Digital Skills Academy

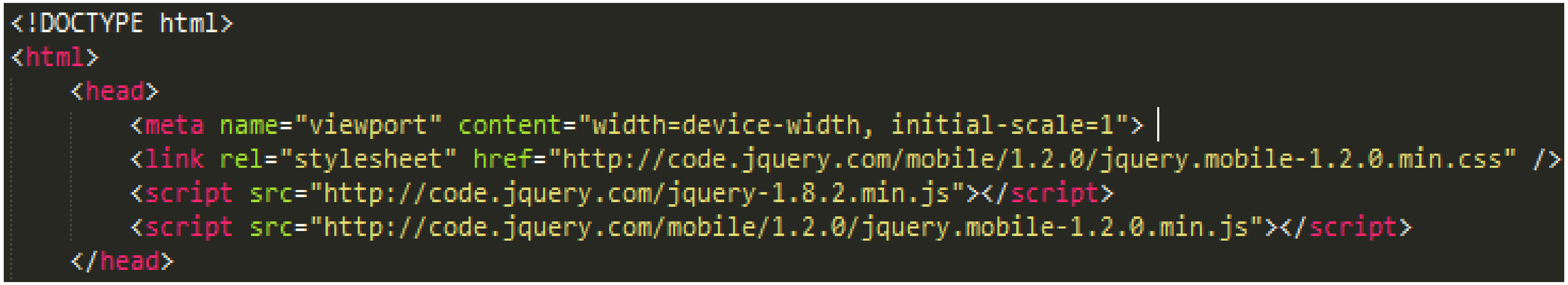
Unit 3: jQuery Mobile II

Last Session

- jQuery Mobile.
  - Anatomy of a page.
  - Single and multi-page sites.
  - Page transitions.
  - Dialog.
  - Applying transitions to dialogs and forms
- Mobile Emulation
- jQuery Mobile Events

A jQuery Mobile Page

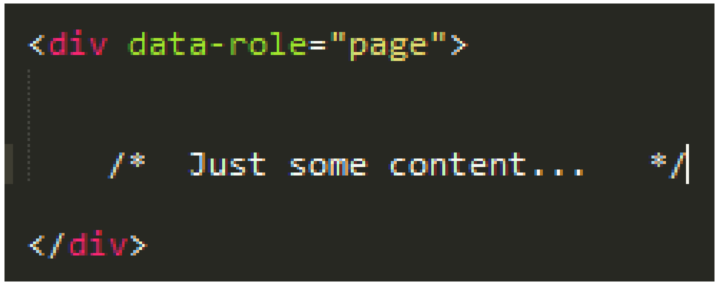
- Minimal requirements for JQM.



```html
<!DOCTYPE html>
<html>
    <head>
        <meta name="viewport" content="width=device-width, initial-scale=1">
        <link rel="stylesheet" href="http://code.jquery.com/mobile/1.2.0/jquery.mobile-1.2.0.min.css" />
        <script src="http://code.jquery.com/jquery-1.8.2.min.js"></script>
        <script src="http://code.jquery.com/mobile/1.2.0/jquery.mobile-1.2.0.min.js"></script>
    </head>
```

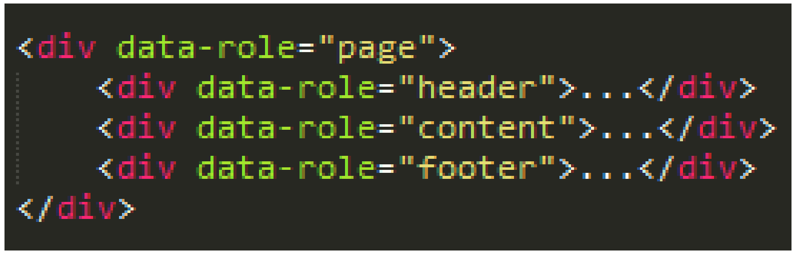
Single and Multi Page Sites

- The rest of the page is standard mark-up but includes some unusual custom data attributes.

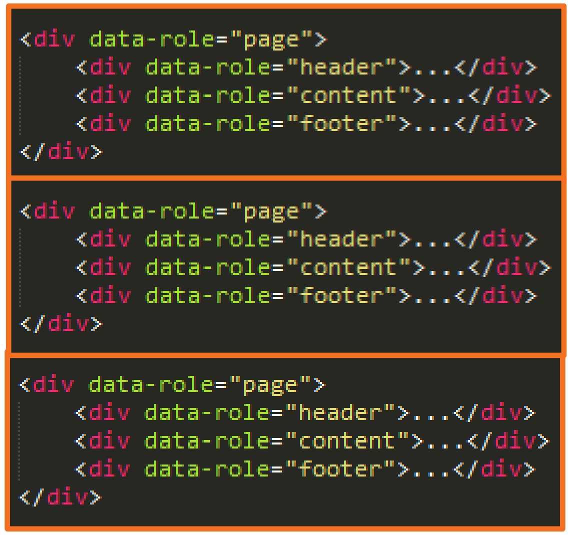


```
<div data-role="page">


    /*  Just some content...    */

</div>
```

- The data-role attribute is telling the browser how to think about this portion of the page. Setting the value to “page” tells it to think of this DIV as a complete page. We can add child elements like so:



```
<div data-role="page">
    <div data-role="header">...</div>
    <div data-role="content">...</div>
    <div data-role="footer">...</div>
</div>
```



```
<div data-role="page">
    <div data-role="header">...</div>
    <div data-role="content">...</div>
    <div data-role="footer">...</div>
</div>
```

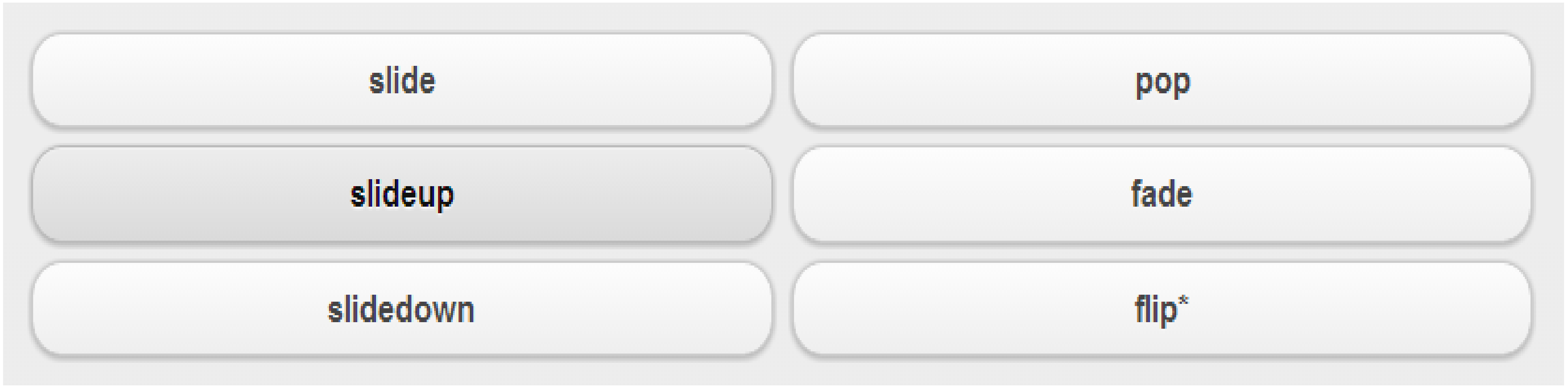
```
<div data-role="page">
    <div data-role="header">...</div>
    <div data-role="content">...</div>
    <div data-role="footer">...</div>
</div>
```

```
<div data-role="page">
    <div data-role="header">...</div>
    <div data-role="content">...</div>
    <div data-role="footer">...</div>
</div>
```

What’s missing from the multipage version?

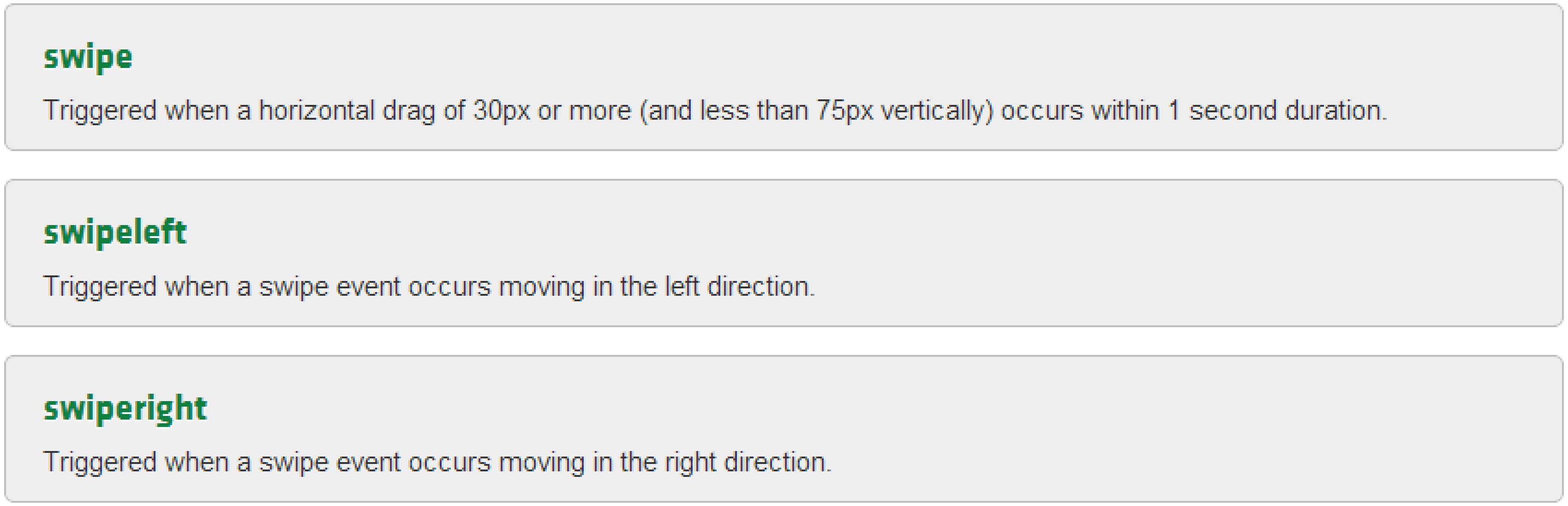
Animated Transitions

JQM provides six CSS-based transitions which we can apply to any page changes, or form submissions.



http://demos.jquerymobile.com/1.0/docs/pages/page-transitions.html

jQuery Mobile Events



swipe

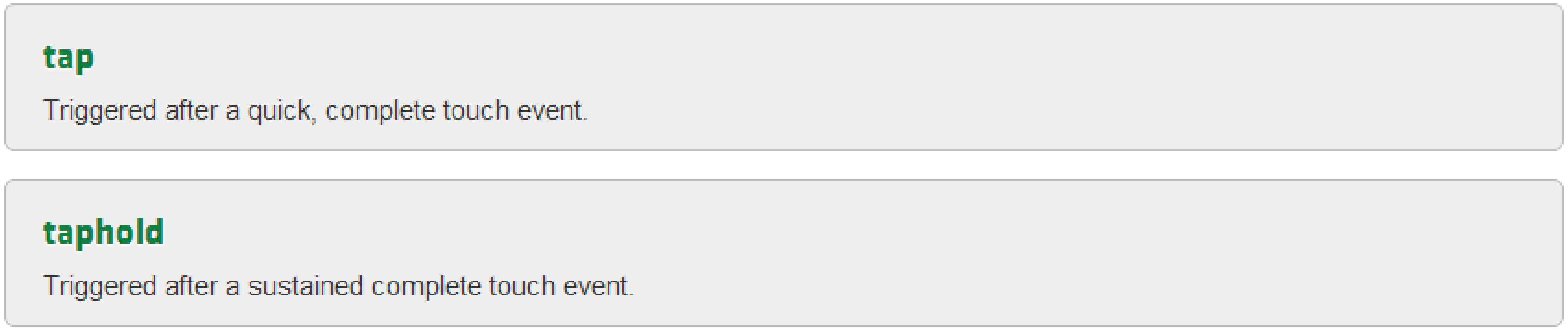
Triggered when a horizontal drag of 30px or more (and less than 75px vertically) occurs within 1 second duration.

swipeleft

Triggered when a swipe event occurs moving in the left direction.

swiperight

Triggered when a swipe event occurs moving in the right direction.



tap

Triggered after a quick, complete touch event.

taphold

Triggered after a sustained complete touch event.

http://api.jquerymobile.com/category/events/

Quiz

Open the recap quiz on Moodle now.

You have five minutes to complete the quiz.

This Session

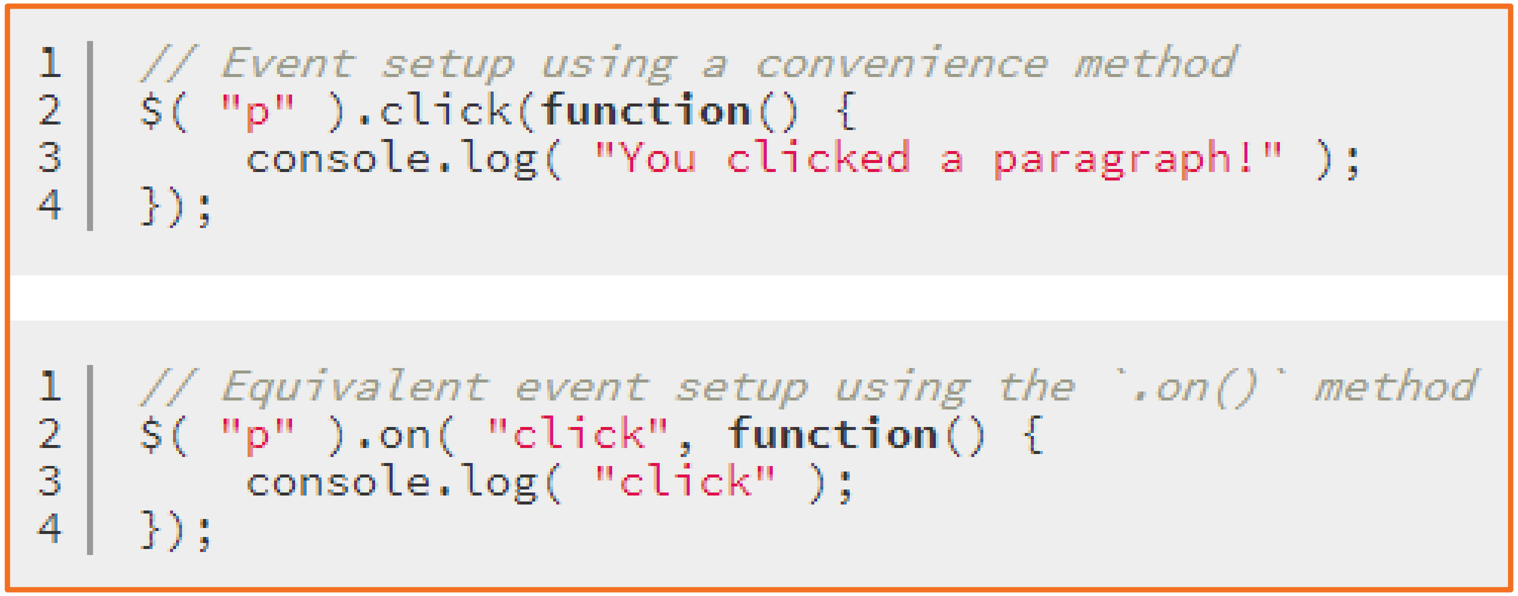
- JQM API.
  - DOM Events.
  - Programmatically triggering JQM.
- An Introduction to XML.
- Live Data.
  - Ajax.
  - JQM and Ajax.

Useful Links

- http://api.jquery.com/
- http://api.jquerymobile.com/
- http://learn.jquery.com/jquery-mobile/

jQuery Mobile API

- Up to now we've used data-attributes to attach JQM to specific DOM elements.
- JQM includes a rich API which exposes these same features to JavaScript.
- Includes "convenience" methods for most native browser events.



```
// Event setup using a convenience method
$( "p" ).click(function() {
    console.log( "You clicked a paragraph!" );
});
```

```
// Equivalent event setup using the `.on()` method
$( "p" ).on( "click", function() {
    console.log( "click" );
});
```

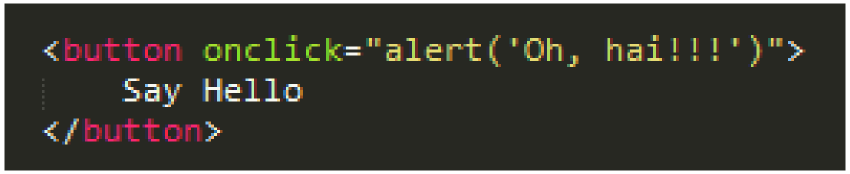
DOM Events

- Many events occur during a typical session:
  https://developer.mozilla.org/en/DOM/DOM_event_reference

- Many are triggered by the user, many others are automatic.

- We can listen for specific events by writing an event handler.

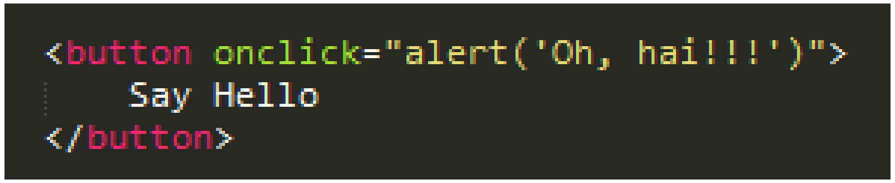
- The easiest way (but poor practice) is to embed some JS inline. In this case, the event is a click and our alert function becomes an event handler.



```
<button onclick="alert('Oh, hai!!!')">
    Say Hello
</button>
```

DOM Events

1. Inline Events.



```
<button onclick="alert('Oh, hai!!!')">
    Say Hello
</button>
```

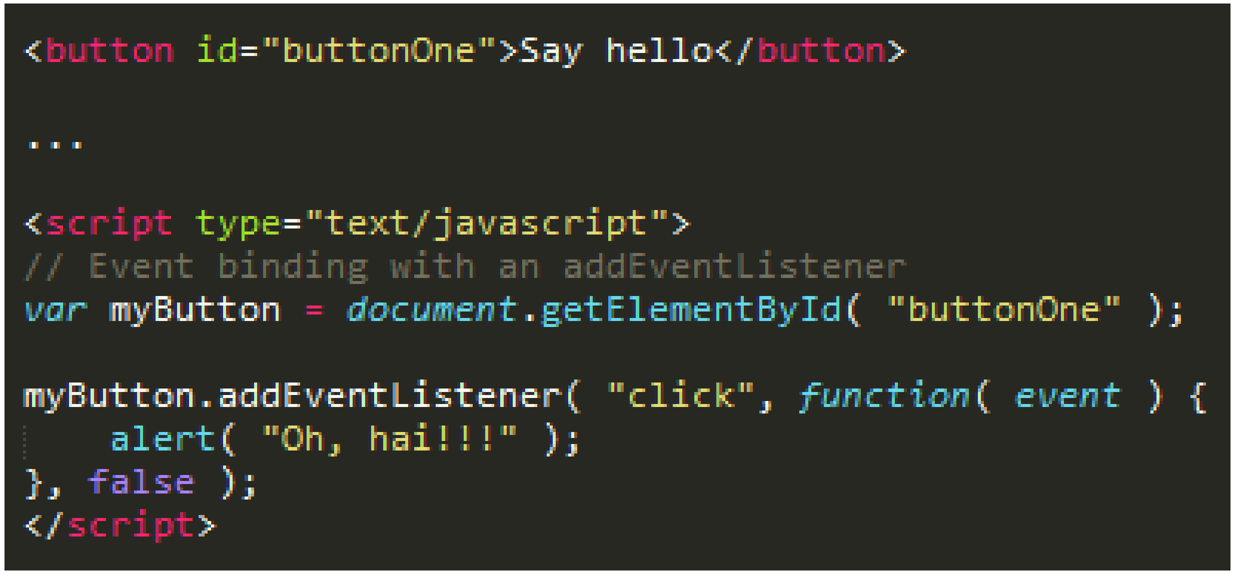
- Bad: Couples the HTML with the JS.
- Bad: Not scalable – every button will require its own code.

2. Event Binding.

- Get a reference for the element using JS and then react to changes.
- Good: HTML and JS are uncoupled.
- Bad: Microsoft browsers have a different way of attaching the event (using **“attachEvent”** instead of **“addEventListener”**).

DOM Events

2. Event Binding.



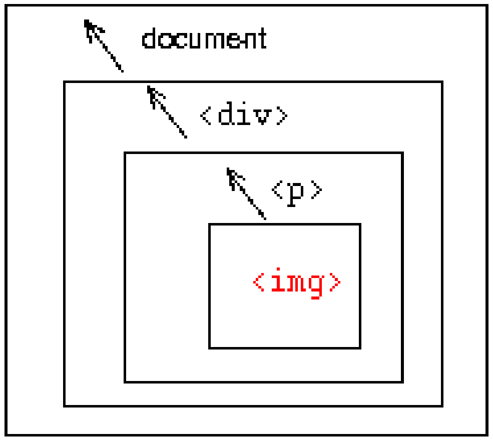
```
<button id="buttonOne">Say hello</button>

...

<script type="text/javascript">
// Event binding with an addEventListener
var myButton = document.getElementById( "buttonOne" );

myButton.addEventListener( "click", function( event ) {
    alert( "Oh, hai!!!" );
}, false );
</script>
```

DOM Events

3. Event Delegation.

DOM event delegation is a mechanism of responding to ui-events via a single common parent rather than each child, through the magic of event "bubbling" (aka event propagation).

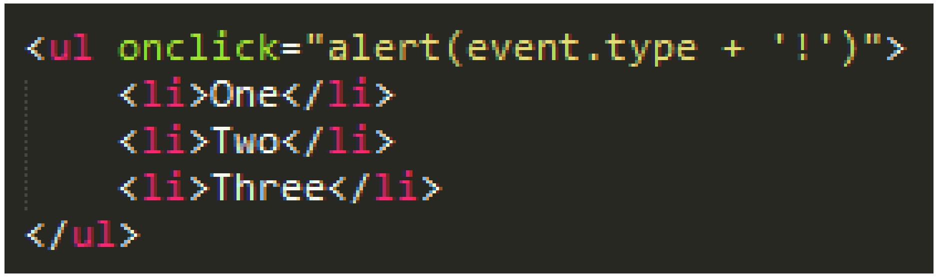
Event Bubbling:
Events chain upwards after they occur.



- Bind an event handler to a single parent element.
- The handler will execute when an event occurs on **any** of its children.

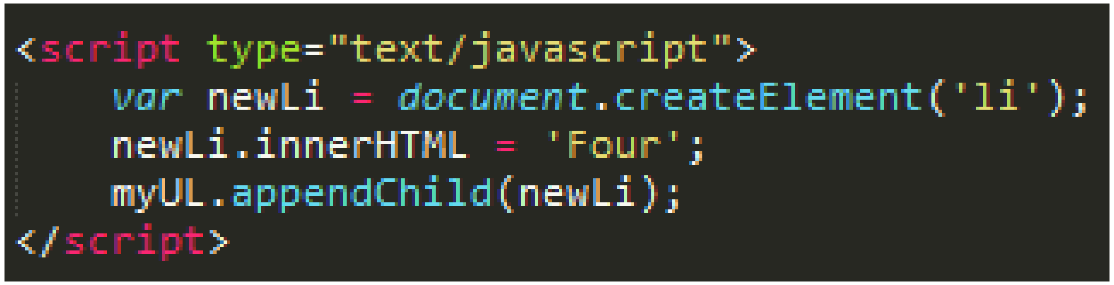
DOM Events

Example of Event Delegation:
Click on an LI node. Click event will bubble to the UL.



```
<ul onclick="alert(event.type + '!')">
    <li>One</li>
    <li>Two</li>
    <li>Three</li>
</ul>
```

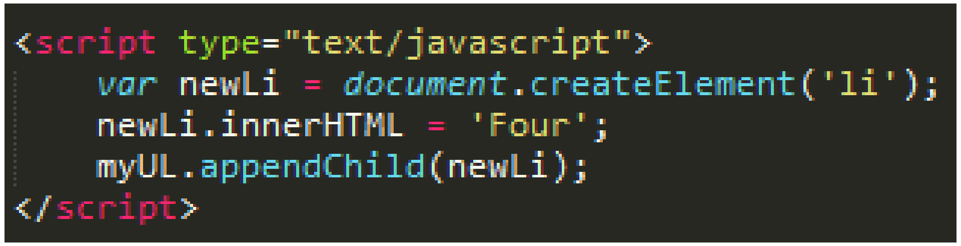
Now imagine we need to add another LI, programmatically through JavaScript.



```
<script type="text/javascript">
    var newLi = document.createElement('li');
    newLi.innerHTML = 'Four';
    myUL.appendChild(newLi);
</script>
```

Event Delegation

Create the node and append it to the parent.



```
<script type="text/javascript">
    var newLi = document.createElement('li');
    newLi.innerHTML = 'Four';
    myUL.appendChild(newLi);
</script>
```

Without using event delegation you would have to "rebind" the "onclick" event handler to the new<li> element, in order for it to act the same way as its siblings.

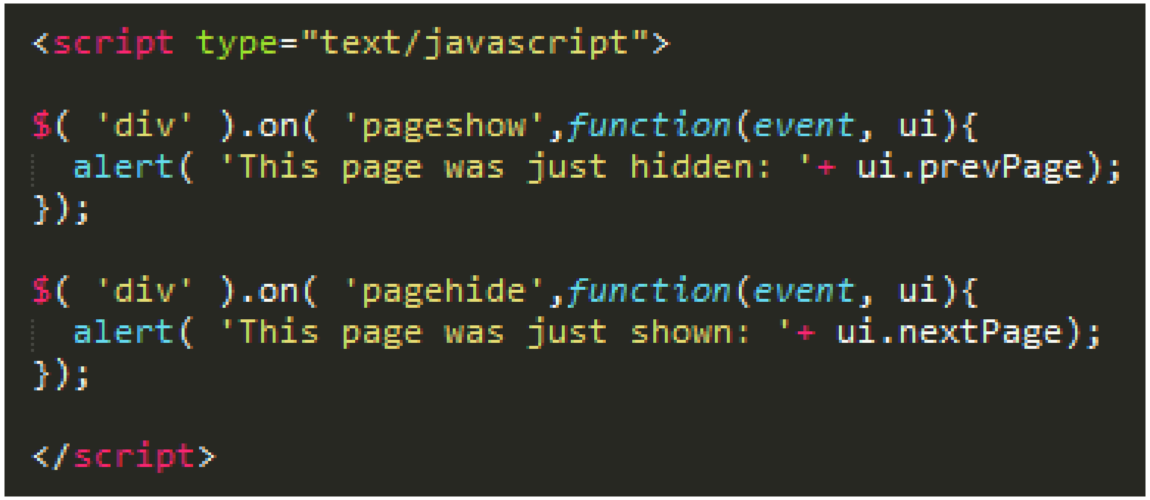
With event delegation you don't need to do anything. Just add the new <li> to the list and you're done.

https://learn.jquery.com/events/event-delegation/
http://stackoverflow.com/questions/1687296/what-is-dom-event-delegation
http://stackoverflow.com/questions/4616694/what-is-event-bubbling-and-capturing

Triggering Event Handlers

We can trigger event handlers without waiting for user interaction using the typical jQuery methods of **.on** or **.bind.**

e.g. pageshow and pagehide events.


```
<script type="text/javascript">

$( 'div' ).on( 'pageshow',function(event, ui){
  alert( 'This page was just hidden: '+ ui.prevPage);
});

$( 'div' ).on( 'pagehide',function(event, ui){
  alert( 'This page was just shown: '+ ui.nextPage);
});

</script>
```

Full list of events: http://demos.jquerymobile.com/1.2.1/docs/api/events.html

Best Practice: Namespacing Events

- When working with large applications or many libraries, many methods could share the same names.

- How can we prevent clashes?

- Name spaces are groupings, usually associated with a similar theme or set of functionalities and properties.

name = <namespace identifier> separator <local name>

```
// Namespacing events
$( "p" ).on( "click.myNamespace", function() { /* ... */ } );
$( "p" ).off( "click.myNamespace" );
$( "p" ).off( ".myNamespace" ); // unbind all events in the namespace
```

- http://en.wikipedia.org/wiki/Namespace

Activity 1: Video

jQuery Mobile Touch Events

http://www.youtube.com/watch?v=htuFui6RxuE

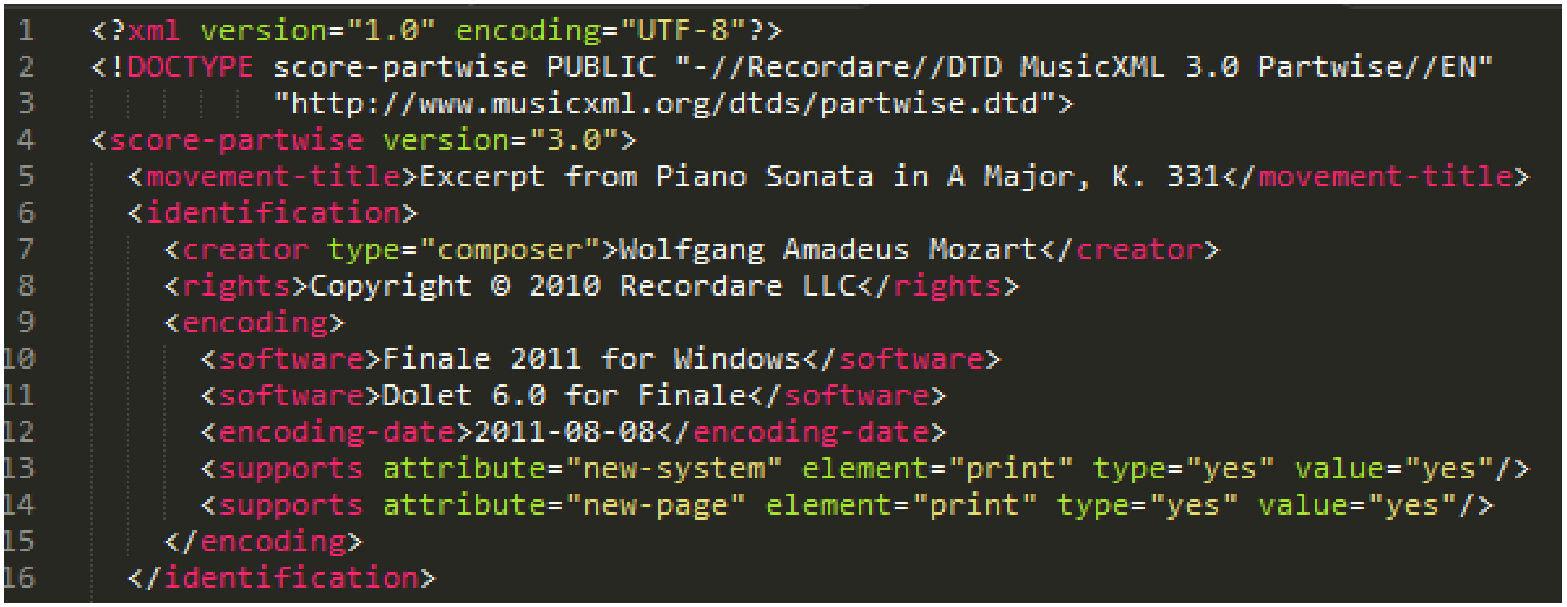
Break: 10 Minutes

eXtensible Mark-up Language

- XML is W3C Recommendation.
- XML doesn't **DO** anything! It's only intended to store and transport data.
- Designed to hold data rather than to display it.
- XML is a mark-up language, similar to HTML.
- XML tags are user defined.
- XML is "extensible" because, (unlike HTML), the mark-up symbols are unlimited and self-defining.
- Used to build a standard or common way to describe information.

Sample XML

- Excerpt from MozartPianoSonata.xml



```xml
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE score-partwise PUBLIC "-//Recordare//DTD MusicXML 3.0 Partwise//EN"
          "http://www.musicxml.org/dtds/partwise.dtd">
<score-partwise version="3.0">
  <movement-title>Excerpt from Piano Sonata in A Major, K. 331</movement-title>
  <identification>
    <creator type="composer">Wolfgang Amadeus Mozart</creator>
    <rights>Copyright © 2010 Recordare LLC</rights>
    <encoding>
      <software>Finale 2011 for Windows</software>
      <software>Dolet 6.0 for Finale</software>
      <encoding-date>2011-08-08</encoding-date>
      <supports attribute="new-system" element="print" type="yes" value="yes"/>
      <supports attribute="new-page" element="print" type="yes" value="yes"/>
    </encoding>
  </identification>
```

- http://www.musicxml.com/

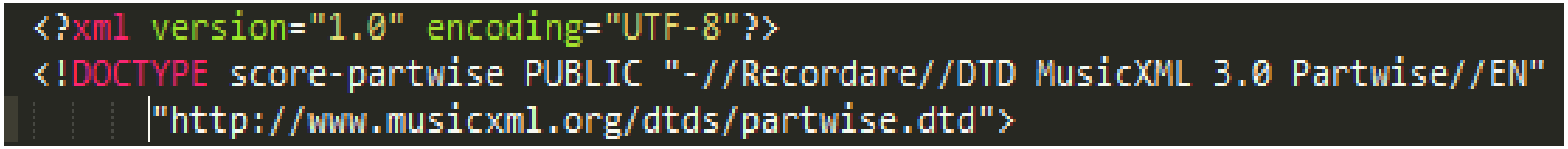
XML

- XML is far less tolerant of errors than HTML.
- Documents must be “well formed”.
  - The document contains only properly encoded legal Unicode characters.
  - None of the special syntax characters such as < and & appear except when performing their mark-up delineation roles.
  - The begin, end, and empty-element tags that delimit the elements are correctly nested, with none missing and none overlapping.
  - The element tags are case-sensitive; the beginning and end tags must match exactly. Tag names cannot contain any of the characters !"#$%&'()*+,/;<=>?@[\]^`{|}~, nor a space character, and cannot start with -, ., or a numeric digit.
  - A single "root" element contains all the other elements.
- http://en.wikipedia.org/wiki/XML#Well-formedness_and_error-handling

XML

- As well as being “well formed”, documents must be “valid”.

- A valid document conforms to a Document Type Definition (DTD). This describes all the tags that your specific xml flavour can use (remember that tags are user defined!).

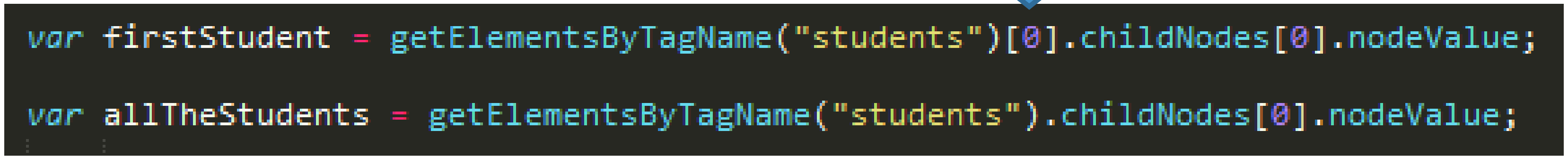


```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE score-partwise PUBLIC "-//Recordare//DTD MusicXML 3.0 Partwise//EN"
      "http://www.musicxml.org/dtds/partwise.dtd">
```

- A DTD is equivalent to a HTML Doctype.

XML with JavaScript

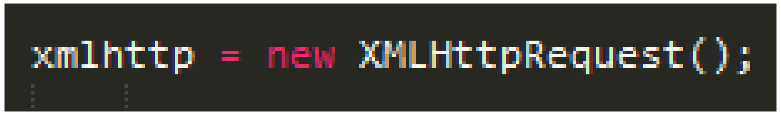
- All modern browsers contain an XML Parser.
- The XML Parser converts an XML document into an XML DOM object.
- We can navigate this DOM with JavaScript.



```
var firstStudent = getElementsByTagName("students")[0].childNodes[0].nodeValue;

var allTheStudents = getElementsByTagName("students").childNodes[0].nodeValue;
```

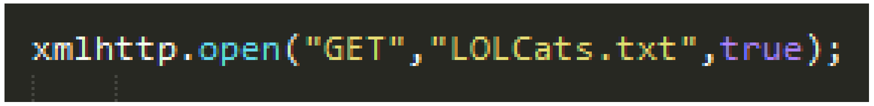
The XMLHttpRequest Object

- We can use JavaScript to exchange XML data with a server using XMLHttpRequest.
  - Update a page without reloading the document.
  - Request or send data from the document.
- All modern browsers contain an XMLHttpRequest Object.
- We instantiate a new XMLHttpRequest like so:



```
xmlhttp = new XMLHttpRequest();
```

- And use it to pull some resource:



```
xmlhttp.open("GET","LOLCats.txt",true);
```

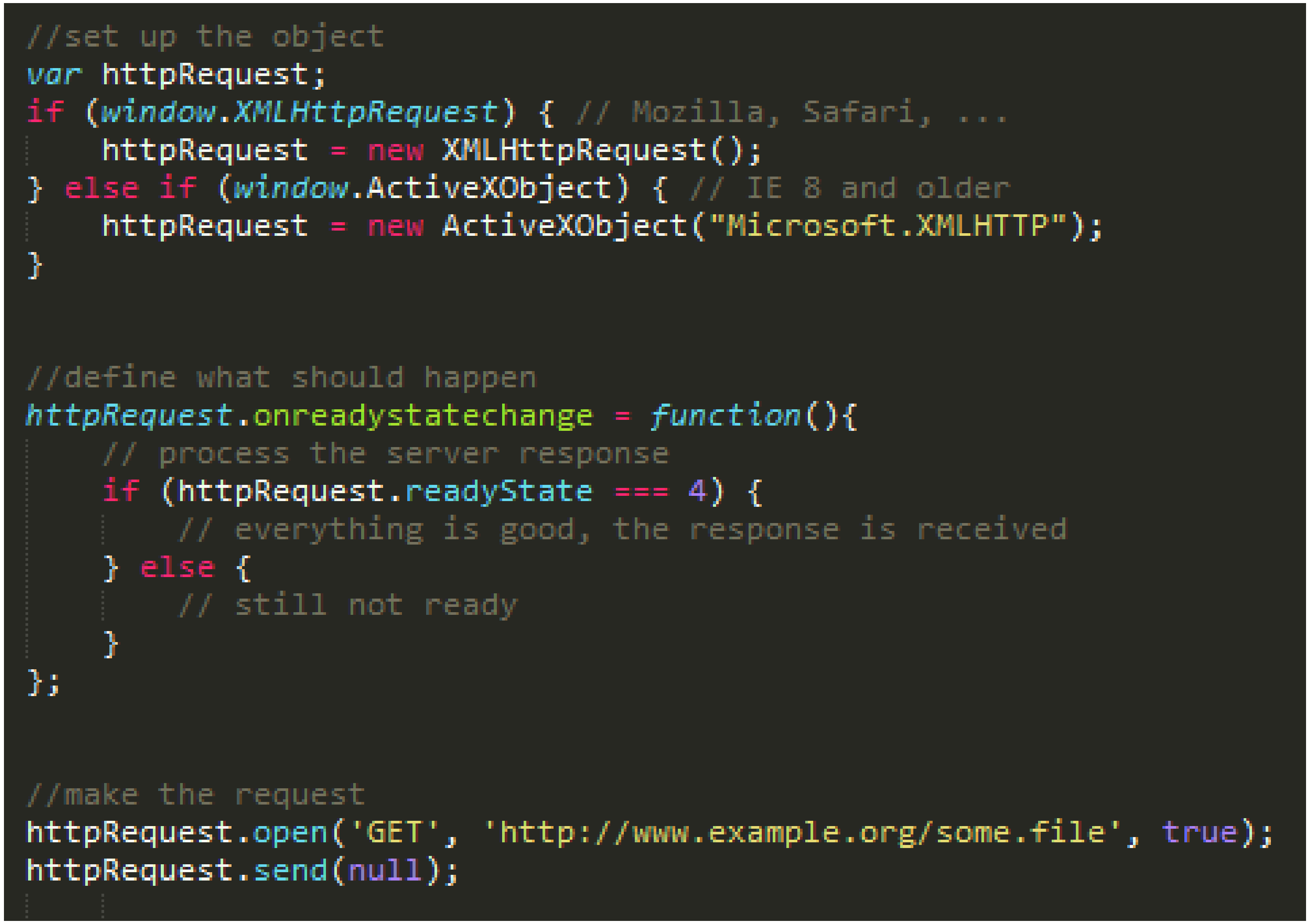
Ajax = XML + JavaScript

- Asynchronous JavaScript and XML.
  - Plus HTML and CSS (and JSON).

- Despite the name, XML is not required! We can replace it with JSON.

- Disclaimer: jQuery and JQM make this allot easier! But lets take a look at the complete script, using pure JavaScript/XML first.

XML and JavaScript: Ajax

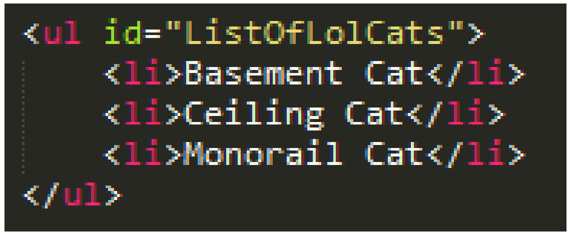


```
//set up the object
var httpRequest;
if (window.XMLHttpRequest) { // Mozilla, Safari, ...
    httpRequest = new XMLHttpRequest();
} else if (window.ActiveXObject) { // IE 8 and older
    httpRequest = new ActiveXObject("Microsoft.XMLHTTP");
}


//define what should happen
httpRequest.onreadystatechange = function(){
    // process the server response
    if (httpRequest.readyState === 4) {
        // everything is good, the response is received
    } else {
        // still not ready
    }
};


//make the request
httpRequest.open('GET', 'http://www.example.org/some.file', true);
httpRequest.send(null);
```

Recap: DOM Manipulation

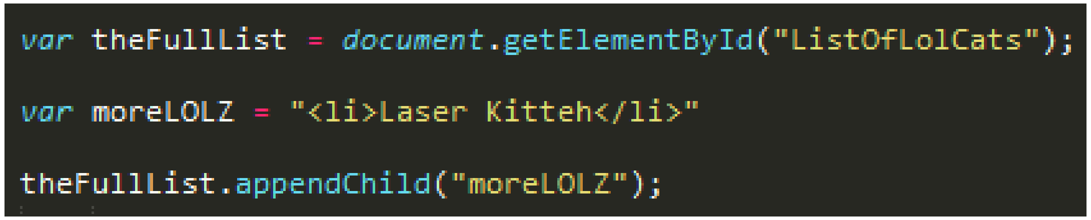
- Remember that we can use JavaScript to inject elements into the DOM.

Some HTML:



```html
<ul id="ListOfLolCats">
    <li>Basement Cat</li>
    <li>Ceiling Cat</li>
    <li>Monorail Cat</li>
</ul>
```

Some JavaScript:



```javascript
var theFullList = document.getElementById("ListOfLolCats");

var moreLOLZ = "<li>Laser Kitteh</li>"

theFullList.appendChild("moreLOLZ");
```

- Step 1 – Find the parent element.
- Step 2 - Add a child element.

jQuery.ajax()

- The jQuery library has a full suite of Ajax capabilities.

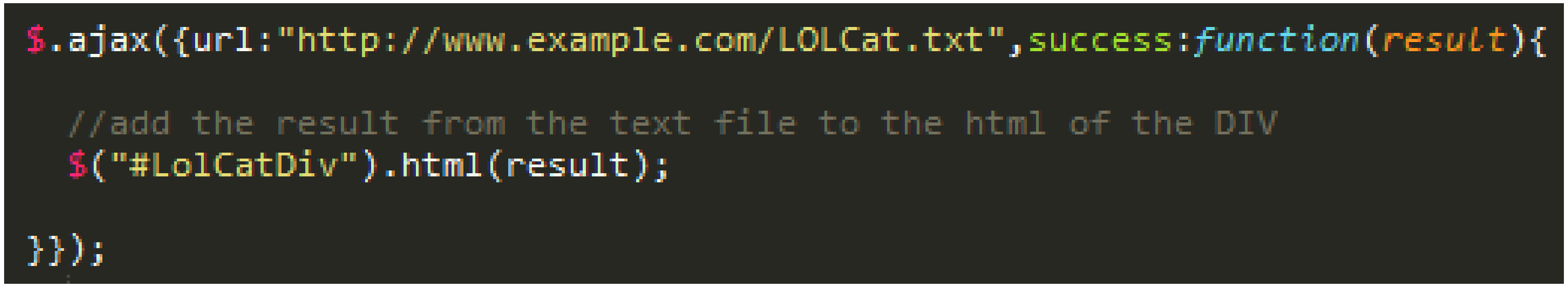
  https://api.jquery.com/jQuery.ajax/

- Part of jQuery. Uses the jQuery object.
  - $.ajax

- Since jQuery 1.5, the jQuery XMLHttpRequest (jqXHR) object returned by $.ajax() is a superset of the browser's native XMLHttpRequest object.

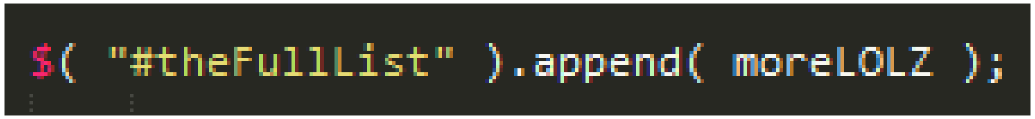
jQuery.ajax()

- Example call to retrieve LOLCats.txt (and add the textual contents to the document).



```
$.ajax({url:"http://www.example.com/LOLCat.txt",success:function(result){

  //add the result from the text file to the html of the DIV
  $("#LolCatDiv").html(result);

}});
```

- We could easily add HTML elements in the same way:



```
$( "#theFullList" ).append( moreLOLZ );
```

jQuery Mobile Ajax

The jQuery Mobile navigation model is based on Ajax.

The basic workflow with page loading is as follows:

- A page is requested with a normal HTTP request, and subsequent "pages" are then requested and injected into that page's DOM.
- Because of this, the DOM may have a number of "pages" in it at a time, each of which can be re-visited by linking to its data-url attribute.

Both for pages already in the DOM and for pages that need to be loaded via Ajax – use the $.mobile.changePage() function.

$.mobile.changePage() contains all of the logic for finding pages to transition to and from, and how to handle various response conditions such as a page not found

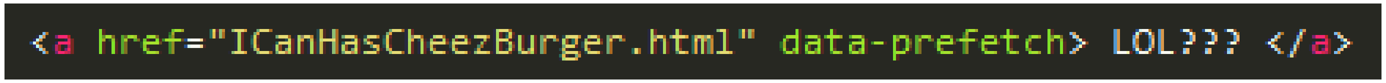
http://demos.jquerymobile.com/1.2.0/docs/pages/page-navmodel.html

jQuery Mobile Ajax

JQM and Ajax resolve a few common headaches.

- jQuery Mobile allows pages to be pulled into the DOM dynamically via its default click hijacking behaviour.

- Pages can be prefetched to optimize network resources. You can prefetch as many linked pages as you like. Just add **data-prefetch** to all the links you want to prefetch.



```
<a href="ICanHasCheezBurger.html" data-prefetch> LOL??? </a>
```

- The Ajax loading message only appears if the framework hasn't finished prefetching the page by the time the link is followed.

- More on using JQM Ajax in the labs!

Activity 2: Video

Reading XML files with jQuery.

http://www.youtube.com/watch?v=w_gReWEq-5g

Summary

DOM Events.

Trapped via:

- Inline
- Event Binding
- Event Delegation

Programmatically triggering JQM.

- .on
- .bind

XML

Ajax.

JQM and Ajax.

- $.ajax();

http://www.lolcatbible.com/index.php?title=How_to_speak_lolcat

References

Slide 5: http://demos.jquerymobile.com/1.0/docs/pages/page-transitions.html

Slide 6: http://api.jquerymobile.com/category/events/

Any jQuery Code: https://github.com/jquery/jquery/blob/master/MIT-LICENSE.txt

Lol: http://www.lolcatbible.com/index.php?title=How_to_speak_lolcat

Slide 18: http://en.wikipedia.org/wiki/Namespace

Slide 22: http://www.musicxml.com/

Slide 23: http://en.wikipedia.org/wiki/XML#Well-formedness_and_error-handling

Slide 30: https://api.jquery.com/jQuery.ajax/

Digital Skills Academy

QUESTIONS???