University of Central Lancashire
UCLan

# CO3409 Distributed Enterprise Systems

## Lab : Creating and Consuming Web Services

## Summary

Create a web service and develop clients to consume it.

## Purpose

On completion of this you should be able to

- Use NetBeans to create a Web Client
- Use an object automatically generated through WSDL which hides SOAP access to a Web Service
- Develop a JSP page and a Servlet

## Things to Remember

Some things to remember...

- Run Netbeans as "Administrator".
- When running your application, if you are using a database please ensure your database is running first (see previous lab worksheets).
- To deploy an Enterprise Application, I suggest using Clean and Build and then Deploy.

## Activities

# 1. Creating and testing a Web Service using the NetBeans IDE

Do the exercise in and read the introduction: https://netbeans.org/kb/docs/websvc/jax-ws.html.

> **Note** :
>
> - If you have a problem that says you need to enable accessing external schema, See http://wiki.netbeans.org/FaqWSDLExternalSchema for more on the problem. Fix it by copying and pasting the NetBeans Icon, right-clicking and choosing properties and adding `-J-Djavax.xml.accessExternalSchema=all` to the target after the current target of the link (e.g. `C:\Program Files\NetBeans 8.0\bin\netbeans64.exe`).
> - When choosing the container, select EJB module from the Java EE category and choose Java EE 7.
> - When creating `Client 1`, make sure that the `CalculatorWSApplication` is deployed and Glassfish is still running.
> - To fix an error about importing, add the following to `build.xml`
>
> ```
> <target name="wsimport-init" depends="init">
>    <mkdir dir="${build.generated.sources.dir}/jax-ws"/>
>       <taskdef name="wsimport"
> classname="com.sun.tools.ws.ant.WsImport">
>             <classpath path="${libs.jaxws21.classpath}"/>
>             <classpath path="${libs.jaxb.classpath}" />
>       </taskdef>
> </target>
> ```
>
> - When creating `Client 2` “the servlet”, you might get an error telling you it can’t find the `wsdl file`. Try and fix the error yourself!
> - Make sure you leave Generate dispatch code unchecked in the New Web Service Client wizard

# 2. Consuming a Web Service using NetBeans IDE

> **Note:** There may be a problem with SSL which may impact your ability to complete this activity. Don't worry!

Now, you should be able to discover a web service and write an application to consume it. Go to this website http://wiki.cdyne.com/Credit_Card_Verification and develop a web service client in NetBeans to verify a credit card number entered by a customer. You could develop JavaSE application or a servlet or a JSP as the web service client. The following instructions create a Facelets application.

> **Hints:**

> 1. Look for the wsdl URL first.
> 2. A valid test credit-card number is `4242424242424242`
> 3. **DON'T ENTER YOUR OWN CREDIT CARDS** for checking!!!!!!!!!!

1. Create a new Java Web, Web Application (Java with Ant)., called `CheckCC`. Make sure the Payara Server and Java EE 7 are selected. In the Frameworks step, choose JavaServer Faces. In the Libraries tab, leave the library as Registered Library (JSF 2.2) and in the Components tab, choose `PrimeFaces`. (There may be a delay.)

2. Add the following to the generated `index` page:

```
<h3>Credit Card</h3>
    <h:form>
        <p:inputTextarea id="counter" rows="1" cols="30" value="#
{listenerView.text}" counter="display" maxlength="25"
counterTemplate="{0} characters remaining." autoResize="false" >
            <p:ajax event="keyup" update="out" listener="#
{listenerView.handleKeyEvent}" />
        </p:inputTextarea>
        <h:outputText id="display" />
        (<h:outputText id="out" value="#{listenerView.status}" />)

    </h:form>
```

3. Fix the imports (remember you are using `PrimeFaces`).
4. Create a new Java class in the package beans and fix the imports

> **Hint**: `javax.enterprise.context.RequestScoped;`):

```
@Named
@RequestScoped
public class ListenerView {
     private String text;
     private String status;

    public String getStatus() {
        return status;
    }

    public void setStatus(String status) {
        this.status = status;
    }

    public String getText() {
        return text;
```

```
    }
    public void setText(String text) {
        this.text = text;
    }

    public void handleKeyEvent() {
        if("hi".equals(text))
            status = "valid";
        else
            status = "";
    }
}
```

5. Add a **new** `beans.xml` from Contexts and Dependency Injection.

6. Right-click on the project name and choose New Web Service Client from the Web Services folder. Enter the CDYNE wsdl URL. Leave the package blank. **Finish**.

7. Right-click in the `ListenerView` class and choose Insert Code, Call Web Service Operation.

   Expand until you come to the `CheckCC` operation.

8. Amend the ListenerView handleKeyEvent method to check the validity of the text;

```
ReturnIndicator ri = checkCC(text);
if(ri.isCardValid())
```

9. After building the application, when you run it, there will be an error: switch to the files tab and copy the appropriate sub folder from within the sources conf ... folder to the appropriate subfolder in build web Web-INF wsdl.

   > Unfortunately, you will have to repeat this every time you clean and build.

> **Note:** There may be a problem with SSL which may impact your ability to complete this activity. Don't worry!

## 3. Understanding WSDL - Question Time

In the projects pane, expand the Configuration Files folder and keep going (and going!) until you reach the `luhnchecker.asmx.wsdl` file. Double-click on it to open it in the editor and answer the following questions:

1. What language is WSDL implemented in?

2. Look at the CheckCCResponse. It contains a sequence. How many elements are in the sequence? What type is each element?

3. What is the name of the service?

4. What operations does the service provide?

Answer the following questions by expanding the nodes displayed by the XML editor.

1. How many Complex Types and how many Elements are there

2. Compare NetBeans with your browser as a way of displaying WSDL.