicRespElement
processApplResponse
response
string
http://www.example.org/LoanService/
Open In New Editor

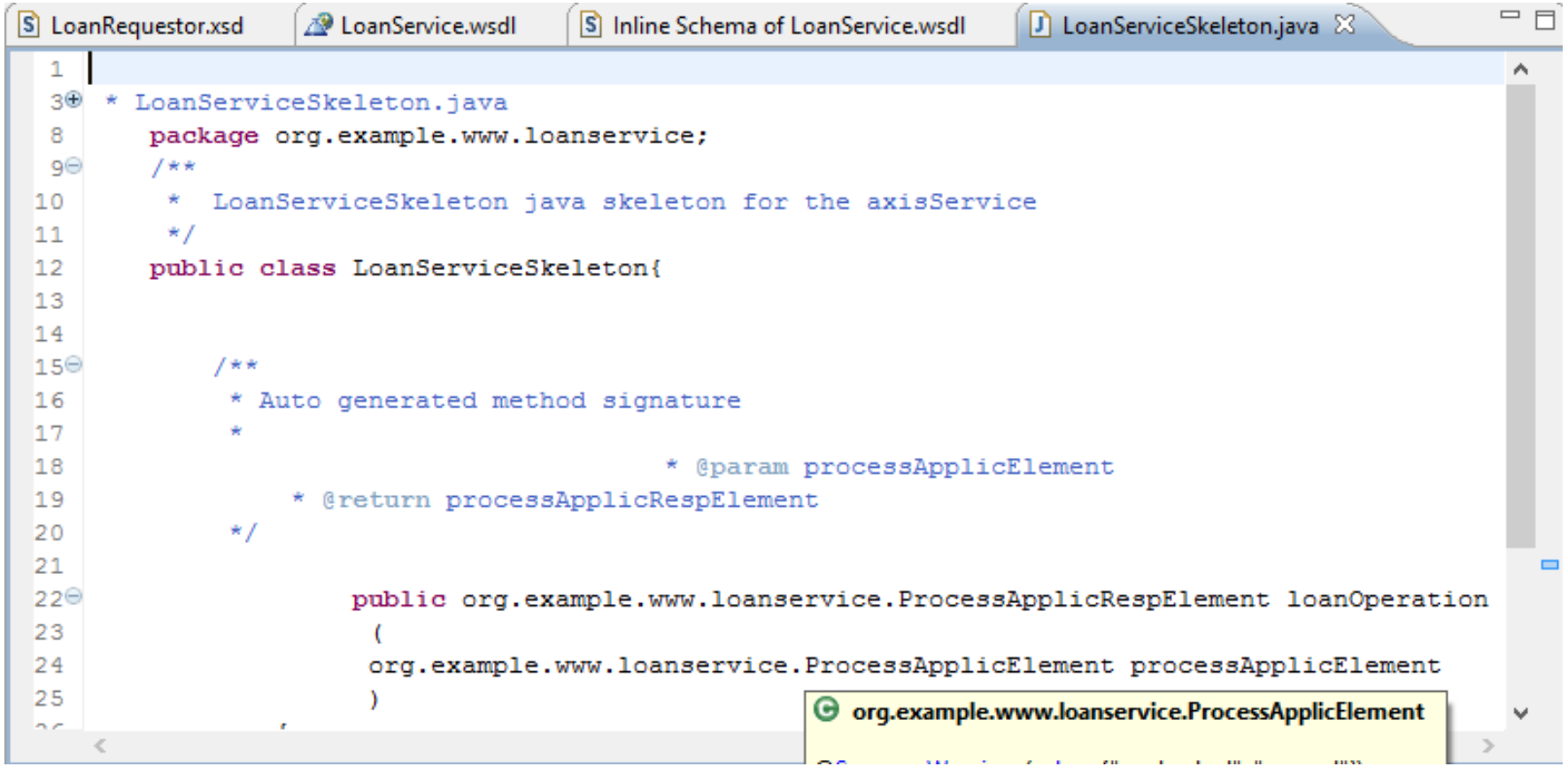
WSDL File


LoanRequestor.xsd
LoanService.wsdl
LoanService
LoanServiceSOAP
http://localhost:8080/Lo...
LoanService
loanOperation
input
parameters
processApplicElement
output
parameters
processApplicRespElement
Design
Source
Properties
Console
port
General
Documentation
Extensions
Name:
LoanServiceSOAP
Binding:
LoanServiceSOAP
Address:
http://localhost:8080/LoanWS/services/LoanService
Protocol:
SOAP

Axis2 Web Service from WSDL



Implementing the operations

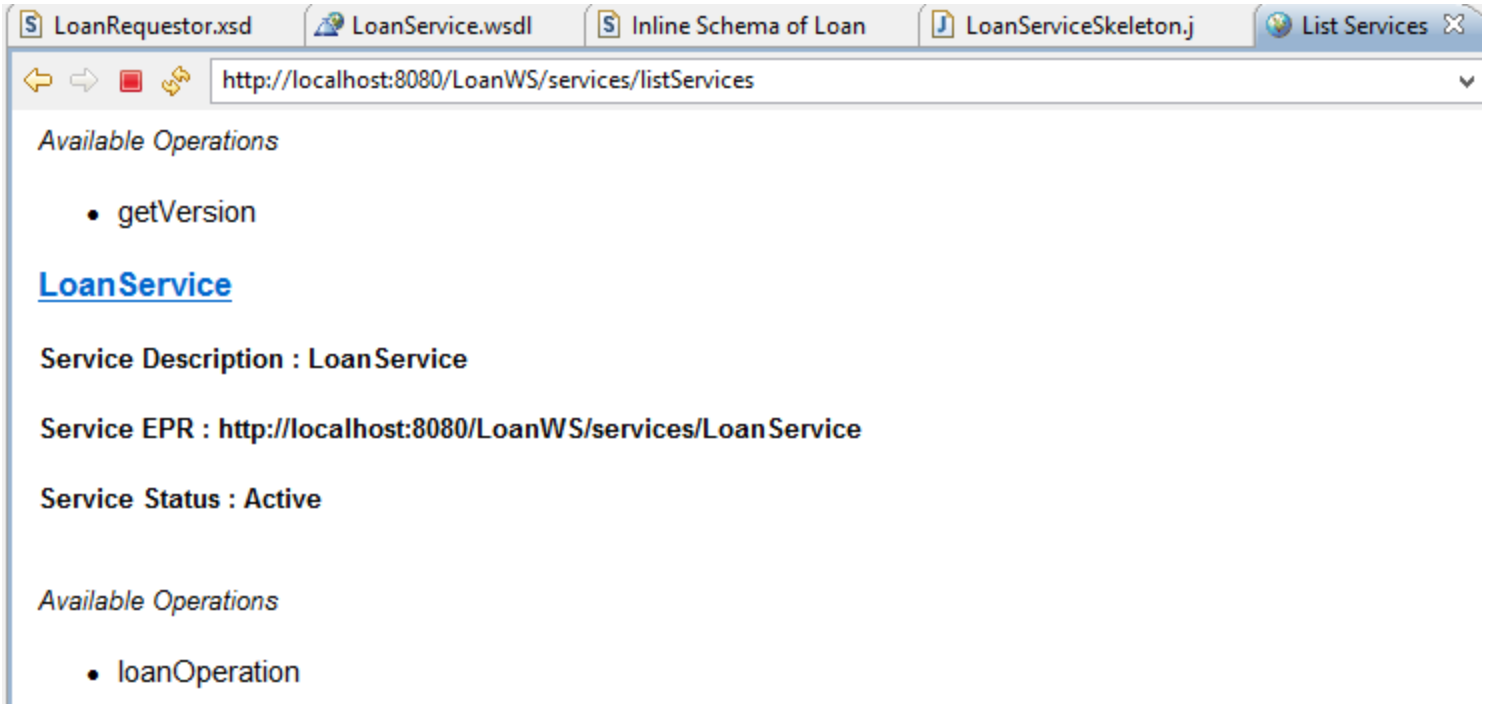


Implementing the operations

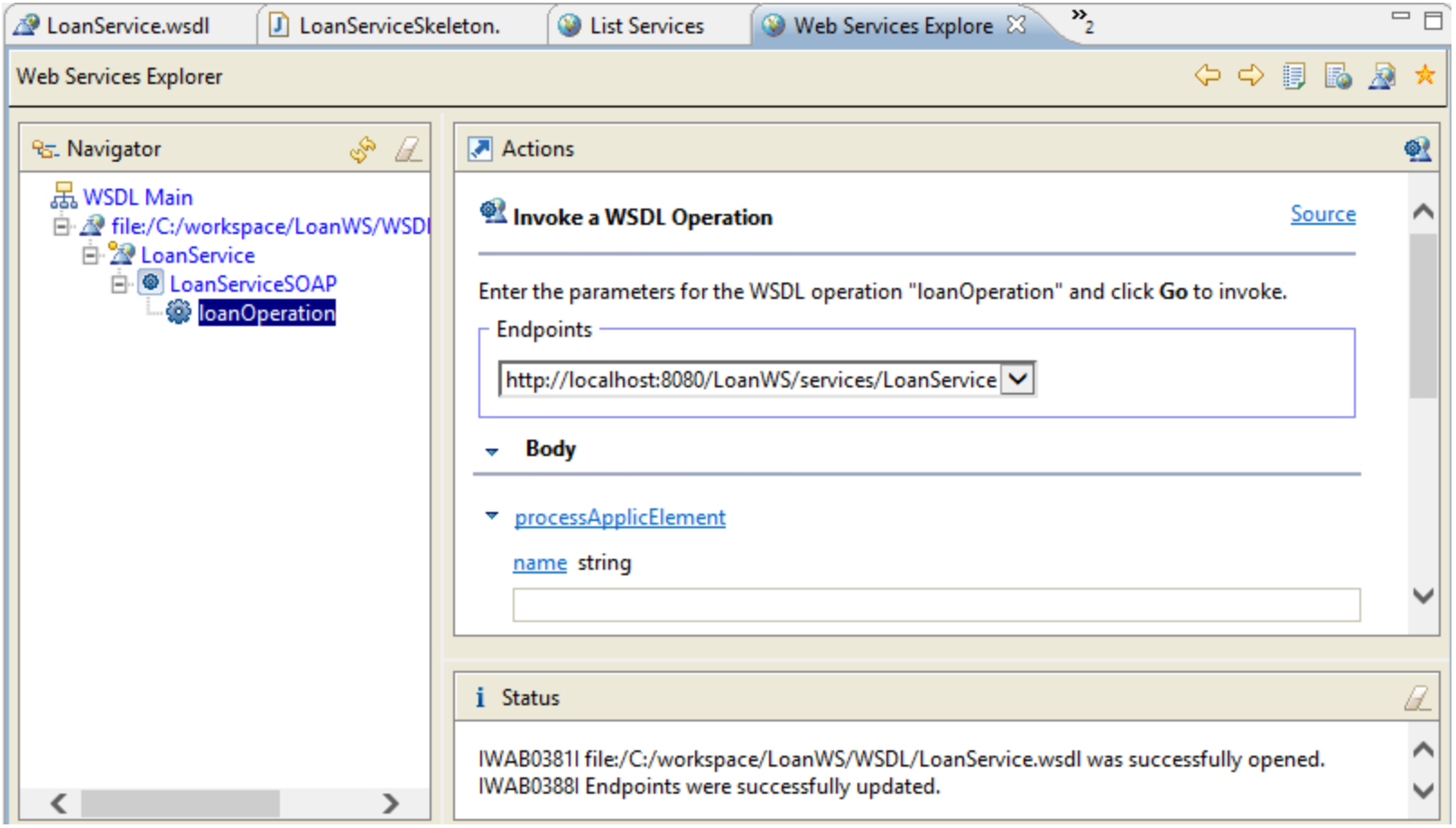
LoanRequestor.xsd | LoanService.wsdl | LoanServiceSkeleton.java

```
16      * Auto generated method signature
17      *
18      * @param processApplicElement
19      * @return processApplicRespElement
20      */
21
22     public org.example.www.loanservice.ProcessApplicRespElement loanOperation
23     (org.example.www.loanservice.ProcessApplicElement processApplicElement)
24     {
25         ProcessApplicRespElement response = new ProcessApplicRespElement();
26         ProcessApplResponse r = new ProcessApplResponse();
27
28         int age = processApplicElement.getProcessApplicElement().getAge();
29         double amountRequested = processApplicElement.getProcessApplicElement().getAmountRequested();
30         double annualSalary = processApplicElement.getProcessApplicElement().getAnnualSalary();
31
```

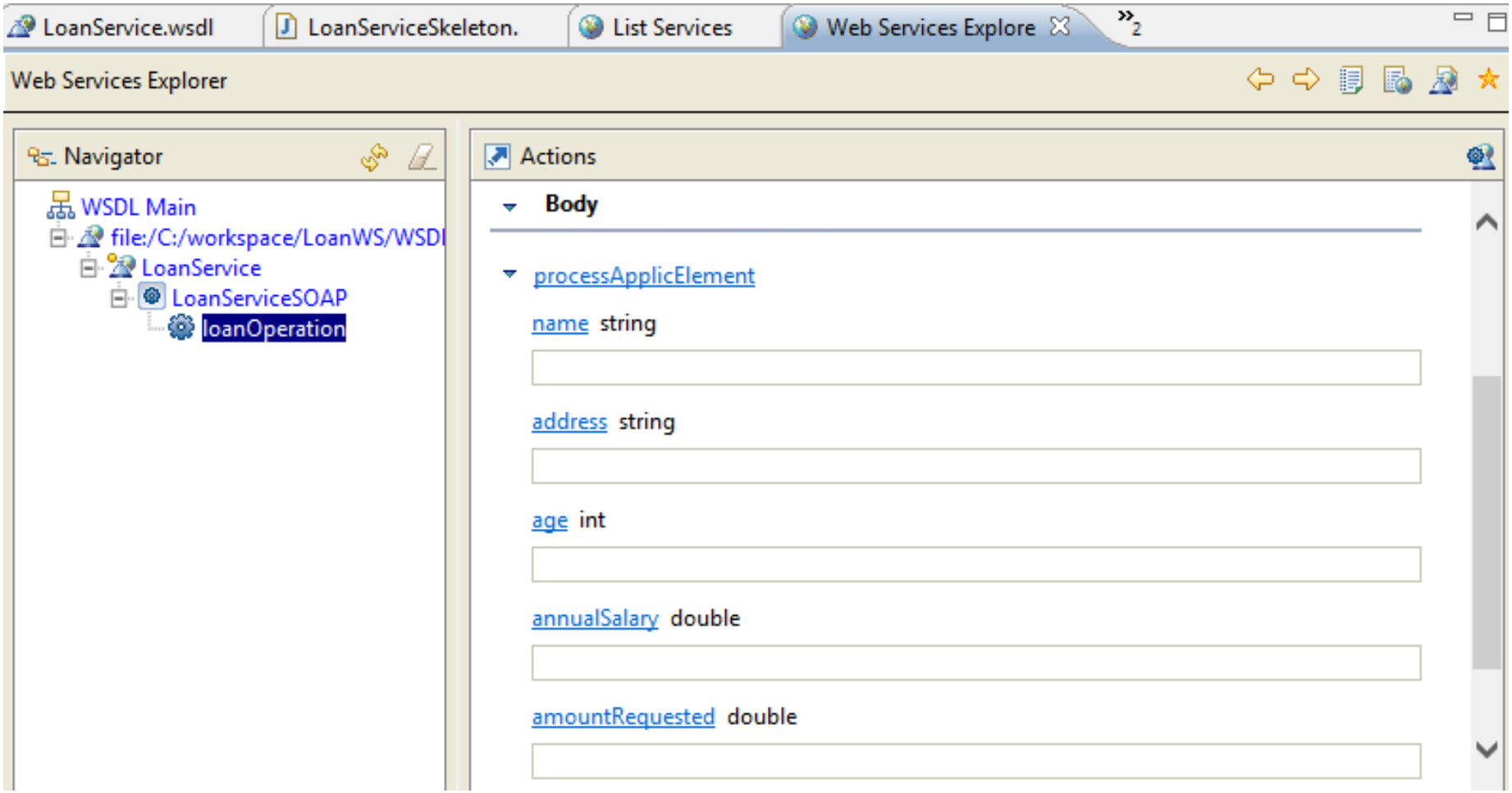
Test if the WS is running



WS in the Web Services Browser

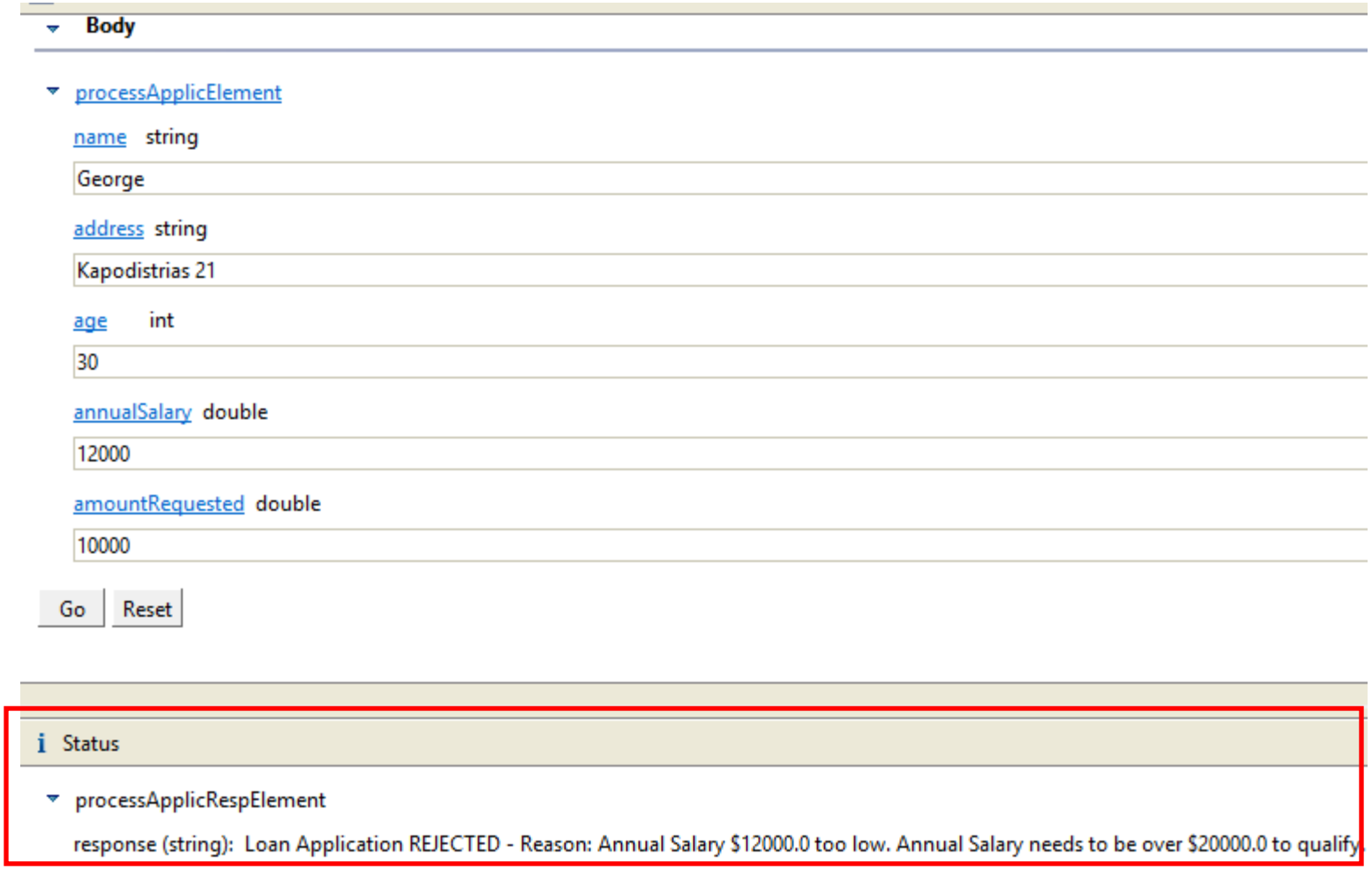


Testing the WS

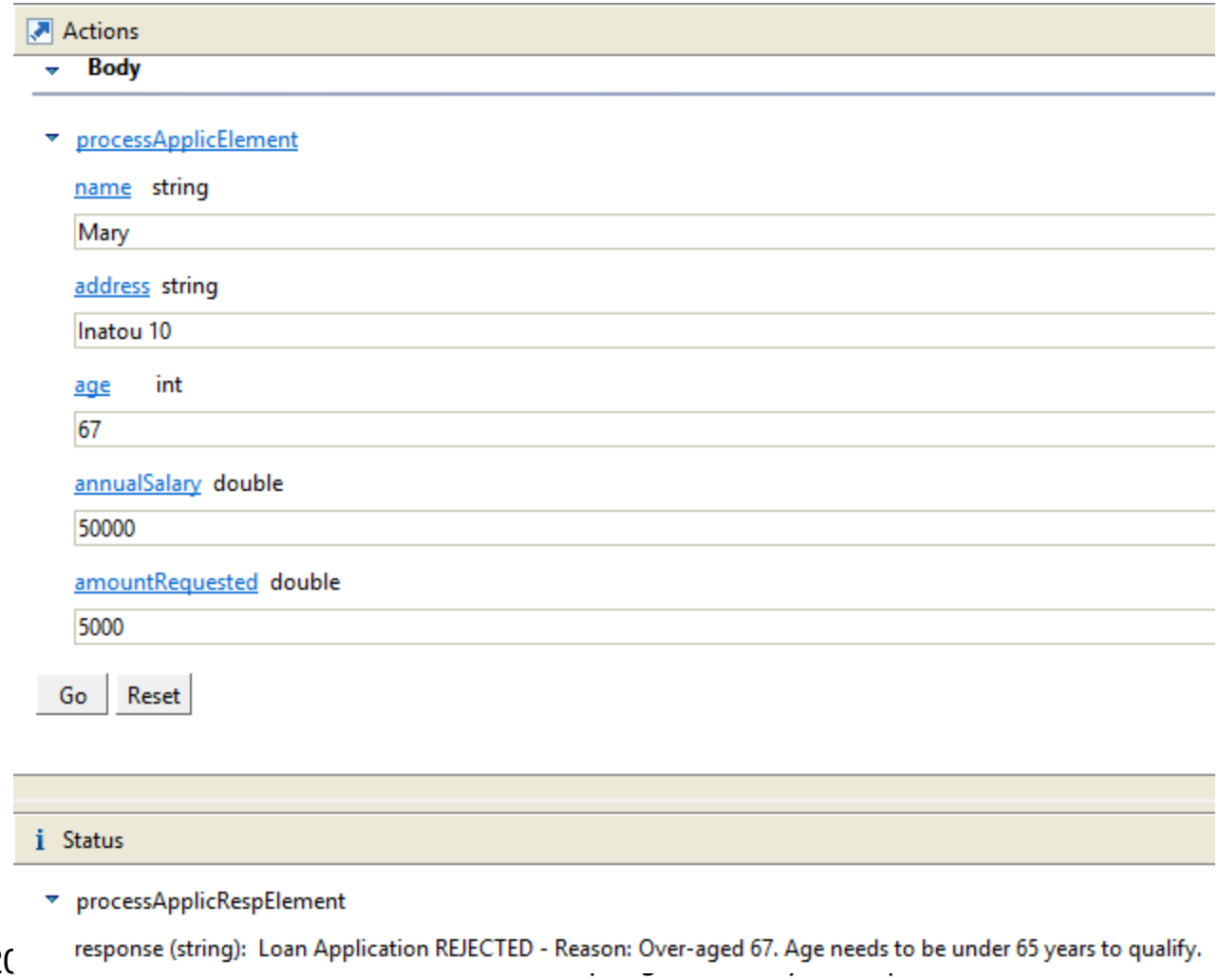


Testing the WS

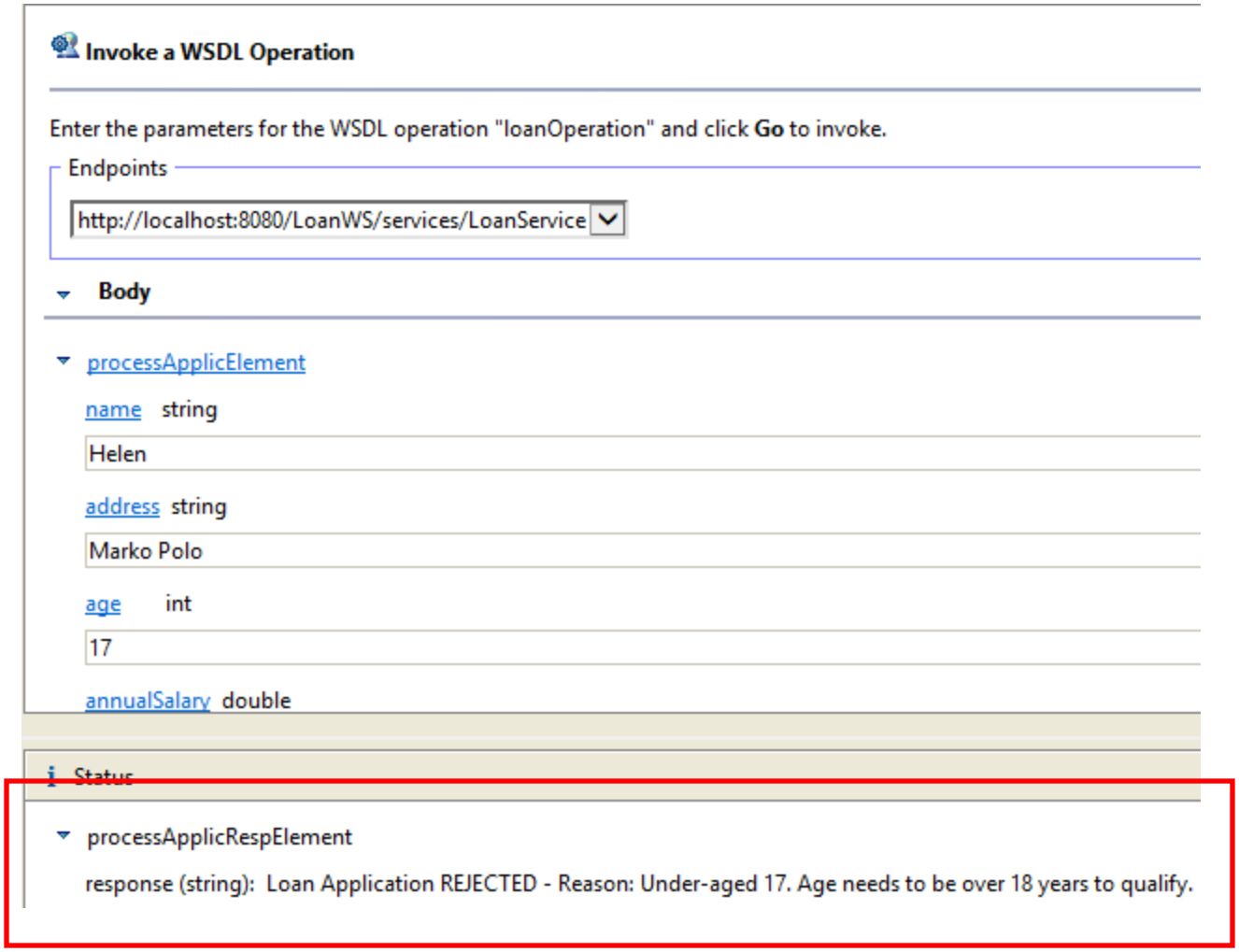
Testing the WS



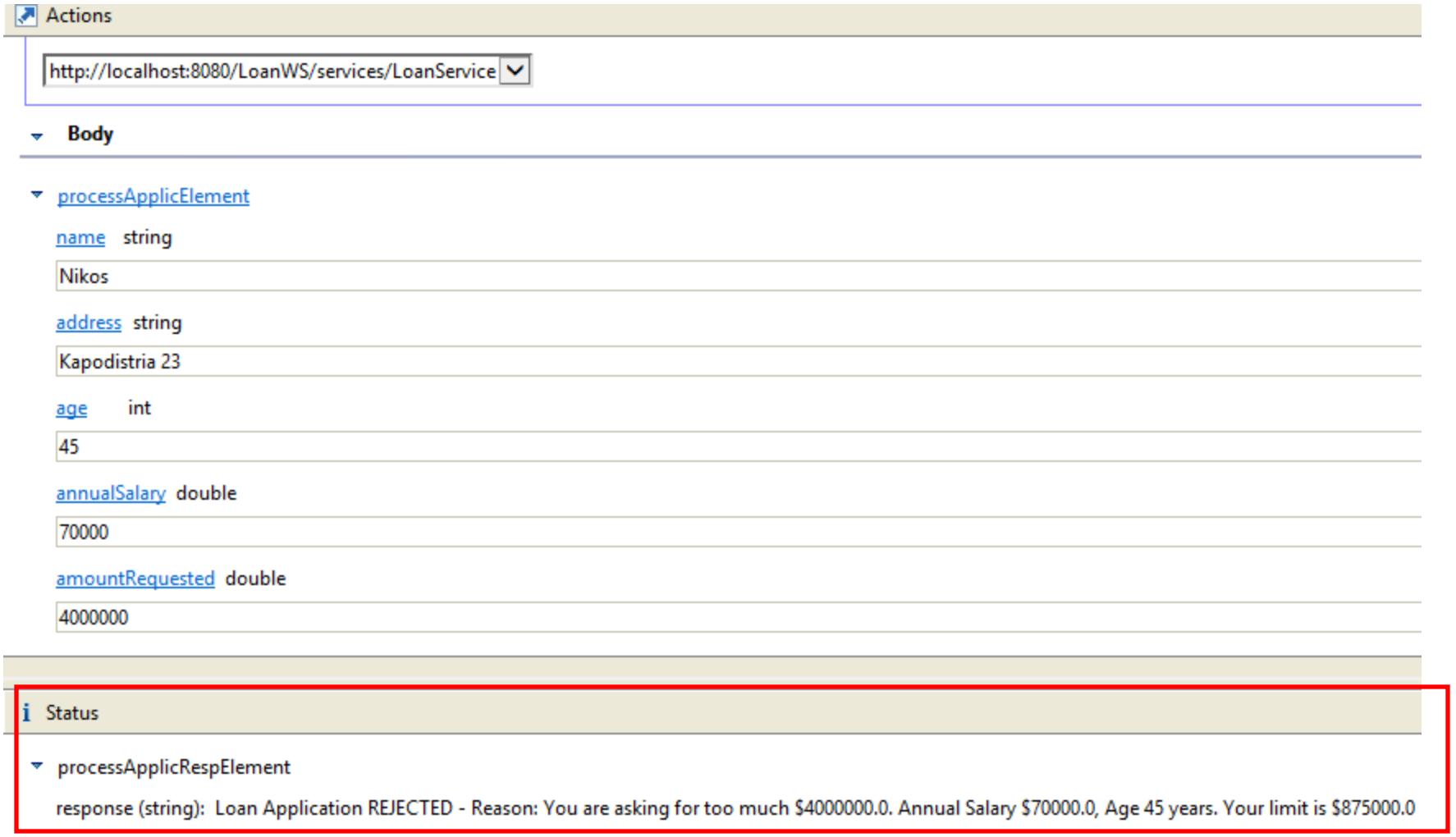
Testing the WS



Testing the WS



Testing the WS



Loan Client Example using PHP

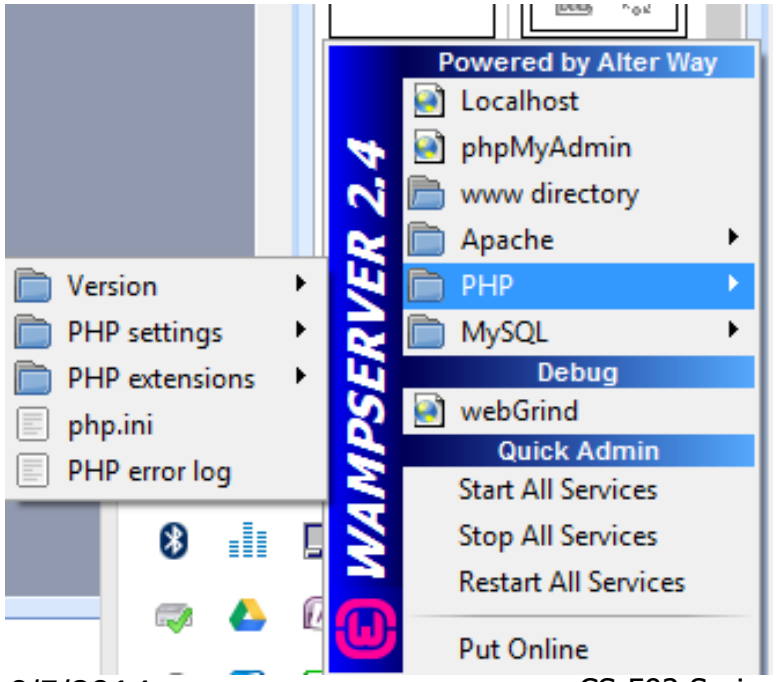
For writing a client manually in Java or testing the Web Service through the Web Services Browser, SOAP UI see the previous Assisting Lecture

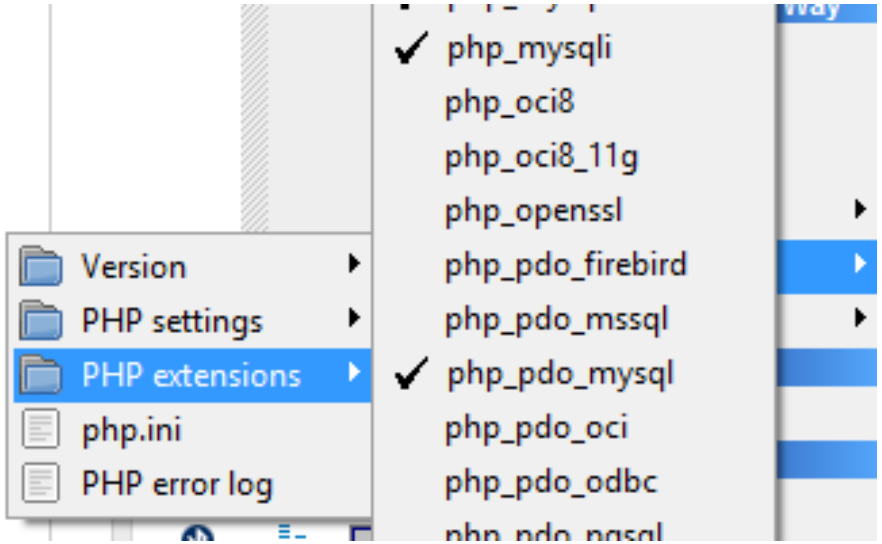
Prerequisites

- Php must be installed on your PC
- A program like XAMP or WAMP can be very helpful when developing php and mysql applications
- Download wamp for example
- Work Done in C:\wamp\www
- Place a test.php there to see if php is running

Php and SOAP

- Enable php_soap extensions(or delete the ';' from the corresponding extension of the php.ini file)
- Native SoapClient, see http://php.net/soapclient





Steps

- A php client can be developed in a very similar way as any other client (jsp client, java client, etc)

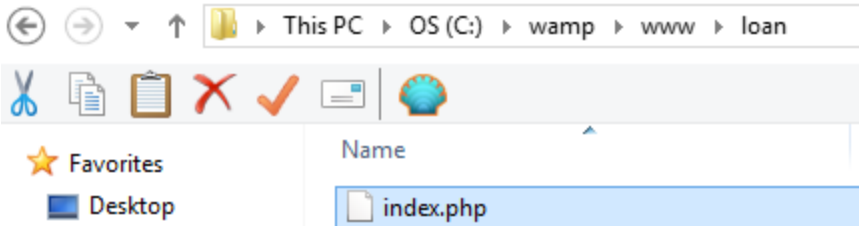
1. Check that the Web Service is running (check the wsdl file)
   - Most common you will have to start Tomcat from the Eclipse IDE (assuming that use one)
2. Develop the php client
   - Using for example a simple editor like Notepad++
3. Test the Web service using the client
   1. Through your Web browser (Firefox, Chrome, etc)

Step 1

- Before testing your Web Service (either in php or wherever) always make sure that the service is running.
- Otherwise you will retrieve a fault similar to the following one (after the development of the client)

Step 2

- Create a php file in the folder you created previously (i.e. index.php)



- We want to create an input form (html) so that users provide their details (name, address, salary, amount requested, etc)
  - You can use some css to style the form etc (e.g. style.css)
- The Web Service will be invoked when users submit their form
- The Web Service responds and users see the result of the loan request (approved or rejected and for which reason)
- In the following slides, a simple example follows (note that I do not tackle exceptional states, etc but you should do..)

Index.php

```
<!DOCTYPE HTML>
<html>
<head>
<link type="text/css" rel="stylesheet" href="style.css">
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<meta name="author" content="Myron Papadakis">
<title>PHP Client for Loan Scenario</title>
</head>
<body>
    <header class="main">
        <h1>Loan Request</h1>
    </header>
    <section class="main">
        <?php
         if(isset($_POST['name']) && isset($_POST['address']) && isset($_POST['age'])
         && isset($_POST['salary']) && isset($_POST['amount'])){
            $name = $_POST['name'];
            $address = $_POST['address'];
            $age = $_POST['age'];
            $salary = $_POST['salary'];
```

Index.php

```
    $amount = $_POST['amount'];
}
if(isset($_POST['submit'])){
 $wsdl = 'http://localhost:8080/LoanWS/services/LoanService?wsdl';
 $debug = false;
    try{
       $client = new SoapClient($wsdl);
       if($debug){
        var_dump ($client->__getFunctions());
        var_dump($client->__getTypes());
       }
       $params = array ('name' => $name, 'address' => $address, 'age' => $age, 'annualSalary' => $salary, 'amou
       $result = $client->loanOperation($params);
        if (is_soap_fault($result)) {
            trigger_error("SOAP Fault: (faultcode: {$result->faultcode},
            faultstring: {$result->faultstring})", E_USER_ERROR);
        }
        echo $result->response;

    }
    catch (Exception $e){
       echo "Error!";
       echo $e -> getMessage ();
    }
```

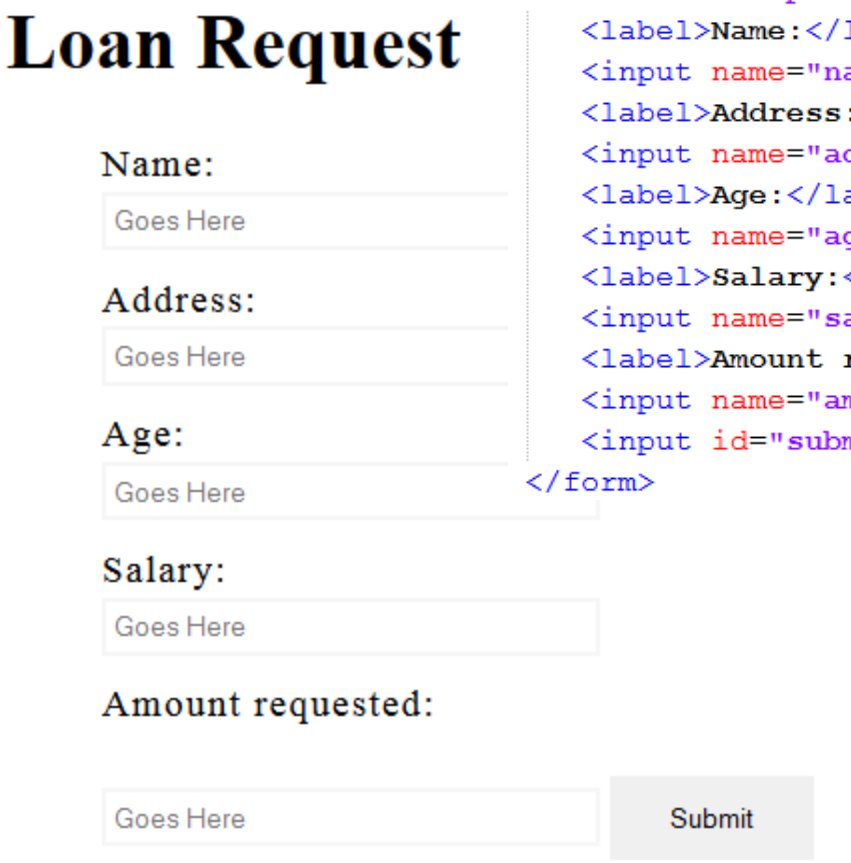
```
array (size=1)
  0 => string 'processApplResponse loanOperation(processApplType $parameters)' (length=62)
```

```
array (size=2)
  0 => string 'struct processApplType {
 string name;
 string address;
 int age;
 double annualSalary;
 double amountRequested;
}' (length=114)
  1 => string 'struct processApplResponse {
 string response;
}' (length=48)
```

Index.php

```
            }
        ?>
        <form method="post" action="index.php">
            <label>Name:</label>
            <input name="name" placeholder="Goes Here">
            <label>Address:</label>
            <input name="address" placeholder="Goes Here">
            <label>Age:</label>
            <input name="age" placeholder="Goes Here">
            <label>Salary:</label>
            <input name="salary" placeholder="Goes Here">
            <label>Amount requested:</label>
            <input name="amount" placeholder="Goes Here">
            <input id="submit" name="submit" type="submit" value="Submit">
        </form>
    </section>
</body>
</html>
```

Step 3



Index.php (part)

```
<form method="post" action="index.php">
    <label>Name:</label>
    <input name="name" placeholder="Goes Here">
    <label>Address:</label>
    <input name="address" placeholder="Goes Here">
    <label>Age:</label>
    <input name="age" placeholder="Goes Here">
    <label>Salary:</label>
    <input name="salary" placeholder="Goes Here">
    <label>Amount requested:</label>
    <input name="amount" placeholder="Goes Here">
    <input id="submit" name="submit" type="submit" value="Submit">
</form>
```

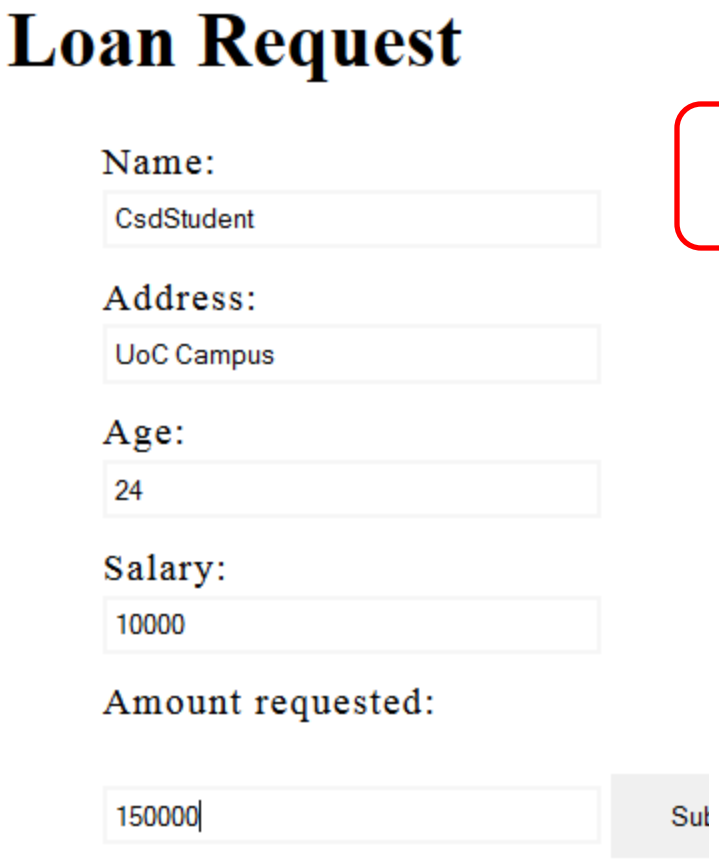
Style.css (part)

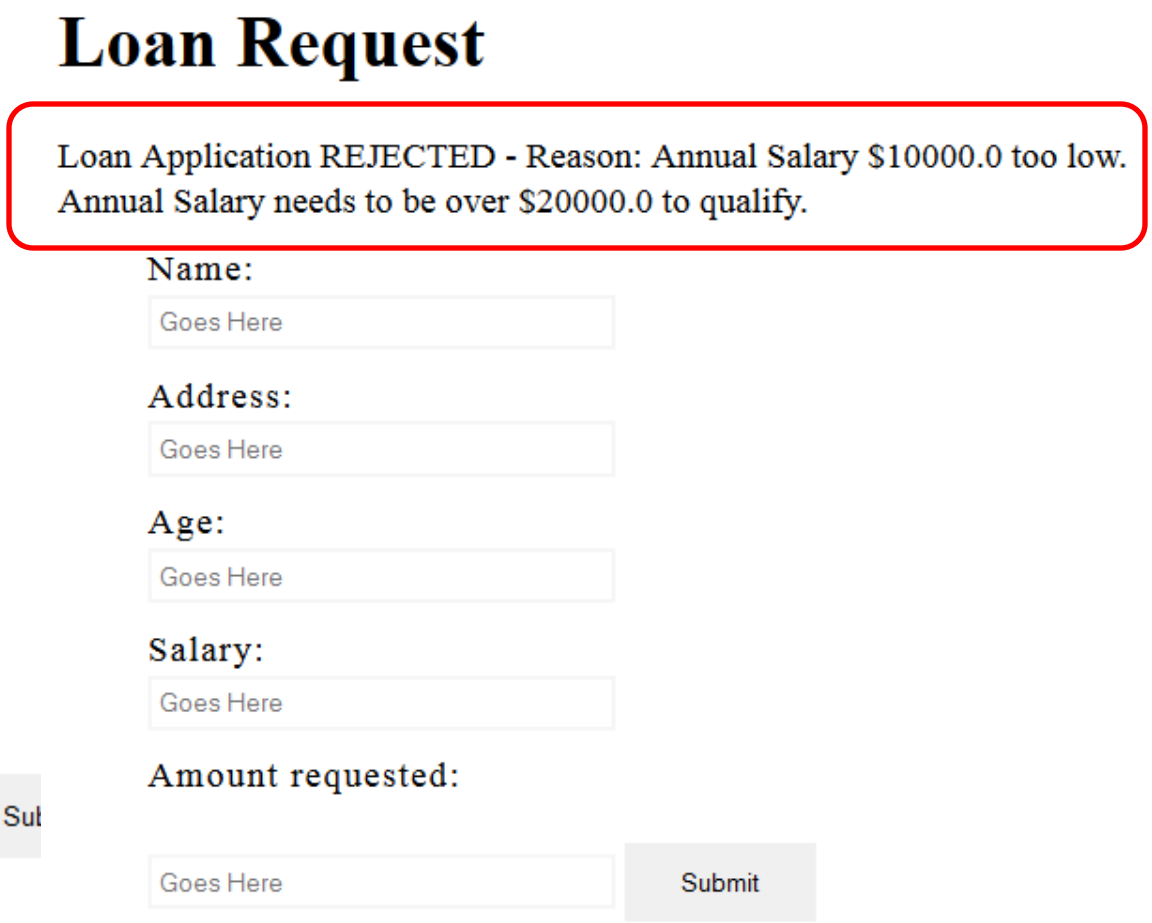
```
label {
    display:block;
    margin-top:10px;
    letter-spacing:1px;
}

/* This section centers our complete page */
.main {
    display:block;
    margin:0 auto;
    width:500px;
}

/* This section centers the form inside our web page*/
form {
    margin:0 auto;
    width:420px;
}
```

Demonstration Example





References

- **Top-down Approach:** http://www.eclipse.org/webtools/community/tutorials/TopDownAxis2WebService/td_tutorial.html

- http://www.soapui.org/Working-with-soapUI/getting-started.html

- http://php.net/manual/en/class.soapclient.php

Τέλος Ενότητας

Χρηματοδότηση

•Το παρόν εκπαιδευτικό υλικό έχει αναπτυχθεί στα πλαίσια του εκπαιδευτικού έργου του διδάσκοντα.

•Το έργο «**Ανοικτά Ακαδημαϊκά Μαθήματα στο Πανεπιστήμιο Κρήτης**» έχει χρηματοδοτήσει μόνο τη αναδιαμόρφωση του εκπαιδευτικού υλικού.

•Το έργο υλοποιείται στο πλαίσιο του Επιχειρησιακού Προγράμματος «Εκπαίδευση και Δια Βίου Μάθηση» και συγχρηματοδοτείται από την Ευρωπαϊκή Ένωση (Ευρωπαϊκό Κοινωνικό Ταμείο) και από εθνικούς πόρους.

Ευρωπαϊκή Ένωση
Ευρωπαϊκό Κοινωνικό Ταμείο

ΕΠΙΧΕΙΡΗΣΙΑΚΟ ΠΡΟΓΡΑΜΜΑ
ΕΚΠΑΙΔΕΥΣΗ ΚΑΙ ΔΙΑ ΒΙΟΥ ΜΑΘΗΣΗ
επένδυση στην κοινωνία της γνώσης
ΥΠΟΥΡΓΕΙΟ ΠΑΙΔΕΙΑΣ ΚΑΙ ΘΡΗΣΚΕΥΜΑΤΩΝ
ΕΙΔΙΚΗ ΥΠΗΡΕΣΙΑ ΔΙΑΧΕΙΡΙΣΗΣ

ΕΣΠΑ 2007-2013
πρόγραμμα για την ανάπτυξη
ΕΥΡΩΠΑΪΚΟ ΚΟΙΝΩΝΙΚΟ ΤΑΜΕΙΟ

Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης

Σημειώματα

Σημείωμα αδειοδότησης

- Το παρόν υλικό διατίθεται με τους όρους της άδειας χρήσης Creative Commons Αναφορά Δημιουργού - Μη Εμπορική Χρήση - Παρόμοια Διανομή 4.0 [1] ή μεταγενέστερη, Διεθνής Έκδοση. Εξαιρούνται τα αυτοτελή έργα τρίτων π.χ. φωτογραφίες, διαγράμματα κ.λ.π., τα οποία εμπεριέχονται σε αυτό και τα οποία αναφέρονται μαζί με τους όρους χρήσης τους στο «Σημείωμα Χρήσης Έργων Τρίτων».

[1] http://creativecommons.org/licenses/by-nc-nd/4.0/

•Ως **Μη Εμπορική** ορίζεται η χρήση:

–που δεν περιλαμβάνει άμεσο ή έμμεσο οικονομικό όφελος από την χρήση του έργου, για το διανομέα του έργου και αδειοδόχο

–που δεν περιλαμβάνει οικονομική συναλλαγή ως προϋπόθεση για τη χρήση ή πρόσβαση στο έργο

–που δεν προσπορίζει στο διανομέα του έργου και αδειοδόχο έμμεσο οικονομικό όφελος (π.χ. διαφημίσεις) από την προβολή του έργου σε διαδικτυακό τόπο

•Ο δικαιούχος μπορεί να παρέχει στον αδειοδόχο ξεχωριστή άδεια να χρησιμοποιεί το έργο για εμπορική χρήση, εφόσον αυτό του ζητηθεί.

.

Σημείωμα Αναφοράς