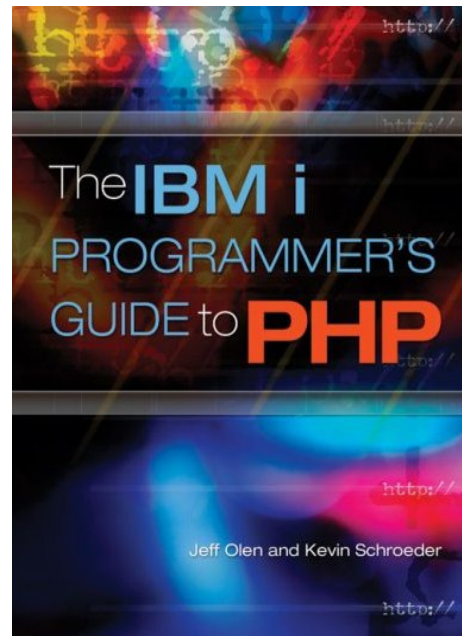


PHP and AJAX on IBM i

Who is Jeff Olen?

- IBM i developer for 20+ years
- Author of “IBM i Programmers Guide to PHP”
- V.P. and Co-founder of Olen Business Consulting, Inc.
- Member of iManifest U.S. board of directors



What you need to know

To get the most from this session you should...

- be familiar with HTML (specifically HTML forms)
- be familiar with PHP – be able to read some code. Just a bit more than a complete novice.
- have some knowledge of object oriented classes and objects.

What is AJAX?

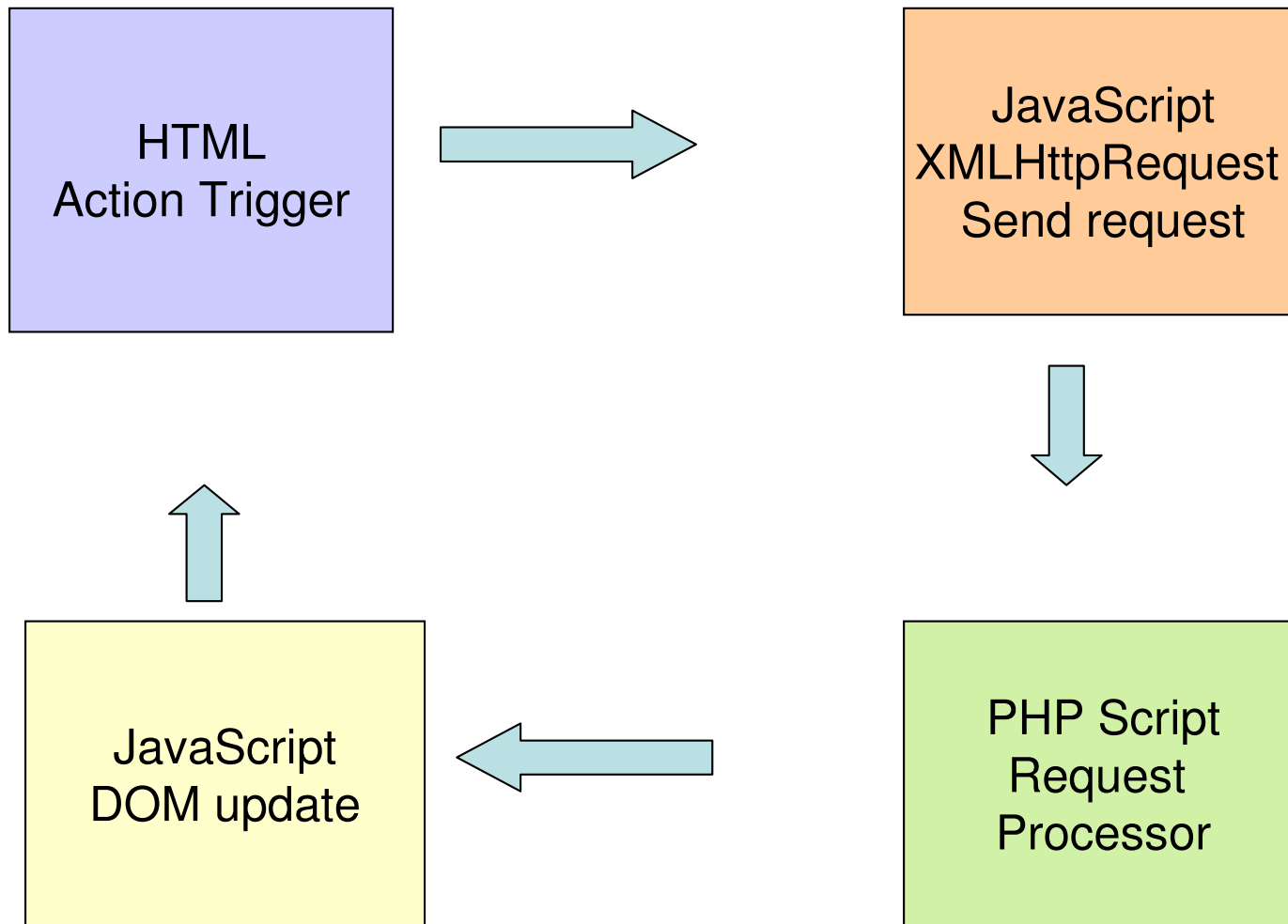
AJAX...

- is an acronym for **A**synchronous **J**avascript **A**nd **X**ML.
- can be used with or without XML.
- is a set of available technologies used together.
- can speed up your web pages.
- can make your back-end scripts/programs simpler.
- is *not* a programming language.
- is *not* owned by any software or hardware vendor.
- is *not* Java.

What makes AJAX work?

- AJAX is made possible by the XMLHttpRequest object.
- The XMLHttpRequest object is used by web browsers to communicate directly with an HTTP Server.
- XMLHttpRequest object provides an interface that allows you to send requests and receive responses from the HTTP Server without reloading the entire web page.
- XMLHttpRequest is included and supported by **all** newer web browsers.
 - Prior to Internet Explorer version 7.0, Microsoft used ActiveX objects to support this functionality.
 - All other browsers (including versions of IE starting with version 7.0) use the XMLHttpRequest object.
- There is nothing you need to install in order to start using the XMLHttpRequest object.
- More information about the XMLHttpRequest object can be found at: [www.w3schools.com/XML/xml http.asp](http://www.w3schools.com/XML/xml_http.asp)

How does it work?



Questions?

Creating an AJAX powered page

Step 1: Create the HTML form

Step 2: Add event handling

Step 3: Create the XMLHttpRequest object

Step 4: Send the request

Step 5: Process the request

Step 6: Receive/Process the response

Real world additions and improvements:

- Separate JavaScript from HTML
- Query a database
- Handle invalid entries

Step 1: Create the HTML form

```
<!-- Source: ziplookup.html -->

<html>
<head>
<title>ZIP Code lookup using AJAX</title>
</head>
<body>
<form action="post">
<p>
ZIP code:
<input type="text" size="5" name="zip" id="zip" />
</p>
City:
<input type="text" name="city" id="city" />
State:
<input type="text" size="2" name="state" id="state" />
</form>
</body>
</html>
```

Step 2: Add Event Handling

```
<!-- Source: ziplookup.html -->

<html>
<head>
<title>ZIP Code lookup using AJAX</title>
<script language="javascript" type="text/javascript">
function updateCityState() {
}
</script>
</head>
<body>
<form action="post">
<p>
ZIP code:
<input type="text" size="5" name="zip" id="zip" onblur="updateCityState();" />
</p>
City:
<input type="text" name="city" id="city" />
State:
<input type="text" size="2" name="state" id="state" />
</form>
</body>
</html>
```

Step 3: Create the XMLHttpRequest Object

```
<html>
<head>
<title>ZIP Code lookup using AJAX</title>
<script language="javascript" type="text/javascript">

var xmlhttp;

function GetXmlHttpRequest()
{
    if (window.XMLHttpRequest)
    {
        // code for IE7+, Firefox, Chrome, Opera, Safari
        return new XMLHttpRequest();
    }
    if (window.ActiveXObject)
    {
        // code for IE6, IE5
        return new ActiveXObject("Microsoft.XMLHTTP");
    }

    return null;
}

function updateCityState() {

}

</script>
...
```

Step 4: Send the Request

```
var xmlhttp;
function GetXmlHttpRequest()
{
    if (window.XMLHttpRequest)
    {
        // code for IE7+, Firefox, Chrome, Opera, Safari
        return new XMLHttpRequest();
    }
    if (window.ActiveXObject)
    {
        // code for IE6, IE5
        return new ActiveXObject("Microsoft.XMLHTTP");
    }

    return null;
}

function updateCityState() {
    xmlhttp=GetXmlHttpRequest();
    var zipCode = document.getElementById('zip').value;
    var url="rtvCityState.php?parm="+zipCode;
    url=url+"&sid="+Math.random();

    xmlhttp.onreadystatechange=handleResponse;
    xmlhttp.open("GET",url,true);
    xmlhttp.send(null);
}
...
```

Step 5: Process the Request

```
<?php  
// source name: rtvCityState.php  
echo "Twin Cactus - ".$_GET['parm'].", TX";  
?>
```

Step 6: Receive/Process the Response

```
...
function updateCityState()
{
xmlhttp=GetXmlHttpRequest();
var zipCode = document.getElementById('zip').value;
var url="rtvCityState.php?parm="+zipCode;
url=url+"&sid="+Math.random();

xmlhttp.onreadystatechange=handleResponse;
xmlhttp.open("GET",url,true);
xmlhttp.send(null);

}

function handleResponse()
{
  if (xmlhttp.readyState==4)
  {
    var results = xmlhttp.responseText.split(",");
    document.getElementById('city').value = results[0];
    document.getElementById('state').value = results[1];
  }
}
...
```

Ready states:

0 = Uninitialized - open() has not been called yet.

1 = Loading - send() has not been called yet.

2 = Loaded - send() has been called, headers and status are available.

3 = Interactive - Downloading, responseText holds the partial data.

4 = Completed - Finished with all operations.

Output

The screenshot shows a web browser window with the following elements:

- Browser Tab:** ZIP Code lookup using AJAX
- Address Bar:** http://vidafon.olen-inc.com/jeffo/Ajax_demo2/ziplookup.html
- Bookmark Bar:** <http://www.olen-inc...> and Other bookmarks
- Form Fields:**
 - ZIP code:
 - City: State:

The Windows taskbar at the bottom shows the Start button and several open applications: 2 Mic..., WinZip..., Mafia..., Micros..., chang..., ZIP Co..., and PDFs. The system tray on the right includes icons for network, volume, and other background processes, with the time displayed as 9:02 AM.

Questions?

Enhancements

Separate JavaScript source

- Cut from `<script>` to `</script>` in HTML code and paste into a separate source member called `ziplookup.js`
- Add the following code in the HTML where you removed the `<script>` tag.

```
<script type="text/javascript" src="/ziplookup.js"></script>
```
- This accomplishes two things.
 - Hides the JavaScript from the end user
 - Makes the HTML code easier to read

Enhancements: Query a database

```
<?php
$datalib = "jeffo";
$zipcode = $_GET['parm'];

$dbh = db2_connect("LEGATO3","jolen","abc123");
if ($dbh == 0) {
    echo "connection failed.<br>";
    echo db2_conn_errormsg() . "<br>";
    die();
}
// retrieve and output rate header info
$sql = "select * from ".$datalib.".zipcodes where zipcode = '".trim($zipcode)."'";
if (!$rs = db2_exec($dbh, $sql)) {
    echo "DB SQL Error ".db2_stmt_errormsg();
}

if (!$row = db2_fetch_assoc($rs)) {
    echo "DB Error ".db2_stmt_errormsg();
}

$response = trim($row['CITY']).",".trim($row['STATE']);
echo $response;
exit;
?>
```

Working versions

Original version without database lookup:

http://vidafon.olen-inc.com/jeffo/Ajax_demo2/ziplookup.html

Enhanced version:

http://vidafon.olen-inc.com/jeffo/Ajax_demo2/enhanced/ziplookup.html

Questions?