

## Chapter - Understanding web service calling process

### Objectives

---

- At the end of this chapter you will be able to
  - ♦ understand what is happening internally when we call a web service.

## Calling a web service without a client stub

- Suppose that we'd like to call a web service without a client stub.
- Create a file *LowLevelClient.java* in a new *com.ttdev.ss.lowlevel* package:

3



## Calling a web service without a client stub

```
import org.apache.axiom.om.OMElement;
import org.apache.axis2.AxisFault;
import org.apache.axis2.addressing.EndpointReference;
import org.apache.axis2.client.Options;
import org.apache.axis2.client.ServiceClient;

public class LowLevelClient {
    public static void main(String[] args) throws AxisFault {
        ServiceClient client = new ServiceClient();
        Options options = new Options();
        options.setTo(new EndpointReference(
            "http://localhost:8080/axis2/services/SimpleService"));
        client.setOptions(options);
        OMElement request = makeRequest();
        OMElement response = client.sendReceive(request);
        System.out.println(response.toString());
    }
}
```

Create a service client object. You will use it to call the web service.

Set the options. Here you only set the endpoint.

Convert the response to a string and print it out

Send the request and get the response

An OMElement is just an XML element. OM means "object model".

You'll write this method yourself which will create a <concatRequest> element.

4



## Calling a web service without a client stub

- Define the makeRequest() method:

5



## Calling a web service without a client stub- makeRequest method

```
import javax.xml.namespace.QName;
import org.apache.axiom.om.OMAbstractFactory;
import org.apache.axiom.om.OMFactory;
import org.apache.axis2.addressing.EndpointReference;

public class LowLevelClient {
    ...
    private static OMElement makeRequest() {
        OMFactory factory = OMAbstractFactory.getOMFactory();
        OMElement request = factory.createOMElement(new QName(
            "http://ttdev.com/ss", "concatRequest"));
        OMElement s1 = factory.createOMElement(new QName("s1"));
        s1.setText("abc");
        OMElement s2 = factory.createOMElement(new QName("s2"));
        s2.setText("def");
        request.addChild(s1);
        request.addChild(s2);
        return request;
    }
}
```

Get the default OMFactory. You'll use it to create XML elements.

Note that the <s1> element has no namespace, just the local name.

Create <s1>

Create the <concatRequest> element

Add <s1> to <concatRequest> as a child

Set the body text to "abc"

```
<foo:concatRequest xmlns:foo="http://ttdev.com/ss">
  <s1>abc</s1>
  <s2>def</s2>
</foo:concatRequest>
```

6



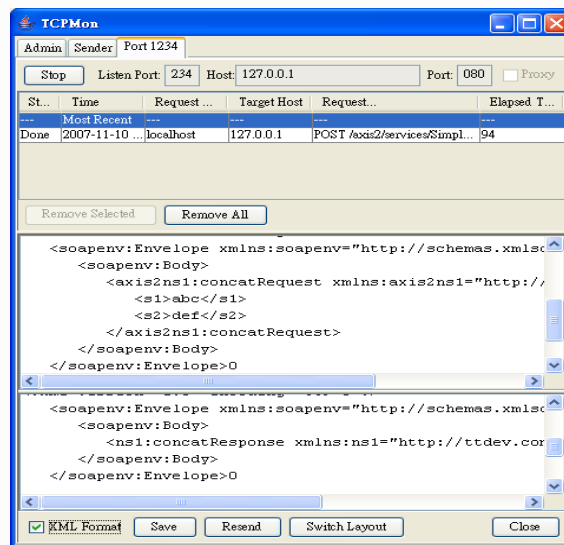
## Seeing the SOAP message

- Run the AXIOM client and output will be as shown:

7

seed  
beyond the obvious

## Seeing the SOAP message



8

seed  
beyond the obvious

## Quick Recap . . .

---

- To call a web service without using a generated stub, we may use the AXIOM interface.
- It is a lower level interface and thus is harder to use, but it provides a lot of flexibility.
- To check the SOAP messages, we can use the TCP Monitor.