AJAX and Asynchronous Design in Web Applications

IT 4203 Advanced Web Development
http://it4203.jackzheng.net

Jack G. Zheng
Fall 2016
Topics

- AJAX concepts and technical elements
- AJAX implications and impacts
- Examples
- jQuery AJAX
  - Basic methods overview
  - Error handling
  - Book search application
AJAX

- AJAX (Asynchronous JavaScript and XML) is a group of interrelated web development techniques used on the client-side to create interactive web applications.
  - Despite the name, the use of XML is not actually required, nor do the requests need to be asynchronous.

- Major AJAX elements
  - standards-based presentation using HTML and CSS;
  - dynamic display and interaction through the Document Object Model;
  - data interchange and manipulation using XML, JSON, or preformatted HTML;
  - asynchronous operations and communications using XMLHttpRequest;
  - JavaScript binding everything together.
A set of requests have been made to get JSON data from the server as I type in the search term box. Observe the “q” parameter in all URLs.

Use Chrome’s developer tools to view network communications while typing the search terms.
Model Difference

With Ajax, web applications can communicate with servers in the background without a complete page loading after every request/response cycle.

http://www.adaptivepath.com/ideas/ajax-new-approach-web-applications/
Traditional Model

The server prepares the whole page.

The client does not generate views/presentations (HTML/CSS).

With Ajax, web applications can communicate with servers in the background without a complete page loading after every request/response cycle.

The client generates views/presentations and update content (partial page) by manipulating DOM.

The server prepares partial pages (partial HTML) or just data (XML or JSON).

XHR (XMLHttpRequest)

- XMLHttpRequest (XHR) is an API available to web browser scripting languages such as JavaScript.
- XHR is used to send HTTP or HTTPS requests to a web server and load the server response data back into the script, in an asynchronous way.
- It is the AJAX “engine” or the AJAX communication layer mentioned previously.
Response Format

- The response from the asynchronous call does not have to be XML
- HTML (AHAH or AJAH)
  - Server returns the preformatted HTML/CSS as a partial page
- JSON (AJAJ, or Asynchronous JavaScript and JSON)
  - JSON has gain popularity for the past years to be used as an exchange format
JavaScript

- JS plays a central role in AJAX and ties everything together
  - Manages the AJAX communication through XHR
    - Triggers AJAX requests in page loading or user actions (clicking, mouse moving, typing, etc.)
  - Update the page dynamically through DOM
    - No complete page refresh
    - Only designated section update
AJAX Impacts

- Two major impacts of AJAX
  - User interface design and rich user experience
  - Application architecture
User Interface and Interaction

- Smoother and continuous user interaction (user experience)
  - Partial page loading; reducing unnecessary page refreshes, less web page transitioning and less wait time
  - No longer (whole) page based interaction
  - Multi-path navigation/action: less navigation interruptions.

- More like applications, building rich interaction/interface: more responsive and interactive; brings desktop experience to the web
  - Automatic content (partial page) update
  - Advanced interaction actions, such as drag and drop, mouse scroll, etc.
  - It leads to more efficient and faster communications over the Internet.
  - In-page navigation: a web application becomes less dependent on navigations between web pages. A web page becomes more like an application holder that many tasks can be completed within the page. Page transitioning is significantly decreased and navigation path is less complex.
  - Desktop GUI like dynamic interfaces are possible, such as MDI (multiple div(ision) interface) which is similar to MDI (multiple document interface) or TDI (tabbed document interface) in desktop applications.
  - Multi tasking is possible in the same page.
Typical AJAX UI and Apps

- Partial page update
  - Display data/items in a template
  - In-page navigation: multi tab/div for content: (Yahoo mail, Gmail, Google Docs, etc.)
  - Content preview: http://www.bing.com

- Application that need intensive server/client communications
  - Dynamic forms
  - Map navigation: Google maps
  - Live spelling correction

- Quick/micro user actions that should not interrupt current task
  - Quick help information,
  - Voting, server side action confirmation
  - Login
  - Auto saving: Google Docs; Quick edit, input validation, spell checking, etc.
  - Error notification
  - Form item dynamic population

- Push type applications
  - Auto notifications of news, message, status change, or updates
  - Auto refresh: http://www.google.com/finance

- Applications that take long processing time
  - File upload (slideshare.com),

- More
  - http://www.noupe.com/design/how-ajax-works.html
  - http://code.tutsplus.com/tutorials/20-excellent-ajax-effects-you-should-know--net-1090
  - http://ajaxian.com/archives/6-places-you-must-use-ajax
Web Application Architectures

- Occasional use of AJAX will greatly improve on user interface and interactions

- But it also impacted web application architectures, where the complete application/site is build upon AJAX (like SPA, in which AJAX is a key element)
  - Web app instead of web page – a page is the app holder
  - Server provides service (not page): serves data and responds to actionable request.
  - Single page app – page as the holder
  - Client MVC – UI logic and part of business logic on client.
  - Service oriented or resource oriented (data service)

- More at
Changing of C/S Interaction

Model 1: classic Web application

- Web browser
  - User interface
  - HTTP requests, HTML + CSS
  - UI generation
  - Business services, Data...
  - Application server

Model 2: AJAX Web application

- Web browser
  - User interface
  - AJAX engine
  - HTTP requests, UI fragments, Raw XML / JSON data
  - Web server
  - Business services, Data...
  - Application server

Model 3: client-side MV* Web application

- Web browser
  - User interface
  - MV* engine
  - HTTP requests, Raw JSON data
  - Business services, Data...
  - Application server

Can also be called the AJAX service model

UI and data logic move to the client

Server provides service

AJAX and Services

- AJAX changes the meaning of the “client”
  - The web app is separated into the app holder (the client) and app content
  - The initial app holder is downloaded from the server and consumes services (which provide app content) from the server

- With AJAX, the presentation logic and business logic can completely move from server to client; data access logic remain on the server
  - Business logic is expose to the client; if not desired, business logic should remain on the server

- In many ways, AJAX applications follow the REST design principles. Each XMLHttpRequest can be viewed as a REST service request, sent using GET. And the response is often in JSON, a popular response format for REST.
AJAX + (RESTful) Services

http://stage.vambenepe.com/archives/1801
# AJAX Programming Examples

- All examples can be downloaded from D2L Brightspace and live demo at [http://it4203.jackzheng.net/demo/ajax](http://it4203.jackzheng.net/demo/ajax)

- It is best to load all files to your server and see its effects. AJAX examples need server-side support.

<table>
<thead>
<tr>
<th>Files</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>2. jquery-ajax-loadhtml.html server-data-html.php</td>
<td>jQuery AJAX methods (load, get, ajax) to load a static or dynamic HTML partial page. server-data-html.php is a dynamic HTML page with PHP code.</td>
</tr>
<tr>
<td>3. jquery-ajax-loadjson.html server-data-json.php</td>
<td>jQuery AJAX methods to load JSON content returned by the server. server-data-json.php is a dynamic JSON file generated in PHP. The example also demonstrate three ways of triggering the click event.</td>
</tr>
<tr>
<td>4. jquery-ajax-post.html</td>
<td>Use the POST method and work with an external service <a href="http://jsonplaceholder.typicode.com">http://jsonplaceholder.typicode.com</a></td>
</tr>
<tr>
<td>5. jquery-ajax-errors.html server-response.php</td>
<td>Demonstrates error handling; the server-response.php page will simulate different situations by passing a parameter in URL.</td>
</tr>
<tr>
<td>6. spa-event-parameter.html</td>
<td>This example shows how to add the click event dynamically to a set of clickable elements and how to pass parameter value to the event.</td>
</tr>
</tbody>
</table>
The Simplest Classic AJAX Call

Also see [http://www.w3schools.com/ajax/tryit.asp?filename=tryajax_first](http://www.w3schools.com/ajax/tryit.asp?filename=tryajax_first)

```html
<html>
  <head>
    <script type="text/javascript">
      function UpdateDivAjax()
      {
        xmlHttp = new XMLHttpRequest();
        xmlHttp.onreadystatechange = function () {
          if (xmlHttp.readyState == 4 && xmlHttp.status == 200)
            document.getElementById('ajaxDiv').innerHTML = xmlHttp.responseText;
        };
        xmlHttp.open("GET", 'ajax-partialpage.html', true);
        xmlHttp.send();
      }
    </script>
  </head>
  <body>
    <input type="button" value="Click me" onclick="UpdateDivAjax()" />
    <div id="ajaxDiv"></div>
  </body>
</html>
```

1. Create a XMLHttpRequest object.

2. Define a callback function to complete tasks when the response is returned. See the next slide.

3. Send the request, with request method and target defined.

The target here is an partial HTML file which will be returned and inserted to the ajaxDiv division.

“onclick” event triggers the UpdateDivAjax function, which will update the “ajaxDiv” section.
Call Back Function

- Call back function is an event handler to complete tasks when an asynchronous response is received. This usually defines the partial page updates.

```javascript
xmlHttp.onreadystatechange = function () {
    if (xmlHttp.readyState == 4 && xmlHttp.status == 200) {
        document.getElementById('ajaxDiv').innerHTML = xmlHttp.responseText;
    }
};
```

This is an inline call back function, which is defined immediately. "function" is the default name for the inline call back function – don’t rename it.

If the result is successfully returned and loaded.

“responseText” contains raw text returned from the Ajax call target.

Partial page update through DOM.
AJAX Call Targets

- The AJAX call target can be set to any files or applications on the server. This can be:
  - A static file, including HTML, XML, JSON, CSV, or just pure text.
  - A dynamic web page (either HTML, XML, or JSON) generated by ASP.Net (.aspx), PHP, Servlet, etc.
  - URL parameters can be included

Examples

```javascript
xmlHttp.open("GET", 'ajax-partialpage.html', true);
xmlHttp.open("GET", 'ajax-partialpage.aspx', true);
xmlHttp.open("GET", 'ajax-partialpage.php?bookid=k123', true);
xmlHttp.open("GET", 'ajax-partialpage.xml', true);
```

- Which server? Normally the target has to be on the same server (domain) as the requesting page.
  - The *same origin policy* prevents some AJAX techniques from being used across domains
    [https://developer.mozilla.org/En/Same_origin_policy_for_JavaScript](https://developer.mozilla.org/En/Same_origin_policy_for_JavaScript)
AJAX and jQuery

- Raw AJAX handling in JS is very tedious
- jQuery provides flexible and strong support to handle AJAX interactions through a set of jQuery functions.

- Main low-level function
  - $.ajax()

- Useful higher level shorthand methods
  - .load()
  - $.get()
  - $.getJSON()
  - $.getScript()
  - $.post()

- See jQuery AJAX API reference
$.ajax()

- The $.ajax() function underlies all AJAX requests sent by jQuery.
- It is often unnecessary to directly call this function,
  - Use higher-level alternatives like $.get() and $.load() which are easier to use.
  - If full customization is required, then use $.ajax() for flexibility
- Usage and reference
  - [http://www.w3schools.com/jquery/ajax_ajax.asp](http://www.w3schools.com/jquery/ajax_ajax.asp)
Shorthand Methods

- Five main shorthand methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$.load()</td>
<td>directly load a partial page to an HTML “div”</td>
</tr>
<tr>
<td>$.get()</td>
<td>send a request using HTTP GET</td>
</tr>
<tr>
<td>$.getJSON()</td>
<td>send a request using HTTP GET; the response is expected to be in JSON format and will be parsed into a JSON object</td>
</tr>
<tr>
<td>$.post()</td>
<td>send a request using HTTP POST</td>
</tr>
<tr>
<td>$.getScript()</td>
<td>get a piece of JavaScript and insert it to the current page dynamically</td>
</tr>
</tbody>
</table>
$.getJSON()

- Load JSON-encoded data from the server using a GET HTTP request

- Basic usage

```javascript
$.getJSON( "http://server/file", function(json)
{
  // processing JSON using JavaScript here.
});
```

  Service point that returns JSON data.

  The callback function which is executed once the response is ready.

  A variable representing the JSON object that is built based on the JSON response from the server, assuming the JSON parsing is correct. An error will be thrown if the parsing fails.

- Reference
  - [http://www.w3schools.com/jquery/ajax_getjson.asp](http://www.w3schools.com/jquery/ajax_getjson.asp)
jQuery AJAX Example

```javascript
$(document).ready(function () {
    $('#button1').click(function () {
        $.getJSON('instructor.json', function (instructor) {
            $('#Instructor').html(instructor.FirstName + ' - ' + instructor.Title);
        });
    });
});
```

The JSON object from the response will be referred to in the function using this variable.

jQuery function to work with AJAX

Dynamically updates the page through DOM.
Notes on Linking/Clicking

- Avoid using normal links ("a" tag with href pointing to a real URL) to initiate an AJAX request. Because it will refresh the whole page.

- Clicking should be handled by JS (click event) to stay on the same page

- Three basic techniques to initiate an AJAX request
  - Use buttons (you may style it like links or other clickable areas)
  - Use the "a" tag but with # (<a href="#">) and handle the clicks using JS/jQuery
  - Use any element and simulate it as a link/button or a clickable area using CSS style.
  - See the example: jquery-ajax-loadjson.html
Passing Parameter Value in Events

- Normally we can pass parameter via URL for multipage applications.

- In SPA, we can pass parameters to the click event through the following two ways:
  - Use custom HTML tag attribute
  - Use text content

- Two ways to read parameter values in the event handler:
    ```javascript
    $('.booklistitem').on('click', function () {
        getBookDetails( $(this).attr("data-bookid") );
    });
    ```
  - Event target parameter [https://api.jquery.com/event.target/](https://api.jquery.com/event.target/)
    ```javascript
    $('.booklistitem2').on('click', function (event) {
        getBookDetails( $(event.target).text());
    });
    ```

- See example #6
Error Handling

- Common error situations
  - Timeout: waiting too long
  - Server error: 404 not found, 500 internal server error, etc.
  - Incorrect/invalid JSON

- Use the fail() function to handle errors.

- See the example #5 on error handling
More Resources

- jQuery AJAX references
  - [http://www.w3schools.com/jquery/jquery_ref_ajax.asp](http://www.w3schools.com/jquery/jquery_ref_ajax.asp)
  - [http://www.openajax.org](http://www.openajax.org)