SimpleGenericHTTPSoapClient

```
//network communication via HTTP
import java.io.*;
import java.util.*;

public class SimpleGenericHTTPSoapClient {
    //Default values used if no command line parameters are set
    private static final String DEFAULT_HOST_URL = "http://localhost:8080/examples/servlet/SimpleHTTPReceive";
    private static final String DEFAULT_DATA_FILENAME   = "./PO.xml";
    private static final String URI   = "urn:oreilly-jaws-samples";
    [...]
    public void sendSOAPMessage() {
        try {

            FileReader fr = new FileReader (m_dataFileName); // get soap body to include in the SOAP envelope

            javax.xml.parsers.DocumentBuilder xdb = org.apache.soap.util.xml.XMLParserUtils.getXMLDocBuilder();

            org.w3c.dom.Document doc = xdb.parse (new org.xml.sax.InputSource (fr));

            if (doc == null) {
                throw new org.apache.soap.SOAPException (org.apache.soap.Constants.FAULT_CODE_CLIENT,
                                                         "parsing error");
            }

            Vector headerElements = new Vector(); // create a vector for collecting the header elements

            org.w3c.dom.Element headerElement =
                doc.createElementNS(URI,"jaws:MessageHeader");// Create a header element in a namespace

            org.apache.soap.Envelope envelope = new org.apache.soap.Envelope(); //Create the SOAP envelope

            Vector bodyElements = new Vector(); // create a vector for collecting the body elements

            //obtain the top-level DOM element and place it into the vector
            bodyElements.add(doc.getDocumentElement ());

            org.apache.soap.Body body = new org.apache.soap.Body(); //Create the SOAP body element
            body.setBodyEntries(bodyElements);
            envelope.setBody(body); //Add the SOAP body element to the envelope

            // Build the Message.
            org.apache.soap.messaging.Message msg = new org.apache.soap.messaging.Message();

            msg.send (new java.net.URL(m_hostURL), URI, envelope);

            // receive response from the transport and dump it to the screen
            org.apache.soap.transport.SOAPTransport st = msg.getSOAPTransport ();
            BufferedReader br = st.receive ();
            String line = br.readLine();
            if(line == null) { System.out.println("HTTP POST was successful. \n"); }
            else  {
               while (line != null) {
                    System.out.println (line);
                    line = br.readLine();
               }
            }
        }
        catch(Exception e) { e.printStackTrace(); }
    }

    /** Main program entry point. */
    public static void main(String args[]) {
        [...]
        // Start the HTTPSoapClient
        try {
            SimpleGenericHTTPSoapClient soapClient =
                new SimpleGenericHTTPSoapClient(hostURL, dataFileName);
            soapClient.sendSOAPMessage();
        }
        catch(Exception e){System.out.println(e.getMessage());}
    }
    [...]
    }
}
```

SimpleHTTPReceive (developed as a servlet)

```
import java.io.*; import java.text.*; import java.util.*; import javax.servlet.*; import javax.servlet.http.*;

public class SimpleHTTPReceive extends HttpServlet {
    // Treat GET requests as errors.
    public void doGet(HttpServletRequest request, HttpServletResponse response)
           throws IOException, ServletException {
        System.out.println("Received GET request"); response.setStatus(HttpServletResponse.SC_BAD_REQUEST);
    }

    // Our SOAP requests are going to be received as HTTP POSTS
    public void doPost(HttpServletRequest request, HttpServletResponse response)
              throws IOException, ServletException  {
        // Traverse the HTTP headers and show them on the screen
        for(Enumeration enum = request.getHeaderNames(); enum.hasMoreElements(); ) {
                String header = (String)enum.nextElement();
                String value  = request.getHeader(header);
                System.out.println("  " + header + " = " + value);
        }
        // If there is anything in the body of the message, dump it to the screen as well
        if(request.getContentLength() > 0) {
                try{
                        java.io.BufferedReader reader = request.getReader();
                        String line = null;
                        while((line = reader.readLine()) != null) { System.out.println(line); }
                }
                catch(Exception e) { System.out.println(e); }
        }
        response.setContentType("text/xml"); // Need this to prevent Apache SOAP from gacking
    }
}
```

PurchaseOrderAcceptor (service exploiting a SOAP router within APACHE)

```
import org.apache.soap.Envelope; [...]

public class PurchaseOrderAcceptor {    [...]

  public void PurchaseOrder(Envelope requestEnvelope, SOAPContext requestContext, SOAPContext responseContext)
           throws SOAPException  {

    java.io.StringWriter writer = new java.io.StringWriter();
    org.apache.soap.Header header = requestEnvelope.getHeader();
    java.util.Vector headerEntries = header.getHeaderEntries();

    writer.write("\nHeader==>\n");
    for (java.util.Enumeration e = headerEntries.elements(); e.hasMoreElements();) {
        org.w3c.dom.Element el = (org.w3c.dom.Element)e.nextElement();
        org.apache.soap.util.xml.DOM2Writer.serializeAsXML((org.w3c.dom.Node)el, writer);


        String mustUnderstand=el.getAttribute("SOAP-ENV:mustUnderstand"); // process mustUnderstand
        writer.write("\nMustUnderstand: ");
        if (mustUnderstand!=null) writer.write(mustUnderstand + "\n"); else  writer.write("null\n");
        String tagName = el.getTagName();
        writer.write("Tag Name: " + tagName + "\n");
        if(tagName.equalsIgnoreCase("jaws:MessageHeader")) { //OK, so we don't understand our own header;
            writer.write("Unsupported header: " + tagName + "\n");  writer.write("Generating Fault...\n");
        }
    }

    org.apache.soap.Body body = requestEnvelope.getBody();
    java.util.Vector bodyEntries = body.getBodyEntries();

    writer.write("\nBody====>\n");
    for (java.util.Enumeration e = bodyEntries.elements(); e.hasMoreElements();)  {
        org.w3c.dom.Element el = (org.w3c.dom.Element)e.nextElement();
        org.apache.soap.util.xml.DOM2Writer.serializeAsXML((org.w3c.dom.Node)el, writer);
    }
    System.out.println(writer.toString());
    try  { //should really be better XML with declaration and namespaces
      responseContext.setRootPart("<PurchaseOrderResponse>Accepted</PurchaseOrderResponse>", "text/xml");
    }
    catch(Exception e) { throw new SOAPException(Constants.FAULT_CODE_SERVER, "Error writing response", e); }
  }
}
```

GetBookPrice (example of SOAP RPC-messages)

```
import java.io.*; import java.util.*;

public class GetBookPrice {

  // default values to be used if not supplied on the command line
  private static final String DEFAULT_SERVICE_URL =
    "http://services.xmethods.com:80/soap/servlet/rpcrouter";
  private static final String DEFAULT_BOOK_ISBN = "0596000685";
  private String m_serviceURL;
  private String m_bookISBN;

  public GetBookPrice (String serviceURL, String bookISBN) throws Exception   {
        //this section displays the status of the call to the service
        m_serviceURL = serviceURL;  m_bookISBN   = bookISBN;

  public static float sendSoapRPCMessage (String url, String isbn) throws Exception   {

    //Build the call.
    org.apache.soap.rpc.Call call = new org.apache.soap.rpc.Call ();

    //This service uses standard SOAP encoding
    String encodingStyleURI = org.apache.soap.Constants.NS_URI_SOAP_ENC;
    call.setEncodingStyleURI(encodingStyleURI);

    //Set the target URI
    call.setTargetObjectURI ("urn:xmethods-BNPriceCheck");

    //Set the method name to invoke
    call.setMethodName ("getPrice");

    //Create the parameter objects
    Vector params = new Vector ();
    params.addElement (new org.apache.soap.rpc.Parameter("isbn", String.class, isbn, null));

    //Set the parameters
    call.setParams (params);

    //Invoke the service
    org.apache.soap.rpc.Response resp = call.invoke (new java.net.URL(url),"");

    //Check the response
    if (resp.generatedFault ()) {
       org.apache.soap.Fault fault = resp.getFault();
       System.err.println("Generated fault: ");
       System.out.println("  Fault Code   = " + fault.getFaultCode());
       System.out.println("  Fault String = " + fault.getFaultString());
       return 0;
    } else {
       org.apache.soap.rpc.Parameter result = resp.getReturnValue ();
       Float FL = (Float) result.getValue();
       return FL.floatValue();
    }
  }

    public static void main(String args[]) {

        [...]

        try   {
            GetBookPrice soapClient = new GetBookPrice(serviceURL, bookISBN);

            // call method that will perform RPC call using supplied Service
            // url and the book ISBN number to query on
            float f = soapClient.sendSoapRPCMessage(serviceURL, bookISBN);

            // output results of RPC service call
            if (bookISBN != DEFAULT_BOOK_ISBN) {
               System.out.println( "The Barnes & Noble price for this book is " + f);
            }else { System.out.println( "The price for O'Reilly's The Java Message Service book is " + f);
            }

        } catch(Exception e) {  System.out.println(e.getMessage());  }
    }
}
```