



Contents

- [1 Introduction](#)
- [2 Install prototype](#)
- [3 Form and AJAX\(JSON\) Demo](#)
 - ◆ [3.1 Prerequisites](#)
 - ◆ [3.2 Welcome](#)
 - ◆ [3.3 Say Hi](#)
 - ◆ [3.4 Form and AJAX](#)
 - ◆ [3.5 JSON demo](#)
- [4 Download Sample Application](#)
- [5 Related topics](#)
- [6 References](#)

Introduction

As you may know, Prototype is a famous cross-web-browser javascript library. It supports almost all current popular web browsers, such as FireFox, Safari, IE, Opera, etc. With the latest version (Prototype 1.6.0), it also supports AppleWebKit, an open source web engine provided by Apple Inc.

The Nokia Web Browser is built upon S60WebKit, a port of the open source WebKit project to the S60 platform. Nokia WRT(Web-RunTime) is based on it.

In the last section, we will learn how to implement AJAX operations using prototype library in WRT widget application.

Install prototype

Please go to [here](#) for how to install prototype javascript library.

Form and AJAX(JSON) Demo

Four cases will be introduced to demonstrate the usage of AJAX functions of Prototype library. For more information about AJAX, please go to <http://www.prototypejs.org/api/ajax> and <http://www.prototypejs.org/learn/introduction-to-ajax>

Prerequisites

There are several server-side php script files used in this sample widget application, and hosted at <http://www.ibomobi.com> (coming soon). The default value for base_url (defined in demo.js) is

Use_prototype_javascript_library:_Form_and_AJAX(JSON)_in_WRT_application

["http://www.ibomobi.com/demo/symbian/wrt/"](http://www.ibomobi.com/demo/symbian/wrt/)

You may want to setup the scripts in your own web server. Please follow the steps below as you need:

- Create a folder named "ajax" under an accessible url of your web server.
- Copy or upload the content of folder "phpscript" at the root folder of this example into that folder just created.
- Open "demo.js" file to change base url, a variable "base_url". Let it point to your base url.

Note: Do not include "ajax" folder in base url.

Welcome

Tab1, "Welcome", shows how to make an AJAX request and use the returned data to update the container identified by a div id -- "welcome". After the "Welcome" button is clicked, ajaxUpdater will be executed. It submits a request to 'ajax/welcome.php' and receives the response text, and then put the text into the container. The received text will be displayed as follows:

```
Welcome to WRT widget world!  
It's 2008-03-27 22:28:39 now.
```

The time will be changed after that button is clicked again.

HTML code snippet from main.html (Client-Side)

```
<div id="tab1">  
<p>  
<input type="button" value="Welcome" onClick="ajaxUpdater('welcome', 'ajax/welcome.php')" />  
</p>  
<hr />  
<div id="welcome"> -- </div>  
<hr />  
// omitted  
// ...  
</div>
```

JavaScript code snippet from demo.js(Client-Side)

```
function ajaxUpdater(id, path) {  
    var this_url = base_url + path;  
    new Ajax.Updater(id, this_url, {asynchronous:true});  
}
```

Php scripts from ajax/welcome.php(Server-Side)

```
<?>  
echo "Welcome to WRT widget world!<br/>";  
echo "It's ".date('Y-m-d H:i:s')." now."  
>
```

Say Hi

In tab2, "sayHi", the input data in "hi-name" field is retrieved("Tom" at default) and sent to server side via "ajax/hi.php" in sayHello function, and the replied content will be display in "hi" container. The content will be "Hi, Tom!" at default.

HTML code snippet from main.html

```
<div class="tab" id="tab2">
  <p>
  <div>
    <label for="hi-name">Enter your name:</label>
    <input id="hi-name" name="hi-name" type="text" value="Tom" />
  </div>
  <input type="button" value="Hi" onClick="sayHello('hi', 'ajax/hi.php')" /> <br/>
</p>
<hr />
<p>
<div id="hi"> -- </div>
</p>
  // omitted...
</div>
```

Javascript code snippet from demo.js

```
<code javascript>
function sayHello(id, path){
  var this_url = base_url + path;
  var pars = 'hi-name='+escape($F('hi-name'));
  var myAjax = new Ajax.Updater(id, this_url, {method: 'get', parameters: pars});
}
```

Php Scripts from ajax/hi.php

```
<?php
  $name = htmlspecialchars($_GET['hi-name']);
  echo "<b>Hi, $name!</b>";
?>
```

Form and AJAX

In tab3, "AJAX", we will collect a login form data and make an AJAX login. The default values for username and password are "Bob" and "Secret" respectively. If it is matched, a "Welcome, <username>!" message will be displayed to indicate login ok, otherwise the message will be "username or password error!".

HTML code snippet from main.html

```
<div class="tab" id="tab3">
  <p>
  <form id="loginForm">
    UserName: <input id="username" name="username" value="Bob" /><br />
    Password: <input id="password" name="password" type="password" value="Secret" /><br/>
    <input type="button" value="Login" onClick="sendAjaxLogin('loginForm', 'loginResult');" />
  </form>
</p>
  <hr />
  <div id="loginResult"> -- Not Logged in -- </div>
```

Use_prototype_javascript_library:_Form_and_AJAX(JSON)_in_WRT_application

```
<hr />
// omitted...
</div>
```

JavaScript code snippet from demo.js

```
function sendAjaxLogin(form, id)
{
    var this_url = base_url + "ajax/ajax_login.php";
    var pars = $(form).serialize();
    alert(pars); // show parameters
    var myAjax = new Ajax.Updater(id, this_url, {method: 'get', parameters: pars});
}
```

Php scripts from ajax/ajax_login.php

```
<?PHP
if (count($_GET) == 0) {
    die ("ERROR: This page has been improperly accessed.");
}

$username = $_GET['username'];
$password = $_GET['password'];

if (strcmp($username, 'Bob') == 0 && strcmp($password, 'Secret') == 0)
    echo "Welcome, ".$username."!";
else
    echo "<b>Username or password error</b>!";
?>
```

JSON demo

In tab4, it's a JSON demo. The server-side will produce some JSON data. After JSON data received at client side, it will be formatted into a HTML table.

HTML code snippet from main.html

```
<div class="tab" id="tab4">
<p>
<input type="button" value='test JSON' onclick='test_json();' />
</p>
<hr />
<div>JSON Result: <br /><span id="tab4_result">This data will be replaced.</span></div>
<hr />
<p>

<div style="display:none;">
<textarea id="table_template">
<table border="1">
<thead><tr>#{heads}</tr></thead>
<tbody>
    #{rows}
</tbody>
</table>
</textarea>
```

Use_prototype_javascript_library:_Form_and_AJAX(JSON)_in_WRT_application

```
</div>

<div style="display:none;">
<textarea id="headcell_template">
<th>#{hname}</th>
</textarea>
</div>

<div style="display:none;">
<textarea id="row_template">
<tr><td>#{product}</td><td>#{price}</td><td>#{quantity}</td></tr>
</textarea>
</div>

// ...
</div>
```

There are three hidden textareas in the above, containing template strings used to create Template objects. As eachone's id hints, the first one is a table template, the second is head cell template, and the last is row template. we will build a HTML table using those template string instead of using a fixed string writtern in Javascript code (In "string manipulation" section, we have used a fixed string as template parameter.)

JavaScript code snippet from demo.js

```
function test_json()
{
  var this_url = base_url + "ajax/test_json.php";
  // show progress animation gif
  $('waitpic').style.display = 'block';
  new Ajax.Request(this_url,
  {
    :method
    onSuccess,
    onFailure() {
      // hide progress animation gif
      $('waitpic').style.display = 'none';
      alert('Something went wrong...');
    }
  });
}

function getResponse(transport, json)
{
  var data = transport.responseText.evalJSON(true);
  // var data = eval(transport.responseText);
  // alert(data);

  // show raw json data.
  // $('tab4_result').innerHTML = transport.responseText;

  // clear result area
  $('tab4_result').innerHTML = '';

  // build table body
  var templ = new Template($('row_template').value); // not $('row_template').innerHTML
  var bodyFormatted = "";
  //populate the list
  for (var i = 0; i < data.length; i++) {
    //
    var item = data[i][0];
```

Use_prototype_javascript_library:_Form_and_AJAX(JSON)_in_WRT_application

```
//      alert(item.product);
      bodyFormatted += templ.evaluate(item) + '\n';
    }
    // build table header
    var item0 = data[0][0];
    var keys = Object.keys(item0);
    templ = new Template($('headcell_template').value);
    var headFormatted = "";
    for (var i=0; i<keys.length; i++) {
      headFormatted += templ.evaluate({"hname": keys[i]});
    }
    // assembly table.
    var tableTemplateString = $('table_template').value;
    var pars = {"heads": headFormatted, "rows": bodyFormatted};
    $('tab4_result').innerHTML = tableTemplateString.mixin(pars);

    // hide progress animation gif
    $('waitpic').style.display = 'none';
  }
}
```

In the above, mixin function is added into String class:

```
Object.extend(String.prototype, {
  mixin: function(obj) {
    return new Template(this).evaluate(obj);
  }
});
```

Php scripts from ajax/test_json.php

```
<?PHP

// ref: http://www.pastebin.ca/611218
if (!function_exists('json_encode')){
// now its portable to php ver. < 5.2.0 where you would otherwise
// have to install the php-json c-extension first
  include_once('Services_JSON.php');
  $json = new Services_JSON();
  function json_encode($value){
    global $json;
    return $json->encode($value);
  }
  function json_decode($value){
    global $json;
    return $json->decode($value);
  }
}

$_array1[] = array(
'product' => 'WRT Internals',
'price' => 24.5,
'quantity' => 1
);
$_array2[] = array(
'product' => 'The Prototype world',
'price' => 5.44,
'quantity' => 3
);
$_array3[] = array(
'product' => 'WRK Reference',
'price' => 10.00,
```

Use_prototype_javascript_library:_Form_and_AJAX(JSON)_in_WRT_application

```
'quantity' => 4
);

header('Content-type: application/x-json');
$data = array($_array1, $_array2, $_array3);
//encode and return json data...
echo json_encode($data);

?>
```

In the above, Services_JSON.php is required to generate JSON data in PHP ver < 5.2.0.

Download Sample Application

Download sample widget of this topic: [File:PrototypeFormAndAjaxDemo.zip](#). To install it, just rename the suffix .zip to .wgz.

For the latest version, please go to <http://code.google.com/p/prototypewrt/downloads/list>

Related topics

- Use prototype javascript library : [Summary in WRT application](#)
- Use prototype javascript library : [basic operations \(Utility fuctions, etc\) in WRT application](#)
- Use prototype javascript library : [string manipulation in WRT application](#)
- Use prototype javascript library : [Object Creation in WRT application](#)
- Use prototype javascript library : [Prototype UI in WRT application](#)
- Use prototype javascript library : [Form and AJAX\(JSON\) in WRT application](#)

References

- [Prototype Homepage\[1\]](#)
- [Prototype UI Homepage\[2\]](#)
- [Sample WRT applications Download \[3\]](#)