Learning Outcomes

• After learning this chapter, you should be able to
  ▪ Build Web service and its client using NuSOAP
  ▪ Understand the task separation between third-party library and your own code
  ▪ Know the basics of hard-cording and how to make it flexible
SOAP Revisited

• We have learned SOAP message and how to create it manually or using tools like http://soapclient.com/.

• We will implement our own SOAP client codes and server codes in this chapter.

• NuSOAP: SOAP Toolkit for PHP Web Site
  • It contains a set of PHP libraries to allow developers to create, publish and call web services
SOAP Web Service and Third-party Library

• We often use third-party library for routine tasks
  ▪ Generate WSDL at provider side
  ▪ Parse WSDL at requester side
  ▪ Generate SOAP at requester side
  ▪ Parse SOAP at provider side

• Additional tasks
  ▪ Implement service at provider side
  ▪ Choose WSDL operation at requester side
  ▪ Get parameter values at requester side
# Web Service Libraries

<table>
<thead>
<tr>
<th>Name</th>
<th>Programming Language Support</th>
<th>Protocol Support</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apache Axis2</td>
<td>Java</td>
<td>SOAP, WSDL, REST</td>
</tr>
<tr>
<td>Apache CXF</td>
<td>Java</td>
<td>SOAP, WSDL, REST</td>
</tr>
<tr>
<td>Windows Communication Foundation</td>
<td>C#</td>
<td>SOAP, WSDL, REST</td>
</tr>
<tr>
<td>NuSOAP</td>
<td>PHP</td>
<td>SOAP, WSDL</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
NuSOAP

• NuSOAP at Sourceforge

• Installation instruction
  ▪ Put the zip file of the downloadable release (from SourceForge) in the www directory of your gl account. The easiest way to do this is to use the wget command. You can also scp it up to your account.
  ▪ Use the unzip command to unzip the file.
  ▪ Rename the long directory name to something shorter such as nusoap.
  ▪ It is installed! It really only required copying the files to the server.
NuSOAP Example 1

• Implement NuSOAP Client for a published service
  ▪ Hard-coded Temperature Conversion Web Service Client (replacing php to txt for source code):
    https://userpages.umbc.edu/~jianwu/is651/programs/ch10/convertTempWS.php
  ▪ Temperature Conversion Web Service Client Support User Input:
    https://userpages.umbc.edu/~jianwu/is651/programs/ch10/convertTempWS.html
Hard-coded Client

- **NuSOAP Shakespeare Client**
- **Client without htmlspecialchars()**
- The example shows WSDL is the only thing a client needs to call the service

```php
<?php
require_once('nusoap/lib/nusoap.php'); // 1. pull in nusoap library
$request = array('Request' => '100'); // 2. create an array
$wsdl="https://www.w3schools.com/xml/tempconvert.asmx?WSDL";
$soapClient = new nusoap_client($wsdl, true); // 3. create a soapClient object
$result = $soapClient->call( // 4. use the call method from the soapClient object
    'FahrenheitToCelsius', // 5. a method name from the web service
    $request, // 6. the Request tag parameter for the method defined above
    'https://www.w3schools.com/xml/'; // 7. the namespace
    'https://www.w3schools.com/xml/FahrenheitToCelsius' // 8. the soapaction
);

echo '<h2>Tempconvert Web Service Response from a PHP Client</h2>';  
echo '<h2>PHP Array Response</h2>'; 
print_r($result); // 9. php function that displays an array variable
// 10. Display the request and response soap with htmlspecialchars
echo '<h2>Request</h2>'; 
echo '<pre>'.htmlspecialchars($soapClient->request, ENT_QUOTES).'</pre>'; 

echo '<h2>Response</h2>'; 
echo '<pre>'.htmlspecialchars($soapClient->response, ENT_QUOTES).'</pre>'; 
?>
```
SOAP Messages

Request
POST /xml/tempconvert.asmx HTTP/1.0
Host: www.w3schools.com
User-Agent: NuSOAP/0.9.5 (1.123)
Content-Type: text/xml; charset=ISO-8859-1
SOAPAction: "https://www.w3schools.com/xml/FahrenheitToCelsius"
Content-Length: 440
  <FahrenheitToCelsius xmlns="https://www.w3schools.com/xml/">
    <Fahrenheit>100</Fahrenheit>
  </FahrenheitToCelsius>
</SOAP-ENV:Body></SOAP-ENV:Envelope>

Response
HTTP/1.1 200 OK
Cache-Control: private, max-age=0,public
Content-Type: text/xml; charset=utf-8
Date: Tue, 17 Apr 2018 03:22:44 GMT
Server: Microsoft-IIS/7.5 X-AspNet-Version: 4.0.30319 X-Frame-Options: SAMEORIGIN X-Powered-By: ASP.NET
Content-Length: 415
  <soap:Body>
    <FahrenheitToCelsiusResponse xmlns="https://www.w3schools.com/xml/">
      <FahrenheitToCelsiusResult>37.7777777777778</FahrenheitToCelsiusResult>
    </FahrenheitToCelsiusResponse>
  </soap:Body>
</soap:Envelope>
A More Flexible NuSOAP Request

- **NuSOAP SOAP Request (HTML calls PHP client)**

```html
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
    <title>HTML form for Temperature Conversion web service</title>
  </head>
  <body>
    <h2>ConvertTemp</h2>
    <h3>HTML Form for convertTemp Web Service</h3>
    <form action="convertTempWS2.php" method="get">
      --1. http GET method -->
      Temperature: <input type="text" size="40" name="temp" />
      --2. parameter -->
      OperationName: <select name="opName">
        <option value="FahrenheitToCelsius">FahrenheitToCelsius</option>
        <option value="CelsiusToFahrenheit">CelsiusToFahrenheit</option>
      </select>
      <input type="submit" />
    </form>
  </body>
</html>
```

It doesn’t use the previous php. Why?
NuSOAP Example 2

• Implement NuSOAP Client and Server
  - CD Store Client: [http://userpages.umbc.edu/~jianwu/is651/programs/ch10/cdc.php](http://userpages.umbc.edu/~jianwu/is651/programs/ch10/cdc.php)
NuSOAP Server for CD Store

- **NuSOAP Server**
- A webpage for the service is automatically generated
NuSOAP Service Implementation for CD Store

- **NuSOAP Server function implements the service operation**

```php
<?php
function cdStore($cd, $quant) {
    $price = array('cd1' => 12, 'cd2' => 9, 'cd3' => 11, 'cd4' => 15, 'cd5' => 20); // array for price
    $name = array('cd1' => 'Led Zeppelin', 'cd2' => 'Buddy Miller', 'cd3' => 'AC/DC', 'cd4' => 'Neko Case', 'cd5' => 'Motorhead'); // array for artists
    $stock = array('cd1' => 8, 'cd2' => 10, 'cd3' => 6, 'cd4' => 2, 'cd5' => 4); // array for inventory
    if ($quant <= $stock[$cd]) // calculate invoice
        $result = "In Stock - \$.quant." . $name[$cd] . ": $" . $quant * $price[$cd];
    else $result = "Sorry, Out of Stock."
    return array('invoice' => $result);
}
?>
```
NuSOAP Client for CD Store

- NuSOAP client calls the service (hard-coded)
- **Invoice** In Stock - 2 Led Zeppelin@$12 - Price= $24

```php
<?php
require_once( "nusoap/lib/nusoap.php" );
$ns="http://userpages.umbc.edu/~jianwu/cds";
$client = new nusoap_client('http://userpages.umbc.edu/~jianwu/is651/programs/ch10/cd.php?wsdl', true);
$params=array("cd"=>"cd1", "quant"=> 2);
$result = $client->call( "cdStore", array("parameters"=>$params), $ns);
echo "<h3>Invoice</h3>".$result['invoice'];
if ( $client->getError() ) {
    print "<h2>Soap Constructor Error:</h2><pre>".$client->getError()."</pre>";
}
?>
```
A More Real Purchase Order Scenario

**Interaction process**

1. **SOAP Response**
2. **SOAP PO**
3. **Normalized Message Router**
4. **BPEL**
5. **Web Component**
6. **Web Browser**

**BPEL workflow**

1. **Create**
2. **Verify**
3. **Approved**
4. **Out of Stock**
5. **Ship**
6. **Completed**
7. **Back-ordered**
8. **Restock & Ship**