PROGRAMMING FOR MOBILE DEVICES

Programming Mobile Devices

AJAX (Asynchronous Javascript and XML)

Making Web Pages Look Like Apps since 1999

- HTML was devised and developed as a ***Document*** standard
  - Its original purpose was to make scientific papers easier to publish to a community
  - Tim Berners Lee had no notion that his invention would lead to widespread e-commerce, entertainment etc.
  - Web browser developers had very different ideas
- Therefore, websites were collections of linked web pages, each of which was a static document
- AJAX was described in an article by Jesse James Garrett in February 2005
  - http://www.adaptivepath.com/ideas/ajax-new-approach-web-applications/
  - The method had been used before (by Google – Google Suggest and Microsoft – Outlook Web App). What Garrett did was to invent the name and help to de-mystify the format

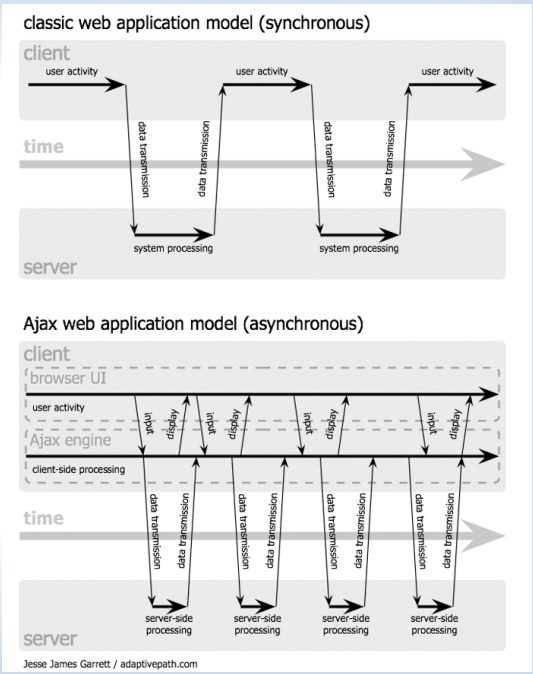
XMLHttpRequest Object

- Blame this on Microsoft
  - In 1996, Microsoft added the <iframe> tag to Internet Explorer
    - This allowed a whole HTML document to be fetched and placed within a block of an otherwise static page
    - It effectively puts one page inside another
  - In 1999, Microsoft added the XMLHTTP ActiveX control to IE
    - This allowed new content to be fetched by Javascript code
    - i.e. content was added ***asynchronously*** (not synchronized with the page the JS code was inside)
  - Other browsers didn't do ActiveX controls, but standardized on the XMLHttpRequest object, which did the same job
    - This leads to a lot of awkward code in AJAX apps, because depending on the browser you need to create either an ActiveX or an XMLHttpRequest
    - See http://www.w3schools.com/ajax/ajax_xmlhttprequest_create.asp for an example
    - Microsoft fixed this in Internet Explorer 7, so now borwsers are compatible in that respect

XML?????

- The most obvious thing to notice about AJAX in use currently, is that it rarely involves XML
  - In the early 20xx years, XML was the future of rock&roll, as far as web applications was concerned
  - Current use is more likely to involve JSON
    - It is more compact
    - It is easier for humans to read)
    - It is a simpler format, so processing is faster

AJAX in Operation



Source: http://www.adaptivepath.com/uploads/archive/images/publications/essays/ajax-fig2.

AJAX Now

- The original idea was to allow web pages to be augmented according to user-interactions
  - AJAX is now a core technology for updating a browser's document object model in applications
  - e.g. wherever a C program would call printf(), or a Visual Basic program would update a screen control (e.g. a TextBox or ListBox), AJAX apps update the DOM

AJAX at its simplest

```
<!DOCTYPE html>
<html>
<head lang="en">
    <meta charset="UTF-8">
    <title>Simple AJAX Demo</title>
</head>
<body>
    <h1 id="title"></h1>
    <script src="ajax.js"></script>
</body>
</html>
```

- AJAX Needs
  - A Target element that will be updated
  - Script to download and update the target element
  - The XHR object (shorthand for XMLHttpRequest)

- XHR Returns
  - .readyState
  - .status
  - .responseText, .responseXML
  - Other info.

```
var xhr = new XMLHttpRequest(),
    readyStates = [
        "0: request not initialized",
        "1: server connection established",
        "2: request received", "3: processing request",
        "4: request finished and response ready"
    ];
xhr.open("GET", "title.txt", true);
xhr.onreadystatechange = function() {
    console.log(readyStates[xhr.readyState]);
    if(xhr.readyState == 4 && xhr.status == 200) {
        document.getElementById("title").innerText
                                    = xhr.responseText;
    }
}
xhr.send();
```

AJAX and jQuery Mobile

- As the example shows, AJAX works with URLs to get data
- The data can be local or remote
  - e.g. the contents of a text file on the server
  - e.g. data from online databases (in the right format)
- The call can even be made to get data from within the file that makes it
  - e.g. in jQM, the readystatechange function is used to show/hide DOM elements, play animations, attach CSS styles to elements etc.
  - How else would the multi-page structure of a jQM HTML document work?
    - AJAX updates the document elements AND keeps the browser history in line

Other uses for AJAX in jQM

- Since jQM apps must also include jQuery
  - The built-in AJAX mechanism is easy to get to
  - AJAX code tends to be simpler to set up and provides more useful information
    - See http://demos.jquerymobile.com/1.3.0-rc.1/docs/demos/widgets/ajax-nav/

Needed for cross-origin

```
$.ajax({
   datatype: "jsonp",
   url: ratesURL + symbol,
   success: function(data) {
      rateList = data;
      doUpdates();
   },
   error: function(err) {
      alert("Error: " + err.message);
   }
});
```

See Lab2 for the whole example

Handling the HTTP Response

- AJAX can be used to send any type of HTTP request
  - GET, POST, PUT, DELETE, HEAD etc.
- In this module, we'll concentrate on GET
  - This is the same HTTP function that web browsers use to download HTML pages
  - The response handler (typically, an anonymous "success" function) is passed the new data from the server
  - A typical implementation will pass this data to a formatting function

Processing the AJAX response

```
function doGetRequest() {
  // The parameters for this are important.
  $.ajax({
    type: "GET",                          // The HTTP operation
    url: "http://<some-service-url>",     // The service URL
    jsonpCallback: 'handleResults',       // Function to call with results.
    contentType: "application/json",      // MIME type function expects...
    dataType: 'jsonp',                    // ...and the type of data expected
    error: function(e) {                  // Do this if it failed
      confirm("Error", e.message);
    }
  });
```

```
function handleResults(messages) {
  var i, list = "";
  for(i=0; i < messages.length; i += 1) {
    list += formatMessage(messages[i]);
  }
  displayResults(list);
}
```

- handleResults() simply has to deal with JSON data
- In this case, a list of messages on a messageboard
- The callback function gets results passed in a parameter

JSONP Data Sources

- For web-apps, we MUST use JSONP (unless it is our own server)
  - Not as big a limitation as you might think
    - The Met Office – weather data from the horses mouth
      - www.metoffice.gov.uk
    - Yahoo – a huge range of services
      - https://developer.yahoo.com/yql/console/
    - Google Apps – return data from Google Docs
      - https://developers.google.com/apps-script/guides/content
    - Also twitter, facebook, news sites
- Of course, all this in addition to RSS feeds via a good proxy service (like jGFeed)