@WebService OUT params via javax.xml.ws.Holder

Example webservice-holder can be browsed at https://github.com/apache/tomee/tree/master/examples/webservice-holder

With SOAP it is possible to return multiple values in a single request. This is impossible in Java as a method can only return one object.

JAX-WS solves this problem with the concept of Holders. A `javax.xml.ws.Holder` is a simple wrapper object that can be passed into the `@WebService` method as a parameter. The application sets the value of the holder during the request and the server will send the value back as an OUT parameter.

Using @WebParam and javax.xml.ws.Holder

The `@WebParam` annotation allows us to declare the `sum` and `multiply` Holders as `WebParam.Mode.OUT` parameters. As mentioned, these holders are simply empty buckets the application can fill in with data to have sent to the client. The server will pass them in uninitialized.

```
@Stateless
@WebService(
        portName = "CalculatorPort",
        serviceName = "CalculatorService",
        targetNamespace = "http://superbiz.org/wsdl",
        endpointInterface = "org.superbiz.ws.out.CalculatorWs")
public class Calculator implements CalculatorWs {

    public void sumAndMultiply(int a, int b,
                               @WebParam(name = "sum", mode = WebParam.Mode.OUT)
Holder<Integer> sum,
                               @WebParam(name = "multiply", mode = WebParam.Mode.OUT)
Holder<Integer> multiply) {
        sum.value = a + b;
        multiply.value = a * b;
    }
}
```

If the Holders were specified as `WebParam.Mode.INOUT` params, then the client could use them to send data and the application as well. The `Holder` instances would then be initialized with the data from the client request. The application could check the data before eventually overriting it with the response values.

The WSDL

The above JAX-WS `@WebService` component results in the folliwing WSDL that will be created automatically. Note the `sumAndMultiplyResponse` complext type returns two elements. These match the `@WebParam` declarations and our two `Holder<Integer>` params.

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
                  name="CalculatorService"
                  targetNamespace="http://superbiz.org/wsdl"
                  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
                  xmlns:tns="http://superbiz.org/wsdl"
                  xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <wsdl:types>
    <xsd:schema attributeFormDefault="unqualified" elementFormDefault="unqualified"
                targetNamespace="http://superbiz.org/wsdl"
                xmlns:tns="http://superbiz.org/wsdl"
                xmlns:xsd="http://www.w3.org/2001/XMLSchema">
      <xsd:element name="sumAndMultiply" type="tns:sumAndMultiply"/>
      <xsd:complexType name="sumAndMultiply">
        <xsd:sequence>
          <xsd:element name="arg0" type="xsd:int"/>
          <xsd:element name="arg1" type="xsd:int"/>
        </xsd:sequence>
      </xsd:complexType>
      <xsd:element name="sumAndMultiplyResponse" type="tns:sumAndMultiplyResponse"/>
      <xsd:complexType name="sumAndMultiplyResponse">
        <xsd:sequence>
          <xsd:element minOccurs="0" name="sum" type="xsd:int"/>
          <xsd:element minOccurs="0" name="multiply" type="xsd:int"/>
        </xsd:sequence>
      </xsd:complexType>
    </xsd:schema>
  </wsdl:types>
  <wsdl:message name="sumAndMultiplyResponse">
    <wsdl:part element="tns:sumAndMultiplyResponse" name="parameters"/>
  </wsdl:message>
  <wsdl:message name="sumAndMultiply">
    <wsdl:part element="tns:sumAndMultiply" name="parameters"/>
  </wsdl:message>
  <wsdl:portType name="CalculatorWs">
    <wsdl:operation name="sumAndMultiply">
      <wsdl:input message="tns:sumAndMultiply" name="sumAndMultiply"/>
      <wsdl:output message="tns:sumAndMultiplyResponse" name="sumAndMultiplyResponse
"/>
    </wsdl:operation>
  </wsdl:portType>
  <wsdl:binding name="CalculatorServiceSoapBinding" type="tns:CalculatorWs">
    <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
    <wsdl:operation name="sumAndMultiply">
      <soap:operation soapAction="" style="document"/>
      <wsdl:input name="sumAndMultiply">
        <soap:body use="literal"/>
      </wsdl:input>
      <wsdl:output name="sumAndMultiplyResponse">
        <soap:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>
  </wsdl:binding>
  <wsdl:service name="CalculatorService">
    <wsdl:port binding="tns:CalculatorServiceSoapBinding" name="CalculatorPort">
      <soap:address location="http://127.0.0.1:4204/Calculator?wsdl"/>
    </wsdl:port>
  </wsdl:service>
</wsdl:definitions>
```

Testing the OUT params

Here we see a JAX-WS client executing the `sumAndMultiply` operation. Two empty `Holder` instances are created and passed in as parameters. The data from the `sumAndMultiplyResponse` is placed in the `Holder` instances and is then available to the client after the operation completes.

The holders themselves are not actually sent in the request unless they are configured as INOUT params via WebParam.Mode.INOUT on `@WebParam`

```java
import org.junit.BeforeClass;
import org.junit.Test;

import javax.ejb.embeddable.EJBContainer;
import javax.xml.namespace.QName;
import javax.xml.ws.Holder;
import javax.xml.ws.Service;
import java.net.URL;
import java.util.Properties;

import static org.junit.Assert.assertEquals;
import static org.junit.Assert.assertNotNull;

public class CalculatorTest {

    @BeforeClass
    public static void setUp() throws Exception {
        Properties properties = new Properties();
        properties.setProperty("openejb.embedded.remotable", "true");
        //properties.setProperty("httpejbd.print", "true");
        //properties.setProperty("httpejbd.indent.xml", "true");
        EJBContainer.createEJBContainer(properties);
    }

    @Test
    public void outParams() throws Exception {
        final Service calculatorService = Service.create(
                new URL("http://127.0.0.1:4204/Calculator?wsdl"),
                new QName("http://superbiz.org/wsdl", "CalculatorService"));

        assertNotNull(calculatorService);

        final CalculatorWs calculator = calculatorService.getPort(CalculatorWs.class);

        final Holder<Integer> sum = new Holder<Integer>();
        final Holder<Integer> multiply = new Holder<Integer>();

        calculator.sumAndMultiply(4, 6, sum, multiply);

        assertEquals(10, (int) sum.value);
        assertEquals(24, (int) multiply.value);
    }
}
```

Inspecting the messages

The above execution results in the following SOAP message.

SOAP sumAndMultiply <small>client request</small>

```xml
<?xml version="1.0" encoding="UTF-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <ns1:sumAndMultiply xmlns:ns1="http://superbiz.org/wsdl">
      <arg0>4</arg0>
      <arg1>6</arg1>
    </ns1:sumAndMultiply>
  </soap:Body>
</soap:Envelope>
```

SOAP sumAndMultiplyResponse <small>server response</small>

```xml
<?xml version="1.0" encoding="UTF-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <ns1:sumAndMultiplyResponse xmlns:ns1="http://superbiz.org/wsdl">
      <sum>10</sum>
      <multiply>24</multiply>
    </ns1:sumAndMultiplyResponse>
  </soap:Body>
</soap:Envelope>
```