Dynamic HTML-Pages for the Omnis WebClient

Problem

You may ask yourself why do I need dynamic web-pages when I use the modern Omnis WebClient-technology? In a remote form with subforms I can dynamically and very flexibly modify the user-dialog in my program code.

On the web-server one only needs a HTML-page (name for instance **webclient.htm**) with the following content:

```
<html>
<head>
<title>Test Omnis Web-Client</title>
</head>
<body>
<h1>Test Omnis Studio Web-Client Application</h1>
<!-- Use for Microsoft Internet Explorer -->
<!-- ActiveX object START -->
<object classid="clsid:13510606-30FA-11D2-B383-444553540000"
  width="360" height="240">
  <param name="_Version" value="65536">
  <param name="_ExtentX" value="7161">
  <param name="_ExtentY" value="7373">
  <param name="_StockProps" value="0">
  <param name="OmnisServer" value="192.168.0.8:5912">
  <param name="OmnisLibrary" value="WEBCLIENT">
  <param name="OmnisClass" value="rfTest">
  <param name="WebServerURL" value="http://192.168.0.2">
  <param name="WebServerScript" value="/cgi-bin/nph-omniscgi">
<!-- Use for Netscape, Firefox and others (Windows, Linux and Mac) -->
<!-- Netscape object START-->
<embed type="application/OMNIS-RCC-plugin" name="rcc1" width="360"
   height="240" OmnisServer="192.168.0.8:5912" OmnisLibrary="WEBCLIENT"
   OmnisClass="rfTest" WebServerUrl="http://192.168.0.2"
   WebServerScript="/cgi-bin/nph-omniscgi">
<!-- Netscape object END-->
</object>
<!-- ActiveX object END-->
<p> </p>
</body>
</html>
```

What can be the benefit of creating that HTML-page dynamically?

Actually, there exist several reasons to create that page dynamically, for instance with PHP:

1. Creation of the web page will become easier and the code will be more lean,
2. the web page can more easily be reconfigured,
3. dynamic HTTP parameters can be passed to the Omnis-application,
4. if the Omnis application server is off-line, one can get a meaningful error message.

So it is not a matter of creating the web-application itself with PHP.

Particularly item 2. gave reason for the technical support in Hamburg for the development. We frequently test Omnis web applications on different machines an then always need to modify the HTML-page. Since the code for the embedded web-client for Microsoft's Internet Explorer is different from the code fo all other browsers, all modifications (for instance height, width, IP-address and port of the Omnis server) need to be entered twice into the HTML-page above.

1. Simple Creation of the Web Page

We have solved that problem with a PHP script, well actually with two PHP scripts: One quite comprehensive class definition in PHP and a very lean PHP script which replaces the HTML page. The text marked blue in the HTML page above will simply be replaced by the following PHP code:

```
<?php
```

```
    $phplib = dirname(__FILE__) . '/../phpsub';
    set_include_path(get_include_path() . PATH_SEPARATOR . $phplib);
    require_once('OmnisWebClass.php');
    $x = new OmnisWebClass('webclient.ini');
    // Insert here Extensions for parameters and / or semaphores
    $x->display_plugin();
?>
```

Furthermore, the page must be renamed from **webclient.htm** into **webclient.php** . The web-server will then send all the code between **<?php** and **?>** to a PHP-parser and –interpreter and the result of that process will be sent to the client. The user will – when looking at the source of the web page – see no difference between the static page **webclient.htm** and the dynamic page **webclient.php**.

This framework will be the same for all pages, only the second line may need modification. Here we assume, that the web server runs on a Linux PC and the class **OmnisWebClass.php** has been copied into the directory **/srv/www/htdocs/phpsub/.** Only this line needs to be modified if you are using for instance an IIS web-server under Windows.

The application-specific properties have been moved into a separate file **webclient.ini** which must be located in the same directory as the **webclient.php**. The content of that file could be:

```
WebServerURL = http://192.168.0.2
WebServerScript = /cgi-bin/nph-omniscgi
OmnisServer = 192.168.0.8
OmnisPort = 5912
OmnisLibrary = WEBCLIENT
OmnisClass = rfTest
OmnisWidth = 360
OmnisHeight = 240
; enable the next line for a Studio 5.x app server
; OmnisUnicodeServer = 1
```

In the code line

```
$x = new OmnisWebClass('webclient.ini');
```

an instance **$x** of the class **OmnisWebClass.php** will be created and the name of the .ini-file is passed to the constructor. Then the next line

```
$x->display_plugin();
```

will create the code for displaying the plugin. To call this Internet page you would enter

```
http://192.168.0.2/test/webclient.php
```

into the address field of the browser.

2. Easy Reconfiguration

If for instance the remote form should get a different width, the according parameter in the .ini-file needs to be modified only once. This will be automatically built into the code both for the Internet-Explorer and for Netscape, Firefox and others.

While testing with the Omnis IDE, the Internet browser and the Omnis application server are running on the same PC. In this case you can omit the line **OmnisServer = ...** PHP will analyze the HTTP request and evaluate among others the client's IP address. If no IP address is provided in the .ini-file, PHP will use the client's IP address. Consequently, several developers can use the same web address from different PCs.

The parameters OmnisXXX from the .ini-file can be superseded by cgi parameters. If you enter the URI

```
http://192.168.0.2/test/webclient.php?lib=TEST&class=rfTest_2&width=200
```

into the browser, PHP will create a web-page with Omnis parameters, so that the remote form rfTest2 from the library TEST is displayed with a width of 200 pixels. All missing parameters will be taken from from the .ini-file. This is documented in detail in the file **OmnisWebClass.php**.

3. HTTP-Parameters

When analyzing the HTTP request, PHP can retrieve further parameters from the HTTP header, for instance the client's browser. This can be passed as additional parameter to the remote task in Omnis. Have a look at the Omnis Programming Manual, chapter "Deploying Your Omnis Web Application" / "Manually editing your HTML Pages". You need to add to the HTML page the following code

```
<param name="Param1" value="myParameter=myValue">
```

for the Internet-Explorer and

```
Param1="myParameter=myValue"
```

for Netscape and others. The class **OmnisWebClass.php** has built-in functionality for that, you only need, after instantiating the PHP object, immediately before the line

```
$x->display_plugin();
```

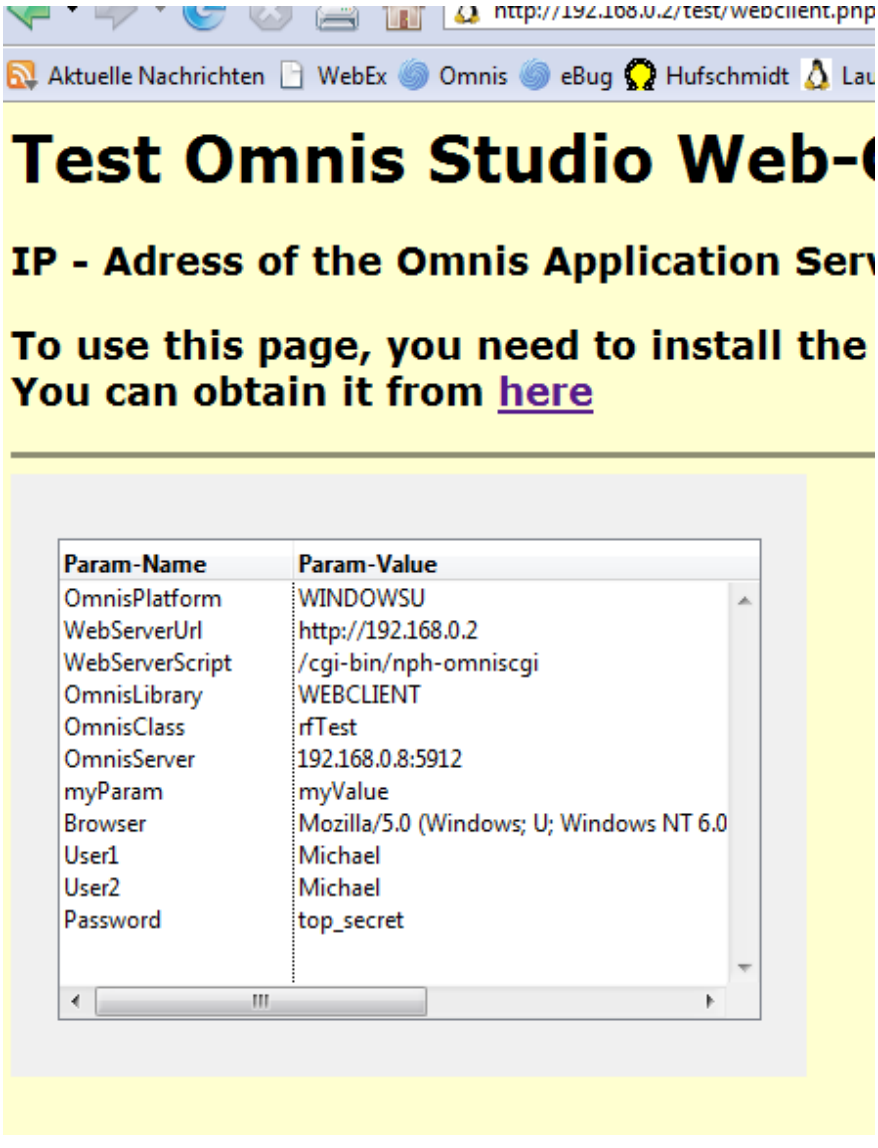
create an array in the **webclient.php**, fill it with data, and pass it to the instance:

```
$params = array ();
$params['myParam'] = 'myValue';
$params['Browser'] = $_SERVER['HTTP_USER_AGENT'];
// insert here further parameters, see below
$x->set_additional_params($params);
```

If the access to web page is protected with .htaccess, you even can retrieve the user name and password and pass that to Omnis. The following PHP code will do that:

```
if (isset($_SERVER['REMOTE_USER']))
  $params['User1'] = $_SERVER['REMOTE_USER'];
if (isset($_SERVER['PHP_AUTH_USER']))
  $params['User2'] = $_SERVER['PHP_AUTH_USER'];
if (isset($_SERVER['PHP_AUTH_PW']))
  $params['Password'] = $_SERVER['PHP_AUTH_PW'];
```

However, this will not work, if „safe mode restrictions“ are set in PHP. Here you see an example of a remote form located in a protected directory which shows the parameters of the remote task in a headed listbox:



4. Server Off-Line - Recognition

Usually, an Omnis application server will run very stable, but even then maintenance will be necessary sometimes. When the server is off-line and you call a web page with the Omnis plug-in with a browser, the browser will freeze for several minutes and after a while respond „Error reading the response from the WEB server“. The most frequent cause for that is – according to the experience of the technical support in Hamburg – a wrong port configuration of the Omnis application server.

This can be avoided by using the class **`OmnisWebClass.php`** and adding a few amendments to the Omnis library. Prerequisite is, that the Omnis application server can write into the file system of the web server. Furthermore, some „safe mode restrictions“ in PHP must be relaxed. Then, the Omnis application server can, immediately after opening the library, create a file – for instance with the name **`Semaphore_WEBCLIENT.txt`** in the directory where the **`webclient.php`** is located. Upon closing the library, this file must be deleted.

The instance of **`OmnisWebClass.php`** can check whether this file exists. If not, instead of the code for the Omnis plug-in, a message like „<h3>Omnis App. Server with Library "WEBCLIENT" is off-line!</h3>”, will be created. The exact text is configurable. To install this behavior, a line must be added to the file **`webclient.php`** which informs the instance about the name of the semaphore file.

```
$x->set_semaphore('Semaphore_WEBCLIENT.txt');
```

5. Omnis JavaScript Client

With Studio 5.2 you can create remote forms based on JavaScript running on the client browser. A typical HTML-page for the JS client will look like

```
<html>
<head>
<title>Studio 5.2 Test</title>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<!-- This makes mobile device support work -->
<meta name="viewport" content="width=device-width">
<!-- The style sheets must be present in the directory /css/ -->
<link type="text/css" href="/css/omn_dlg.css" rel="stylesheet"
  media="print,screen" />
<link type="text/css" href="/css/omn_menu.css" rel="stylesheet"
  media="print, screen" />
<link type="text/css" href="/css/smoothness/jquery-ui-1.8.15.custom.css"
  rel="stylesheet" />
<link type="text/css" href="/css/slick.grid.css" rel="stylesheet" />
<link type="text/css" href="/css/slick.columnpicker.css" rel="stylesheet"
  />
<link type="text/css" href="/css/slick.pager.css" rel="stylesheet" />
<!-- Must occur after other stylesheets e.g. jquery-ui-1.8.15.custom.css,
  as it overrides values -->
<link type="text/css" href="/css/omnis.css" rel="stylesheet"/>
<!-- Omnis Studio JavaScript client scripts, must be present in the
  directory /scripts/ -->
<script type="text/javascript" src="/scripts/omjsclnt.js"></script>
<script type="text/javascript" src="/scripts/omjqclnt.js"></script>
</head>
<body onload="jOmnis.onLoad()" onunload="jOmnis.onUnload()"
  style="margin:0;">
<h1>Test JavaScript Client Studio 5.2</h1>
<div id="omnisobject1" WebServerUrl="http://192.168.0.4/omnis_apache"
  OmnisServerAndPort="192.168.0.8:5914" OmnisLibrary="ScriptClient52_Test"
  OmnisClass="rfTest" param1="A1" param2="B1" param3="C1"
  param4="D1"></div>
</body>
</html>
```

Using the same .ini-file as for the web-client, the blue lines can be created by php. Thus a complete PHP / HTML page for the JavaScript client would be something like

```
<html>
<head>
<title>Studio 5.2 Test</title>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<?php
  $phplib = dirname(__FILE__) . '/../phpsub'
  set_include_path(get_include_path() . PATH_SEPARATOR . $phplib);
  require_once('OmnisWebClass.php');
  $x = new OmnisWebClass('ScriptClient52_Test.ini', 'rfTest');
  $params = array ("A1", "B1", "C1", "D1");
  $x->set_additional_params($params);
  $x->display_header_body_omnis_js('/css','/scripts','margin:0;');
?>
<h1>Test JavaScript Client Studio 5.2</h1>
<?php
  $x->display_client_omnis_js();
?>
</body>
</html>
```

Several different options and parameters can be set, have a look at the comments in **OmnisWebClass.php** from line 185 onwards.

Particularly you can define your own absolute path to the css and scripts directories.

Programm Package

The class **OmnisWebClass.php** consists of 650 code lines, more than one third however are comments and explanations how to use. We are not printing the code here, for our customers we have put together all necessary files into a .zip-archive „**DynamicWebPages.zip**“. In that archive you will also find this documentation, the demo library **webclient.lbs** (for Studio 4.3.1 non-unicode) and the files **webclient.ini** and **webclient.php**. If you are interested, ask the technical support in Hamburg at ge_support@omnis.net for that archive.

File:
\\SATURN\Public\SupportFAQs\Techtips\Im_Newsletter_verwendet\DynamischeWebSeiten_en.doc

Michael Hufschmidt