AJAX

Asynchronous Javascript And XML

Typical AJAX Flow

- Make the call
  - Gather information (possibly from HTML form)
  - Set up the URL
  - Open the connection
  - Set a callback method
  - Send the request
- Handle the response (in callback method)
  - When request.readyState == 4 and request.status == 200
  - Get the response in either text or xml
    - request.responseText or request.responseXML
  - Process the response appropriately for viewing
  - Get the objects on the page that will change
    - document.getElementById or document.getElementByName, etc.
  - Make the changes

The jqXHR Object

- Superset of the XMLHTTPRequest
- Contains responseText and responseXML properties and getResponseHeader() method
- Other functions
  - jqXHR.done(function(data,textStatus, jqXHR){})
  - jqXHR.fail(function(jqXHR, textStatus, errorThrown){})
  - jqXHR.always(function(data, textStatus, error){})

AJAX & the jqXHR Object

```
var jqxhr = $.ajax( "example.php" )
  .done(function() {
    alert( "success" );
  })
  .fail(function() {
    alert( "error" );
  })
  .always(function() {
    alert( "complete" );
  });
```

AJAX in JQuery

- $.get(url [, data] [, success(data,textStatus, jqXHR){})

```
$.get( "ajax/test.html", function( data ) {
  $( ".result" ).html( data );
  alert( "Load was performed." );
});
```

- $.post(url [, data] [, success(data,textStatus, jqXHR){})

```
$.post( "ajax/test.html", postdata, function( data ) {
  $( ".result" ).html( data );
});
```

- $.getJSON(url [, data] [, success(data,textStatus, jqXHR){})
  - Use an AJAX get request to get JSON data

Ajax

- The jQuery $.post() method loads data from the server using an HTTP POST request.
- Syntax

```
$.post(URL, {data}, function(data){…});
$.post("myScript.php", {name:txt}, function(result) {
    $("span").html(result);
  });
```

Parameter	Description
URL	Required. Specifies the url to send the request to.
data	Optional. Specifies data to send to the server along with the request.
function(data)	•Optional. Specifies a function to run if the request succeeds. *data* - contains the resulting data from the request

ajax.php

http://www.w3schools.com/jquery/ajax_post.asp

Ajax

show_product.php

```
<?php

 $id = $_POST['id'];

 mysql_connect("localhost", "omuser", "omuser") or die("Error
connecting");
 mysql_select_db("om") or die("Error selecting DB");
 $query = "SELECT * FROM items WHERE item_id = $id";
 $result = mysql_query($query);

 if (mysql_num_rows($result) == 0) {
   echo "Product ID $id not found.";
   return;
 }
 $row = mysql_fetch_array($result);

 echo "<strong>Item ID:</strong> {$row['item_id']}<br>";
 echo "<strong>Title:</strong> {$row['title']}<br>";
 echo "<strong>Artist:</strong> {$row['artist']}<br>";
 echo "<strong>Price:</strong> {$row['unit_price']}<br>";
```

Get this from the Ajax call

Ajax

ajax.php

```
$('#show').click(function(){

        var prodId = $('#id').val();

        $.post(

          "show_product.php",

          {id:prodId},

          function(result) {
             $('#detail').html(result);
          }
    );
});
```

When the button is clicked

Get the text box value

Ajax POST

Name of the PHP script

The key/value to be passed

Update the "detail" div
With the output of the PHP script

Another Example

```
function submitdata()
{

var name=document.getElementById( "name_of_user" );
var age=document.getElementById( "age_of_user" );
var course=document.getElementById( "course_of_user" );

$.ajax({
        type: 'post',
        url: 'insertdata.php',
        data: {
        user_name:name,
        user_age:age,
        user_course:course
        },
        success: function (response) {
        $('#success__para').html("You data will be saved");
        }
    });

return false;

}
```

Ajax Form Submit

```
var frm = $('#myform');
    frm.submit(function (ev) {
        $.ajax({
            type: frm.attr('method'),
            url: frm.attr('action'),
            data: frm.serialize(),
            success: function (data) {
                alert('ok');
            }
        });

        ev.preventDefault();
    });
```

Form Submition Best Way

```
$(document).ready(function () {
    $('#myform').on('submit', function(e) {
        e.preventDefault();
        $.ajax({
            url : $(this).attr('action') || window.location.pathname,
            type: "GET",
            data: $(this).serialize(),
            success: function (data) {
                $("#form_output").html(data);
            },
            error: function (jXHR, textStatus, errorThrown) {
                alert(errorThrown);
            }
        });
    });
});
```

Web Services(REST Vs SOAP)

- REST
  - Everything in REST is considered as a resource.
  - Every resource is identified by an URI.
  - Uses uniform interfaces. Resources are handled uing POST, GET, PUT, DELETE operations which are similar to Create, Read, update and Delete(CRUD) operations.

Web Services(REST Vs SOAP)

- Be stateless. Every request is an independent request.
- Each request from client to server must contain all the information necessary to understand the request.
- Communications are done via representations. E.g. XML, JSON RESTful Web Services A RESTFul web services are based on HTTP methods and the concept of REST.
- A RESTFul web service typically defines the base URI for the services, the supported MIME-types (XML, text, JSON, user-defined, ...) and the set of operations (POST, GET, PUT, DELETE) which are supported.

Web Services(REST Vs SOAP)

- SOAP
  - WSDL defines contract between client and service and is static by its nature.
  - SOAP builds an XML based protocol on top of HTTP or sometimes TCP/IP.
  - SOAP describes functions, and types of data.

Web Services(REST Vs SOAP)

- SOAP is a successor of XML-RPC and is very similar, but describes a standard way to communicate.
- Several programming languages have native support for SOAP, you typically feed it a web service URL and you can call its web service functions without the need of specific code.
- Binary data that is sent must be encoded first into a format such as base64 encoded.
- Has several protocols and technologies relating to it: WSDL, XSDs, SOAP, WS-Addressing.