如何在Aurora框架中提供Web Service及调用Web Service

Web Service

Web Service是一种面向服务的一项技术，通过标准的Web协议提供服务，目的是保证不同平台的应用服务可以互相操作。
Aurora框架可以方便的提供Web Service服务及调用由他人发布的Web Service。

在Aurora中发布Web Service

重要概念

- SOAP：简单对象访问协议，它规定了web service传输的内容格式。
  不管用的.net、java、php或者其他技术，最终在网络上传输的都是一段特殊格式的XML内容。例子如下：

```
POST /InStock HTTP/1.1
Host: www.example.org
Content-Type: application/soap+xml; charset=utf-8
Content-Length: nnn

<?xml version="1.0"?>
<soapenv:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org
/2001/XMLSchema" xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:web="http://www.ora
cle-base.com/webservices/">
<soapenv:Header/>
<soapenv:Body>
  <!-- 这部分是自定义的-->
  <web:ws_add soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
     <int1 xsi:type="xsd:string">456</int1>
     <int2 xsi:type="xsd:string">123</int2>
  </web:ws_add>
  <!--自定义部分结束 -->
</soapenv:Body>
</soapenv:Envelope>
```

- wsdl：web service描述语言，它描述了自定义部分的格式。
  可以理解为web service的文档，不过这个文档更适合程序去阅读（比如axis2可以根据wsdl把这段xml结构封装成java对象结构，方便程序员编写java代码），而不是个人阅读。例如：

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions xmlns:tns="http://www.oracle-base.com/webservices/" xmlns:xsd="http://www.w3.org/2
001/XMLSchema" xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/" xmlns:xsi="http://www.w3.o
rg/2001/XMLSchema-instance" xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" xmlns:wsdl="http://sc
hemas.xmlsoap.org/wsdl/" xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/" targetNamespace=
"http://www.oracle-base.com/webservices/">
<wsdl:types>
    <xsd:schema targetNamespace="http://www.oracle-base.com/webservices/">
        <xsd:import namespace="http://schemas.xmlsoap.org/soap/encoding/"/>
        <xsd:import namespace="http://schemas.xmlsoap.org/wsdl/"/>
    </xsd:schema>
</wsdl:types>
<wsdl:message name="ws_addRequest">
    <wsdl:part name="int1" type="xsd:string"/>
    <wsdl:part name="int2" type="xsd:string"/>
</wsdl:message>
<wsdl:message name="ws_addResponse">
    <wsdl:part name="return" type="xsd:string"/>
</wsdl:message>
<wsdl:portType name="CalculatorPortType">
    <wsdl:operation name="ws_add">
        <wsdl:input message="tns:ws_addRequest"/>
        <wsdl:output message="tns:ws_addResponse"/>
    </wsdl:operation>
</wsdl:portType>
<wsdl:binding name="CalculatorBinding" type="tns:CalculatorPortType">
    <soap:binding style="rpc" transport="http://schemas.xmlsoap.org/soap/http"/>
```

```
    <wsdl:operation name="ws_add">
        <soap:operation soapAction="http://oracle-base.com/webservices/server.php/ws_add" style="rpc
"/>
        <wsdl:input>
            <soap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" namespace="http://w
ww.oracle-base.com/webservices/" use="encoded"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" namespace="http://w
ww.oracle-base.com/webservices/" use="encoded"/>
        </wsdl:output>
    </wsdl:operation>
</wsdl:binding>
<wsdl:service name="Calculator">
    <wsdl:port name="CalculatorPort" binding="tns:CalculatorBinding">
        <soap:address location="http://oracle-base.com/webservices/server.php"/>
    </wsdl:port>
</wsdl:service>
</wsdl:definitions>
```

推荐工具

soapui：web service测试最好用的工具。
它可以根据wsdl，自动生成soap样例，然后填入参数就可以直接发起web service请求，并获得返回结果。效果如下：

可以用http://www.oracle-base.com/webservices/server.php?wsdl这个wsdl做一下简单试用。

部署

1. 从Aurora-framework上下载最新的aurora.jar
2. 添加web service监听器
   打开[WEB-HOME]\WEB-INF\aurora.feature\service-listener.config。在

   ```
   <participant-list category="service">
   ```

   节点下，添加

   ```
   <participant class="aurora.service.ws.SOAPServiceInterpreter"/>
   ```

   结果类似如下：

   ```
   <participant-list category="service">
       <participant class="aurora.service.json.JSONServiceInterpreter" />
       <participant class="aurora.service.ws.SOAPServiceInterpreter"/>
       <participant class="aurora.plugin.export.ModelExport" />
       <participant class="aurora.plugin.export.task.TaskServiceInterpret" />
       <participant class="aurora.application.features.DataSetInit"/>

       <participant class="aurora.application.action.DoDispatch"/>

       <participant class="aurora.application.features.CachedScreenListener"/>
       <participant class="aurora.application.features.UploadInit"/>
       <participant class="aurora.service.lock.SessionLockChecker"/>
       <participant class="aurora.application.features.RequestRecorder"/>
   </participant-list>
   ```

3. 在web.xml下添加wsdl的servlet

   ```
   <!--第一段-->
   <servlet>
       <servlet-name>wsdl</servlet-name>
   ```

```
    <servlet-class>aurora.service.http.WSDLServlet</servlet-class>
</servlet>

<!--第二段-->
<servlet-mapping>
    <servlet-name>wsdl</servlet-name>
    <url-pattern>/wsdl/*</url-pattern>
</servlet-mapping>
```

以上步骤配置后。可以根据bm自动生成wsdl文件。例如：可以通过http://linjinxiao-pc:8080/hec/wsdl/db.sys_notify_pkg.update_notify/update 来查看db/sys_notify_pkg/update_notify.bm这个bm自动生成的wsdl文件。

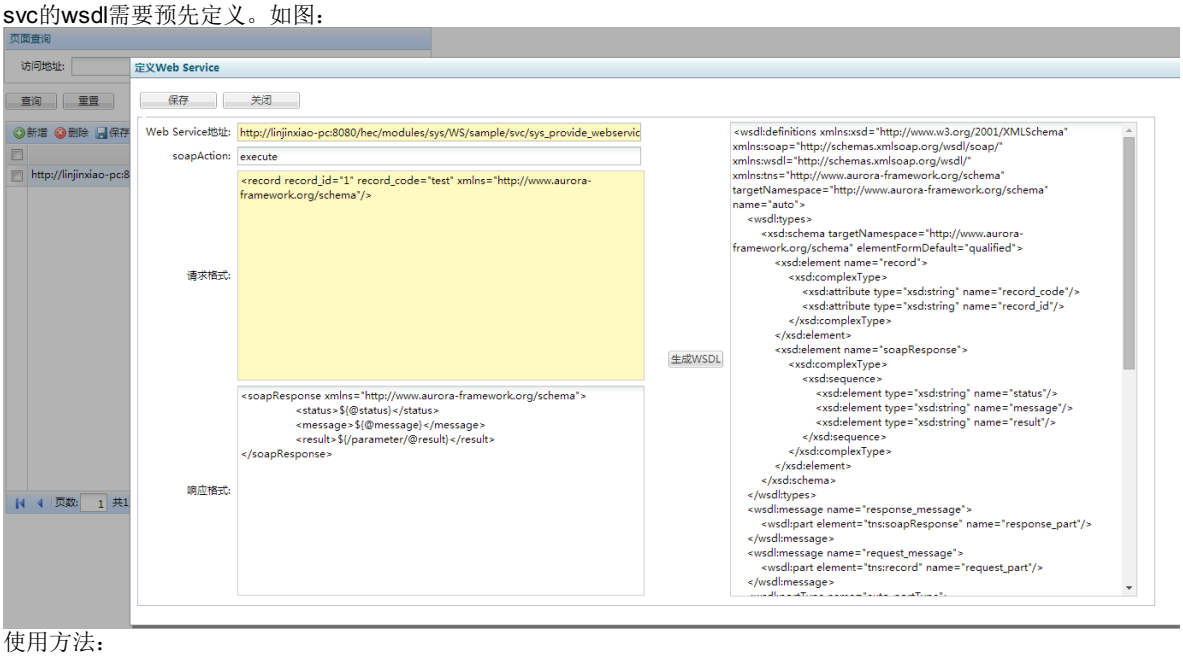
4. 定义默认返回值
   添加[WEB-HOME]\WEB-INF\aurora.feature\soap.config，例如

```
<ws:SOAPConfiguration xmlns:ws="aurora.service.ws" model="sys.WS.sys_web_services_for_query">
<soapResponse xmlns="http://www.aurora-framework.org/schema">
        <status>${@status}</status>
    <message>${@message}</message>
        <result>${/parameter/@result}</result>
</soapResponse>
</ws:SOAPConfiguration>
```

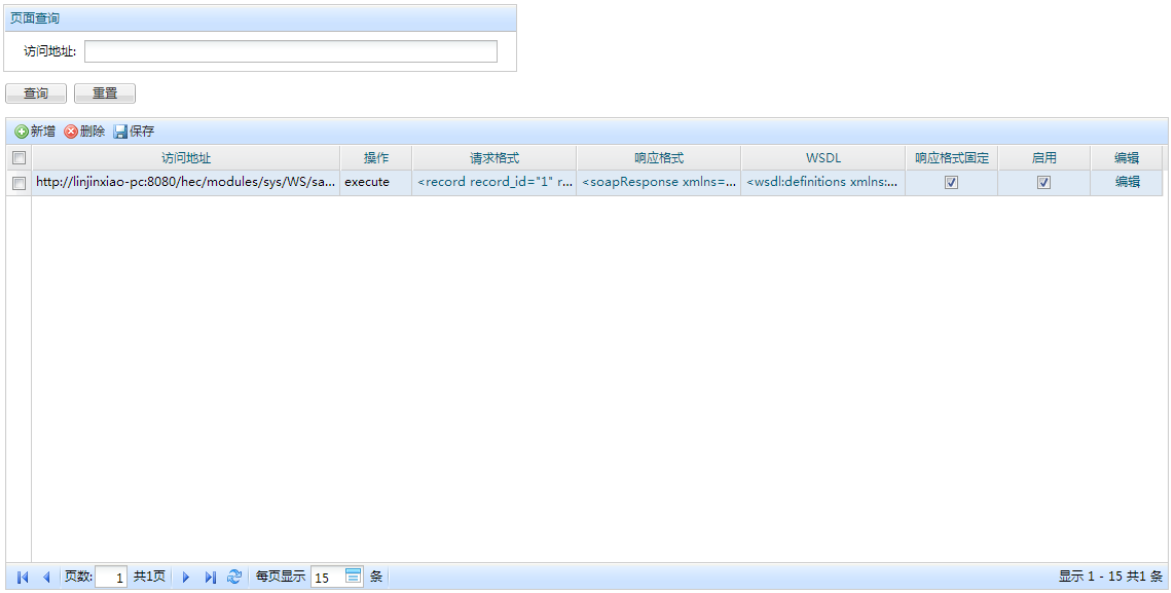
5. 访问BM的WSDL
   支持所有bm自动生成wsdl。例如：
   http://linjinxiao-pc:8080/hec/wsdl/db.sys_notify_pkg.update_notify/update
   http://linjinxiao-pc:8080/hec/wsdl/sys.SYS1070.sys_tasks/query?multi=Y

6. 访问SVC的WSDL

   - svc的wsdl需要预先定义。如图：



     使用方法：
     1. 输入真实调用的svc地址。
     2. 定义输入格式
     3. 定义输出格式，或者先任意输入xml格式，后续再改
     4. 点击按钮生成wsdl，并保存
     5. 根据wsdl，用soapui生成请求模版并调用，记录真实返回结果
     6. 再更新输出格式，生成最终的wsdl
   - 管理所有svc的wsdl。如图：



1. 响应格式固定：默认是选中。如果输入是任意多行，返回格式也要求是任意多行，即把每行的处理结果返回到行上，再统一返回。而不是仅仅显示全局处理成功或失败。那么这时候请取消“响应格式固定”，因为它可能是返回任意多行。其他情况下，仅需要返回整体成功或失败，或者返回一些少数的几个参数，可以选择“响应格式固定”。在不选中的情况下，以svc实际返回的结果为准。
2. 注意：这个功能也支持bm wsdl的定义。目前bm的返回结果，默认是采取soap.config中定义的格式。如果需要自定义特殊格式，在这里定义。
3. 定义好后，可以此svc的wsdl可以在线访问了。例如：

http://linjinxiao-pc:8080/hec/wsdl/modules/sys/WS/sample/svc/sys_provide_webservice_one_record_sample.svc

- 撰写bm/svc文件，提供给他人调用即可。

- 添加soap用户名和密码校验

修改[WEB-HOME]\WEB-INF\aurora.feature\service-procedure.config文件为

```
<?xml version="1.0" encoding="UTF-8"?>
<p:procedure-registry xmlns:t="aurora.application.action" xmlns:a="http://www.aurora-framework.org/application" xmlns:p="uncertain.proc" xmlns:ws="aurora.service.ws">
    <p:procedures>
        <p:procedure name="pre-service">
            <p:set field="@success" value="true"/>
            <t:session-copy/>
            <!--主要是这段逻辑-->
            <p:switch test="/request/@soapaction">
                <p:case Value="*">
                    <ws:WS-login-checker model="db.sys_ws_login_check_pkg.login_check"  modelaction="execute" field="/parameter/@return_value" value="N" message="没有权限登录"/>
                    <p:set field="/session/@user_id" value="-1"/>
                </p:case>
                <!--以上是新增的-->
                <p:case>
                    <p:switch test="@is_autocrud_service">
                        <p:case Value="true">
                            <t:bm-access-check errorMessage="没有权限访问此BM，或登录已失效"/>
                        </p:case>
                        <p:case>
                            <t:resource-access-check resultPath="/access-check/@status_code" errorMessage="没有权限访问此Screen，或登录已失效"/>
                        </p:case>
                    </p:switch>
                    <t:check-dispatch dispatchUrl="${/request/@context_path}/error_screen_unregistered.screen" field="/access-check/@status_code" message="页面没有注册" value="unregistered"/>
                    <t:check-dispatch dispatchUrl="${/request/@context_path}/error_screen_unauthorized.screen" field="/access-check/@status_code" message="没有权限访问指定的页面" value="unauthorized"/>
                    <t:check-dispatch dispatchUrl="${/request/@context_path}/error_session_expired.screen" field="/access-check/@status_code" message="登录已失效，请重新登录" value="login_required"/>
                </p:case>
            </p:switch>
            <p:switch test="/session/@lang">
                <p:case value="ZHS">
                    <a:model-execute model="sys.sys_register_nls_language_zh"/>
                </p:case>
                <p:case value="US">
                    <a:model-execute model="sys.sys_register_nls_language_us"/>
                </p:case>
            </p:switch>
```

```
        </p:procedure>

        <p:procedure name="session-destroy">
            <a:model-update model="sys.sys_expire_session"/>
        </p:procedure>
    </p:procedures>
</p:procedure-registry>
```

简单例子

我们提供一个svc，此svc把请求的数据插入到一个数据库表中。

假如场景：

- svc路径：

  http://linjinxiao-pc:8080/hec/modules/sys/WS/sample/svc/sys_provide_webservice_one_record_sample.svc

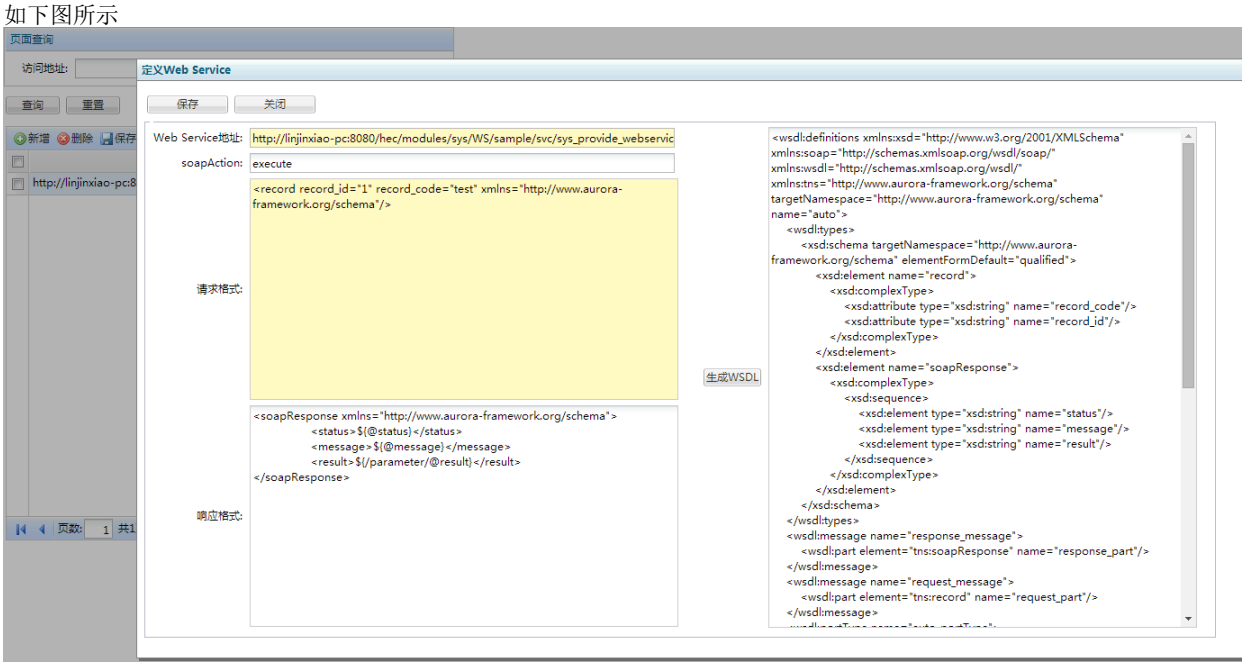
- 请求内容格式

```
<record record_id="1" record_code="test" xmlns="http://www.aurora-framework.org/schema"/>
```

- 反馈内容格式

```
<soapResponse xmlns="http://www.aurora-framework.org/schema">
        <status>${@status}</status>
    <message>${@message}</message>
        <result>${/parameter/@result}</result>
</soapResponse>
```

- 如下图所示



技术实现

1. 如上图，在系统中定义此web service的输入输出格式，并生成wsdl，保存。然后在浏览器中输入

   http://linjinxiao-pc:8080/hec/wsdl/modules/sys/WS/sample/svc/sys_provide_webservice_one_record_sample.svc

   显示如下内容：

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" xmlns:tns="http://www.aurora-framework.org/schema" targetNamespace="http://www.aurora-framework.org/schema" name="auto">
<wsdl:types>
    <xsd:schema targetNamespace="http://www.aurora-framework.org/schema" elementFormDefault="qualified">
        <xsd:element name="record">
            <xsd:complexType>
                <xsd:attribute type="xsd:string" name="record_code"/>
                <xsd:attribute type="xsd:string" name="record_id"/>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="soapResponse">
            <xsd:complexType>
```

```
                <xsd:sequence>
                    <xsd:element type="xsd:string" name="status"/>
                    <xsd:element type="xsd:string" name="message"/>
                    <xsd:element type="xsd:string" name="result"/>
                </xsd:sequence>
            </xsd:complexType>
        </xsd:element>
    </xsd:schema>
</wsdl:types>
<wsdl:message name="response_message">
    <wsdl:part element="tns:soapResponse" name="response_part"/>
</wsdl:message>
<wsdl:message name="request_message">
    <wsdl:part element="tns:record" name="request_part"/>
</wsdl:message>
<wsdl:portType name="auto_portType">
    <wsdl:operation name="execute">
        <wsdl:input message="tns:request_message"/>
        <wsdl:output message="tns:response_message"/>
    </wsdl:operation>
</wsdl:portType>
<wsdl:binding type="tns:auto_portType" name="svc.sys_provide_webservice_one_record_sample.svc_bindin
g">
    <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
    <wsdl:operation name="execute">
        <soap:operation soapAction="execute"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
</wsdl:binding>
<wsdl:service name="auto_service">
    <wsdl:port name="auto_port" binding="tns:svc.sys_provide_webservice_one_record_sample.svc_bindin
g">
        <soap:address location="http://linjinxiao-pc:8080/hec/modules/sys/WS/sample/svc/sys_provide_
webservice_one_record_sample.svc"/>
    </wsdl:port>
</wsdl:service>
</wsdl:definitions>
```

2. 新建一个对应的结果表

```
create table SYS_WEBSERVICE_SAMPLE
(
  RECORD_ID   NUMBER,
  RECORD_CODE VARCHAR2(100),
  RECORD_DATE DATE
)
```

3. 建立一个此表的bm，sys.WS.sample.sys_webservice_sample

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
    $Author: linjinxiao
$Date: 2013-6-8 上午10:46:48
    $Revision: 1.0
$Purpose:
-->
<bm:model xmlns:bm="http://www.aurora-framework.org/schema/bm" alias="t1" baseTable="SYS_WEBSERVICE_
SAMPLE">
    <bm:fields>
        <bm:field name="record_id" databaseType="NUMBER" datatype="java.lang.Long" physicalName="REC
ORD_ID" prompt="SYS_WEBSERVICE_SAMPLE.RECORD_ID"/>
        <bm:field name="record_code" databaseType="VARCHAR2" datatype="java.lang.String" physicalNam
e="RECORD_CODE" prompt="SYS_WEBSERVICE_SAMPLE.RECORD_CODE"/>
        <bm:field name="record_date" databaseType="DATE" datatype="java.util.Date" physicalName="REC
ORD_DATE" insertExpression="(select sysdate from dual)" prompt="SYS_WEBSERVICE_SAMPLE.RECORD_DATE"/>
    </bm:fields>
</bm:model>
```

4. 新建modules/sys/WS/sample/svc/

5. 文件，并输入以下内容

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
$Author: linjinxiao
$Date: 2012-12-27 上午11:07:34
```

```
$Revision: 1.0
$Purpose:
-->
<a:service xmlns:a="http://www.aurora-framework.org/application" xmlns:p="uncertain.proc" trace="tru
e">
<a:init-procedure>
    <a:model-insert model="sys.WS.sample.sys_webservice_sample"/>
    <!--
    构造 返回格式，跟错误返回格式保持一致
    <soapResponse xmlns="http://www.aurora-framework.org/schema" success="true">
        <message>OK</message>
        <success>Y</success>
    </soapResponse>
    -->
    <p:set field="/soapResponse/@success" value="Y"/>
    <p:set field="/soapResponse/@message" value="OK"/>
    <p:set field="/soapResponse/@result"  sourceField="/parameter/@record_date"/>
    <p:echo/>
    <p:method-invoke className="uncertain.composite.CompositeUtil" methodName="expand">
        <p:arguments>
            <p:argument path="/soapResponse" type="uncertain.composite.CompositeMap"/>
        </p:arguments>
    </p:method-invoke>
    <p:set-element namespace="http://www.aurora-framework.org/schema" target="/soapResponse"/>
</a:init-procedure>
<a:service-output output="/soapResponse"/>
</a:service>
```

以上内容就完成了服务器的编写。

各种客户端的调用代码示例。

1. 开源框架axis2

```
import java.util.ArrayList;
import java.util.List;
import javax.xml.namespace.QName;
import org.apache.axiom.om.OMAbstractFactory;
import org.apache.axiom.om.OMElement;
import org.apache.axiom.om.OMFactory;
import org.apache.axis2.AxisFault;
import org.apache.axis2.addressing.EndpointReference;
import org.apache.axis2.client.Options;
import org.apache.axis2.client.ServiceClient;
import org.apache.commons.httpclient.Header;

import com.sun.xml.internal.messaging.saaj.util.Base64;

public class OneRecordClient {

        public static void main(String[] args) throws AxisFault {
                ServiceClient client = new ServiceClient();
                Options options = new Options();
                options.setTo(new EndpointReference("http://linjinxiao-pc:8080/hec/modules/sys/WS/sa
mple/svc/sys_provide_webservice_one_record_sample.svc"));// 修正为实际工程的URL
                addAuthorization("linjinxiao", "ok", options);
                client.setOptions(options);
                OMElement request = makeRequest();
                OMElement response = client.sendReceive(request);
                System.out.println("response:" + response.toString());
        }

        private static void addAuthorization(String userName, String password, Options options) {
                String encoded = new String(Base64.encode(new String(userName + ":" + password).getB
ytes()));
                List list = new ArrayList();
                // Create an instance of org.apache.commons.httpclient.Header
                Header header = new Header();
                header.setName("Authorization");
                header.setValue("Basic " + encoded);
                list.add(header);
                options.setProperty(org.apache.axis2.transport.http.HTTPConstants.HTTP_HEADERS, list
);
        }

        private static OMElement makeRequest() {
                OMFactory factory = OMAbstractFactory.getOMFactory();
                OMElement request = factory.createOMElement(new QName("", "parameter"));
                request.addAttribute("record_id", "1", null);
```

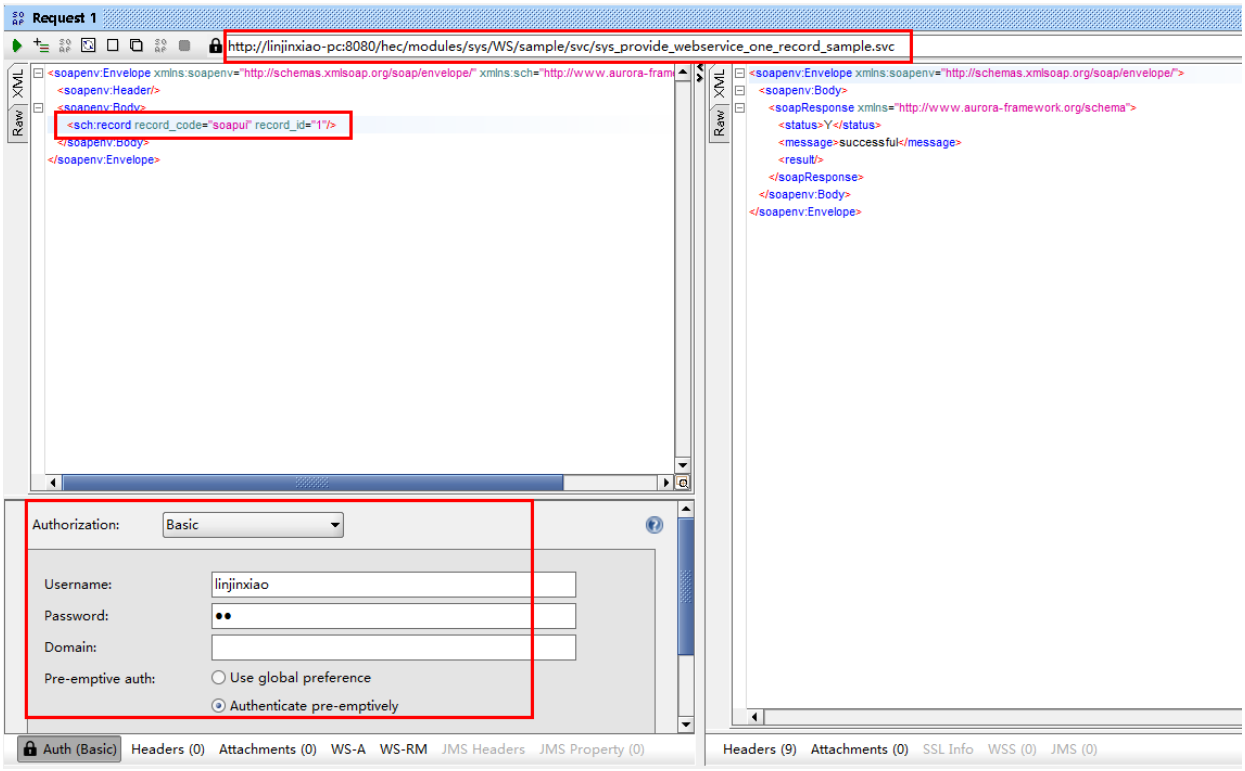
```
                request.addAttribute("record_code", "axis2", null);
                return request;
        }
}
```

查看插入数据库记录是否成功和控制台的结果是否正确。

2. soapUI客户端
   wsdl的地址是

   http://linjinxiao-pc:8080/hec/wsdl/modules/sys/WS/sample/svc/sys_provide_webservice_one_record_sample.svc

   如图所示：



3. PL/SQL客户端调用
   先添加公共的函数，

   https://dl.dropboxusercontent.com/s/5buc6i74v2o5e7k/SYS_WEBSERVICE_UTIL_PKG.pck

   然后再写调用方法

```
create or replace procedure sample_invoke_one_record(po_request_record_id out number,
                                                      po_response_xml      out sys.xmltype,
                                                      po_error_message     out varchar2) is
  v_soap_body_content varchar2(500) := '<record xmlns="http://www.aurora-framework.org/schema" record_id="1" record_code="plsql"/>';
  v_webservice_url    varchar2(200) := 'http://linjinxiao-pc:8080/hec/modules/sys/WS/sample/svc/sys_provide_webservice_one_record_sample.svc';
begin
  sys_webservice_util_pkg.invoke_webservie(p_webservice_url     => v_webservice_url,
                                           p_user_name          => 'linjinxiao',
                                           p_password           => 'ok',
                                           p_soap_action        => 'execute',
                                           p_request_str        => v_soap_body_content,
                                           po_request_record_id => po_request_record_id,
                                           po_response_xml      => po_response_xml,
                                           po_error_message     => po_error_message);
  dbms_output.put_line(po_response_xml.getclobval());

end sample_invoke_one_record;
```

复杂例子

这个例子处理有头行结构的例子。

假如场景：

- svc路径：

http://linjinxiao-pc:8080/hec/modules/sys/WS/sample/svc/sys_provide_webservice_one_record_sample.svc

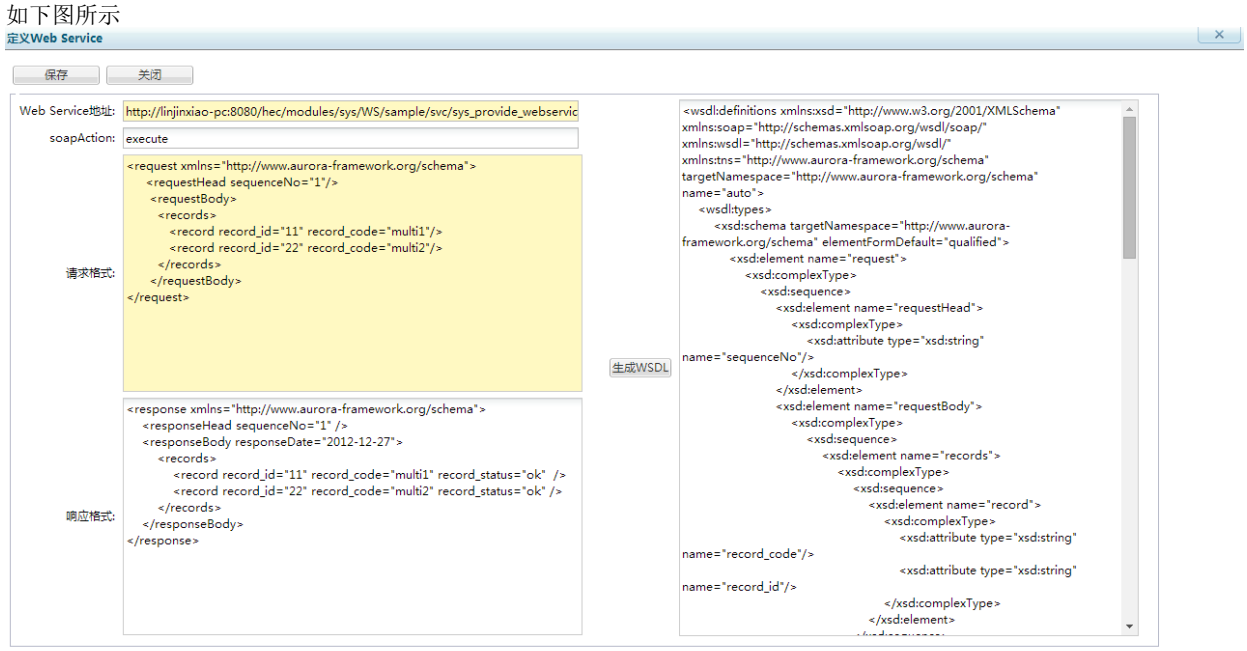
- 请求内容格式

```
<request xmlns="http://www.aurora-framework.org/schema">
     <requestHead sequenceNo="1"/>
      <requestBody>
        <records>
           <record record_id="11" record_code="multi1"/>
           <record record_id="22" record_code="multi2"/>
        </records>
      </requestBody>
</request>
```

- 反馈内容格式

```
<response xmlns="http://www.aurora-framework.org/schema">
    <responseHead sequenceNo="1" />
    <responseBody responseDate="2012-12-27">
        <records>
            <record record_id="11" record_code="multi1" record_status="ok"  />
            <record record_id="22" record_code="multi2" record_status="ok" />
        </records>
    </responseBody>
</response>
```

- 如下图所示



技术实现

1. 如上图，在系统中定义此web service的输入输出格式，并生成wsdl，保存。然后在浏览器中输入

   http://linjinxiao-pc:8080/hec/wsdl/modules/sys/WS/sample/svc/sys_provide_webservice_multi_records_sample.svc

   显示如下内容：

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" xmlns:tns="http://www.aurora-framework.org/schema" targetNamespace="http://www.aurora-framework.org/schema" name="auto">
<wsdl:types>
    <xsd:schema targetNamespace="http://www.aurora-framework.org/schema" elementFormDefault="qualified">
        <xsd:element name="request">
            <xsd:complexType>
                <xsd:sequence>
                    <xsd:element name="requestHead">
                        <xsd:complexType>
                            <xsd:attribute type="xsd:string" name="sequenceNo"/>
                        </xsd:complexType>
                    </xsd:element>
                    <xsd:element name="requestBody">
                        <xsd:complexType>
                            <xsd:sequence>
                                <xsd:element name="records">
```

```
                                    <xsd:complexType>
                                        <xsd:sequence>
                                            <xsd:element name="record">
                                                <xsd:complexType>
                                                    <xsd:attribute type="xsd:string" name="record_co
de"/>
                                                    <xsd:attribute type="xsd:string" name="record_id
"/>
                                                </xsd:complexType>
                                            </xsd:element>
                                        </xsd:sequence>
                                    </xsd:complexType>
                                </xsd:element>
                            </xsd:sequence>
                        </xsd:complexType>
                    </xsd:element>
                </xsd:sequence>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="response">
            <xsd:complexType>
                <xsd:sequence>
                    <xsd:element name="responseHead">
                        <xsd:complexType>
                            <xsd:attribute type="xsd:string" name="sequenceNo"/>
                        </xsd:complexType>
                    </xsd:element>
                    <xsd:element name="responseBody">
                        <xsd:complexType>
                            <xsd:sequence>
                                <xsd:element name="records">
                                    <xsd:complexType>
                                        <xsd:sequence>
                                            <xsd:element name="record">
                                                <xsd:complexType>
                                                    <xsd:attribute type="xsd:string" name="record_co
de"/>
                                                    <xsd:attribute type="xsd:string" name="record_id
"/>
                                                    <xsd:attribute type="xsd:string" name="record_st
atus"/>
                                                </xsd:complexType>
                                            </xsd:element>
                                        </xsd:sequence>
                                    </xsd:complexType>
                                </xsd:element>
                            </xsd:sequence>
                            <xsd:attribute type="xsd:string" name="responseDate"/>
                        </xsd:complexType>
                    </xsd:element>
                </xsd:sequence>
            </xsd:complexType>
        </xsd:element>
    </xsd:schema>
</wsdl:types>
<wsdl:message name="response_message">
    <wsdl:part element="tns:response" name="response_part"/>
</wsdl:message>
<wsdl:message name="request_message">
    <wsdl:part element="tns:request" name="request_part"/>
</wsdl:message>
<wsdl:portType name="auto_portType">
    <wsdl:operation name="execute">
        <wsdl:input message="tns:request_message"/>
        <wsdl:output message="tns:response_message"/>
    </wsdl:operation>
</wsdl:portType>
<wsdl:binding type="tns:auto_portType" name="svc.sys_provide_webservice_multi_records_sample.svc_bin
ding">
    <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
    <wsdl:operation name="execute">
        <soap:operation soapAction="execute"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
</wsdl:binding>
```

```
<wsdl:service name="auto_service">
    <wsdl:port name="auto_port" binding="tns:svc.sys_provide_webservice_multi_records_sample.svc_bin
ding">
        <soap:address location="http://linjinxiao-pc:8080/hec/modules/sys/WS/sample/svc/sys_provide_
webservice_multi_records_sample.svc"/>
    </wsdl:port>
</wsdl:service>
</wsdl:definitions>
```

2. 在db.sys_webservice_util_pkg.insert_sys_webservice_sample bm中编写如下

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
$Author: linjinxiao
$Date: 2012-12-27 下午2:19:28
$Revision: 1.0
$Purpose:
-->
<bm:model xmlns:bm="http://www.aurora-framework.org/schema/bm">
<bm:operations>
    <bm:operation name="execute">
        <bm:parameters>
            <bm:parameter name="record_id" dataType="java.lang.Long" input="true" output="false"/>
            <bm:parameter name="record_code" dataType="java.lang.String" input="true" output="false"
/>
            <bm:parameter name="record_status" dataType="java.lang.String" input="false" output="tru
e"/>
        </bm:parameters>
        <bm:update-sql><![CDATA[
            begin
                SYS_WEBSERVICE_UTIL_PKG.INSERT_SYS_WEBSERVICE_SAMPLE
                (
                    p_record_id=>${@record_id},
                p_record_code=>${@record_code},
                    p_record_status=>${@record_status}
                );
            end;]]></bm:update-sql>
    </bm:operation>
</bm:operations>
</bm:model>
```

3. 撰写modules/sys/WS/sample/svc/sys_provide_webservice_multi_records_sample.svc如下

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
$Author: linjinxiao
$Date: 2012-12-27 上午11:07:34
$Revision: 1.0
$Purpose:
-->
<a:service xmlns:a="http://www.aurora-framework.org/application" xmlns:p="uncertain.proc" trace="tru
e">
<a:init-procedure>
    <!-- 对数据进行循环操作-->
    <batch-apply sourcepath="/parameter/requestBody/records">
        <a:model-execute model="db.sys_webservice_util_pkg.insert_sys_webservice_sample"/>
    </batch-apply>
    <p:set-element name="soapResponse" namespace="http://www.aurora-framework.org/schema" target="/p
arameter"/>
    <!--更改节点的名称和namespace-->
    <p:set-element name="responseHead" target="/soapResponse/requestHead"/>
    <p:set-element name="responseBody" target="/soapResponse/requestBody"/>
    <!-- 获得当前的时间-->
    <a:model-query fetchAll="true" fethOneRecord="true" model="sys.WS.sample.sys_query_sysdate"/>
    <p:echo/>
    <p:set field="/soapResponse/responseBody/@responseDate" sourceField="/model/record/@sysdate"/>
</a:init-procedure>
<a:service-output output="/soapResponse"/>
</a:service>
```

其中sys.WS.sample.sys_query_sysdate的bm内容是

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
$Author: linjinxiao
$Date: 2012-12-27 下午2:26:31
$Revision: 1.0
$Purpose:
-->
<bm:model xmlns:bm="http://www.aurora-framework.org/schema/bm">
```

```
<bm:operations>
    <bm:operation name="query">
        <bm:query-sql><![CDATA[select sysdate  from dual]]></bm:query-sql>
    </bm:operation>
</bm:operations>
</bm:model>
```

撰写客户端代码

1. 开源框架axis2

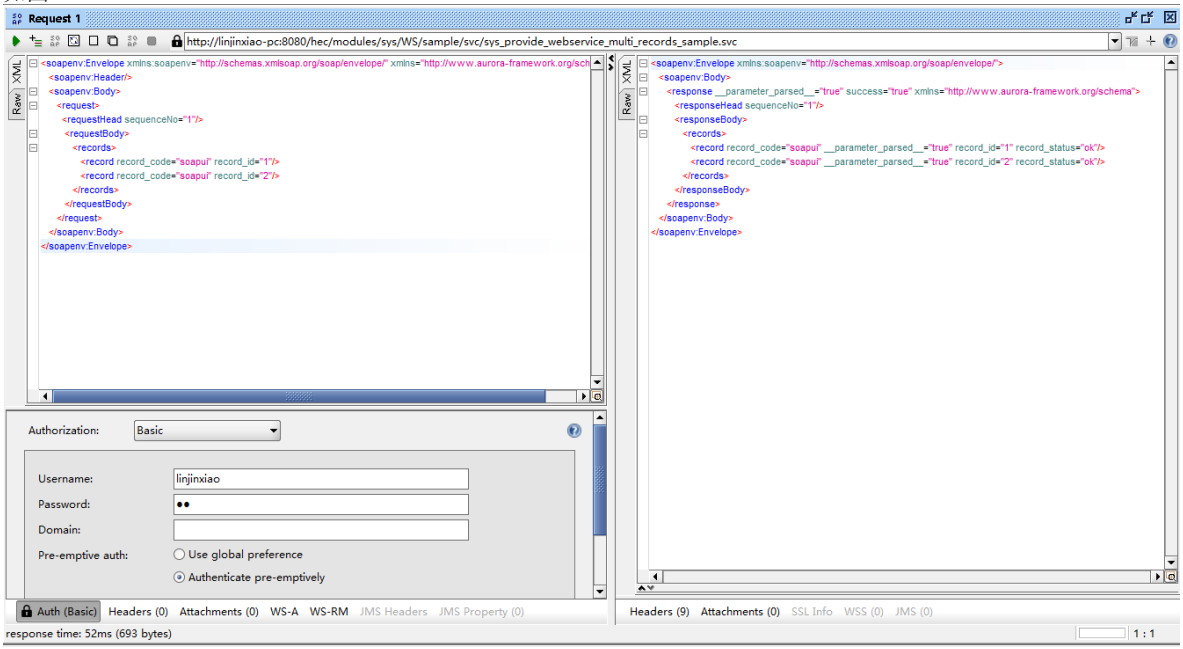
```
import java.util.ArrayList;
import java.util.List;
import javax.xml.namespace.QName;
import org.apache.axiom.om.OMAbstractFactory;
import org.apache.axiom.om.OMElement;
import org.apache.axiom.om.OMFactory;
import org.apache.axis2.AxisFault;
import org.apache.axis2.addressing.EndpointReference;
import org.apache.axis2.client.Options;
import org.apache.axis2.client.ServiceClient;
import org.apache.commons.httpclient.Header;

import com.sun.xml.internal.messaging.saaj.util.Base64;

public class MultiRecordsClient {
        public static void main(String[] args) throws AxisFault {
                ServiceClient client = new ServiceClient();
                Options options = new Options();
                options.setTo(new EndpointReference("http://linjinxiao-pc:8080/hec/modules/sys/
WS/sample/svc/sys_provide_webservice_multi_records_sample.svc"));
                addAuthorization("linjinxiao", "ok", options);
                client.setOptions(options);
                OMElement request = makeRequest();
                OMElement response = client.sendReceive(request);
                System.out.println("response:" + response.toString());
        }

        private static void addAuthorization(String userName, String password, Options options)
 {
                String encoded = new String(Base64.encode(new String(userName + ":" + password)
.getBytes()));
                List list = new ArrayList();
                // Create an instance of org.apache.commons.httpclient.Header
                Header header = new Header();
                header.setName("Authorization");
                header.setValue("Basic " + encoded);
                list.add(header);
                options.setProperty(org.apache.axis2.transport.http.HTTPConstants.HTTP_HEADERS,
 list);
        }

        private static OMElement makeRequest() {
                OMFactory factory = OMAbstractFactory.getOMFactory();
                OMElement request = factory.createOMElement(new QName("http://www.aurora-framew
ork.org/schema", "request"));
                OMElement requestHead = factory.createOMElement(new QName("requestHead"));
                OMElement requestBody = factory.createOMElement(new QName("requestBody"));
                OMElement records = factory.createOMElement(new QName("records"));
                OMElement record1 = factory.createOMElement(new QName("record"));
                record1.addAttribute("record_id", "11", null);
                record1.addAttribute("record_code", "multi1", null);
                records.addChild(record1);
                OMElement record2 = factory.createOMElement(new QName("record"));
                record2.addAttribute("record_id", "22", null);
                record2.addAttribute("record_code", "multi2", null);
                records.addChild(record2);
                requestBody.addChild(records);
                request.addChild(requestHead);
                request.addChild(requestBody);
                return request;
        }
}
```

2. soapUI客户端
   wsdl的地址是

```
http://linjinxiao-
```

pc:8080/hec/wsdl/modules/sys/WS/sample/svc/sys_provide_webservice_multi_records_sample.svc

如图



3. PL/SQL客户端调用
请看SYS_WEBSERVICE_UTIL_PKG.sample_invoke_multi_records的例子

```
    procedure sample_invoke_multi_records(po_request_record_id out number,
                                          po_response_xml      out sys.xmltype,
                                          po_error_message     out varchar2) is
    v_init_soap_body_content varchar2(500);
    v_record                 varchar2(100) := '<record record_id="11111" record_code="滫滫生僻字
" />';

    v_soap_body_xmltype sys.xmltype;

    v_aurora_server_url varchar2(30) := '10.213.208.86:8080';
    v_webservice_url    varchar2(200) := 'http://' || v_aurora_server_url ||
                                         '/hec2dev/modules/sys/test/sys_provide_webservice_mult
i_records_sample.svc';

  begin
    v_init_soap_body_content := '<af:insertRequesttype xmlns:af="http://www.aurora-framework.or
g/schema">
        <requestHead sequenceNo="1"/>
        <requestBody>
           <records>
              <record record_id="1111" record_code="中文" />
              <record record_id="2222" record_code="滫滫生僻字" />
           </records>
        </requestBody>
      </af:insertRequesttype>';
    v_soap_body_xmltype      := xmltype(v_init_soap_body_content);

    for i in 1 .. 10 loop
      v_record            := '<record record_id="' || i ||
                             '" record_code="滫滫生僻字" />';
      v_soap_body_xmltype := v_soap_body_xmltype.appendChildXML('/af:insertRequesttype/requestB
ody/records',
                                                                xmltype(v_record),
                                                                'xmlns:af="http://www.aurora-fr
amework.org/schema"');
    end loop;

    invoke_webservie(p_webservice_url     => v_webservice_url,
                     p_user_name          => 'linjinxiao',
                     p_password           => 'ok',
                     p_soap_action        => 'insert',
                     p_soap_header        => '',
                     p_request_xml        => v_soap_body_xmltype,
                     p_db_9i_version_flag => 'N',
                     po_request_record_id => po_request_record_id,
                     po_response_xml      => po_response_xml,
                     po_error_message     => po_error_message);
```

```
    dbms_output.put_line(po_response_xml.getclobval());
end sample_invoke_multi_records;
```

用server-js实现web service的例子

```
<?xml version="1.0" encoding="UTF-8"?>
<a:service xmlns:at="aurora.plugin.script" xmlns:a="http://www.aurora-framework.org/application" xmlns:p="uncertain.proc" trace="true">
    <a:init-procedure>
        <at:server-script><![CDATA[

                var context = $ctx.getData();
                context.putString('soapResponseFullControl','Y');

                importPackage(Packages.uncertain.composite);
                var headerCtx = context.getObject('/parameter/Header');
                var envHeader = context.getObject('/parameter/Header').clone();
                var envBody  = context.getObject('/parameter/Body').clone();
                context.getChilds().clear();

                var envelope = new CompositeMap('soapenv','http://schemas.xmlsoap.org/soap/envelope/','Envelope').getData();
                envelope.addChild(envHeader);
                envelope.addChild(envBody);
                context.addChild(envelope);

                var as400Body = context.getObject('/Envelope/Body/AS400/AS400Body');
                as400Body.getChilds().clear();

                var errorCode = new CompositeMap('gateway','http://www.agree.com.cn/GDBGateway','field').getData();
                errorCode.setText('0000');
                errorCode.put('name','errorCode');

                var errorMsg = new CompositeMap('gateway','http://www.agree.com.cn/GDBGateway','field').getData();
                errorMsg.put('name','errorMsg');

                as400Body.addChild(errorCode);
                as400Body.addChild(errorMsg);

        ]]></at:server-script>
        <p:echo/>
    </a:init-procedure>
    <a:service-output output="/Envelope"/>
</a:service>
```

更信息的server-js请看Aurora 服务器端脚本支持。

调用Web Service

假设服务端要求提交的格式如下：

```
 <a:soapRequest xmlns:a="http://www.aurora-framework.org/schema">
        <a:param1>1</a:param1>
        <a:param2>code</a:param2>
 </a:soapRequest>
```

返回的格式如下：

```
<a:soapResponse xmlns:a="http://www.aurora-framework.org/schema">
        <a:records>
                <a:record>
                        <a:record_id>1111</a:record_id>
                        <a:record_code>axis2</a:record_code>
                </a:record>
                <a:record>
                        <a:record_id >2222</a:record_id>
                        <a:record_code>soapUI</a:record_code>
                </a:record>
        </a:records>
</a:soapResponse>
```

1. 模拟第三方发布服务，以axis2为例
    - 下载Axis2
    - 把aurora_call Web Service部署到[AXIS2_HOME]/repository/services下，然后启动服务axis2server.bat/axis2server.sh
    - wsdl文件如下：

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions name="simple" targetNamespace="http://www.aurora-framework.org/schema" xmlns:
```

```
wsdl="http://schemas.xmlsoap.org/wsdl/" xmlns:tns="http://www.aurora-framework.org/schema" xmln
s:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/">
<wsdl:types>
<xsd:schema targetNamespace="http://www.aurora-framework.org/schema" elementFormDefault="qualif
ied">
  <xsd:element name="soapRequest">
    <xsd:complexType>
      <xsd:sequence>
            <xsd:element name="param1" type="xsd:string"/>
            <xsd:element name="param2" type="xsd:string"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
  <xsd:element name="soapResponse">
    <xsd:complexType>
            <xsd:sequence>
                    <xsd:element name="records" maxOccurs="1" minOccurs="1">
                            <xsd:complexType>
                                    <xsd:sequence>
                                            <xsd:element name="record" maxOccurs="unbounded" mi
nOccurs="1">
                                                    <xsd:complexType>

                                                            <xsd:sequence>
                                                                    <xsd:element name="record_i
d"
                                                                            type="xsd:string">
                                                                    </xsd:element>
                                                                    <xsd:element name="record_c
ode"
                                                                            type="xsd:string">
                                                                    </xsd:element>
                                                            </xsd:sequence>
                                                    </xsd:complexType>
                                            </xsd:element>
                                    </xsd:sequence>
                            </xsd:complexType>
                    </xsd:element>
            </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
</xsd:schema>
</wsdl:types>
<wsdl:message name="soapRequest">
<wsdl:part name="parameters" element="tns:soapRequest">
</wsdl:part>
</wsdl:message>
<wsdl:message name="soapResponse">
<wsdl:part name="parameters" element="tns:soapResponse">
</wsdl:part>
</wsdl:message>
<wsdl:portType name="auroraCallportType">
<wsdl:operation name="auroraCall">
  <wsdl:input message="tns:soapRequest">
</wsdl:input>
  <wsdl:output message="tns:soapResponse">
</wsdl:output>
</wsdl:operation>
</wsdl:portType>
<wsdl:binding name="auroraCallbinding" type="tns:auroraCallportType">
<soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
<wsdl:operation name="auroraCall">
  <soap:operation soapAction="http://www.aurora-framework.org/schema/auroraCall"/>
  <wsdl:input>
    <soap:body use="literal"/>
  </wsdl:input>
  <wsdl:output>
    <soap:body use="literal"/>
  </wsdl:output>
</wsdl:operation>
</wsdl:binding>
<wsdl:service name="auroraCall">
<wsdl:port name="auroraCallSOAP" binding="tns:auroraCallbinding">
  <soap:address location="http://localhost:8080/axis2/services/auroraCall"/>
</wsdl:port>
</wsdl:service>
</wsdl:definitions>
```

2. 编写客户端代码

- 编写svc文件

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
    $Author: linjinxiao
    $Date: 2012-12-27 上午11:07:34
    $Revision: 1.0
    $Purpose:
-->
<a:service xmlns:a="http://www.aurora-framework.org/application" xmlns:as="aurora.service" xmln
s:p="uncertain.proc" trace="true">
    <a:init-procedure>
        <!-- 构建符合的请求的格式-->
        <p:set field="/soapRequest/@param1" value="1"/>
        <p:set field="/soapRequest/@param2" value="code"/>
        <p:set-element namespace="http://www.aurora-framework.org/schema" target="/soapRequest"
/>
        <p:method-invoke className="uncertain.composite.CompositeUtil" methodName="expand">
            <p:arguments>
                <p:argument path="/soapRequest" type="uncertain.composite.CompositeMap"/>
            </p:arguments>
        </p:method-invoke>
        <!-- 请求WebService-->
        <a:ws-invoker inputPath="/soapRequest" raiseExceptionOnError="false" returnPath="/soapR
esponse" url="http://localhost:8080/axis2/services/auroraCall"/>
        <!--把子节点中cdata的内容整合成父节点中的一个属性 -->
        <p:method-invoke className="uncertain.composite.CompositeUtil" methodName="collapse">
            <p:arguments>
                <p:argument path="/soapResponse" type="uncertain.composite.CompositeMap"/>
            </p:arguments>
        </p:method-invoke>
        <p:echo />
        <as:SetParameterParsed/>
        <!-- 对数据进行循环操作-->
        <batch-apply sourcepath="/soapResponse/records">
            <a:model-insert model="sys.test.sys_web_service_test"/>
        </batch-apply>
    </a:init-procedure>
    <a:service-output output="/parameter"/>
</a:service>
```

- 调用此svc，并查看日志和数据库记录是否正确。