JQuery, DOM Events, JQuery UI, JSON and AJAX

WS Slides #3

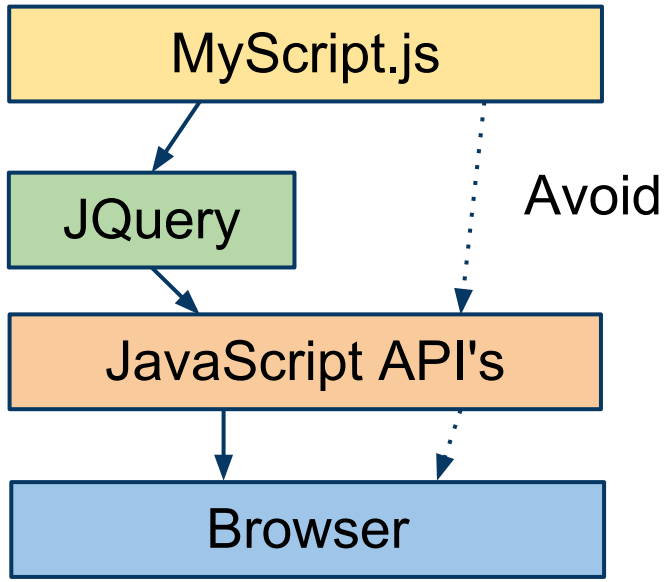
JQuery Overview

Javascript library by John Resig
- Cross browser: Trying to eliminate differences
- Free, Open Source
- Very good documentation http://docs.jquery.com/Main_Page
- Lib. in file **jquery-1.9.0.js** if using version 1.9 (put under /Web Pages in application)

Features
- Much better DOM, DOM Events and CSS API's
- Better AJAX API (upcoming)
- Effects, animations
- A GUI component suit (jQuery UI)
- ...

JQuery Design

Library built on top of JavaScript API's
- Hides much of JavaScripts "awful" parts
- Interesting in it's own right, excellent design
- Extensible: "Plugins", ...



JQuery Philosophy

1. Find (or create) some DOM elements using CSS selectors (or HTML). Elements will be "wrapped" in the JQuery object (functionality enhanced)

2. Do something with them using JQuery methods (instead of native JS API)
- Chain multiple method calls to process a set of elements (filters)*

3. Don't need to explicitly traverse the set. Use the wrapper and implicit iteration (each()-method)
- Possible to get "previous" set back

*) Design pattern pipes and filters

JQuery API

API represented as JQuery object (abbreviated to $)

```
// Any JQuery statement starts
JQuery. ...

// Same as above but quicker
$. ...
```

Possible to manipulate

- Elements and element values
- Attributes
- CSS properties
- ...

Deferring Execution

JQuery (JavaScript) should normally not execute before DOM fully constructed

- To defer execution we use (first of all)

```
// Function called when DOM ready (wrapping document)
$(document).ready(function(){
    // Code to execute when DOM ready
});
```

Or (simpler, same behaviour as above)

```
// Wrapping a function
$(function(){
    // Code to execute when DOM ready
});
```

Manipulate Elements

Find

```
// Get reference to JQuery object wrapping element myList (using
element attribute id)
var ul = $("#myList");

// Selecting in DOM (many more methods)
var children = $("#myList").children();
var parent = $("#myList").parent();
```

Add

```
// Add input element to DOM
$("<input id='btnClose' type='button'
                          value='Save'/>").appendTo('#popUp');
```

Modify

```
$('div.second').replaceWith('<h2>New heading</h2>');
```

Delete

```
$('.productTableBody tr').remove();
```

Manipulate Special Elements

Often need values of input, select and textarea elements

```
// Get some user input (# = using the id-attribute)
var input = $(“#myInput).val();
```

Manipulate Attributes and CSS

Manipulating attributes and CSS

Attribute

```
$('#greatphoto').attr('alt');  // Get
$('#greatphoto').attr('alt', 'Beijing Brush Seller'); // Set
```

CSS

```
$('#mydiv').css('color') // Get
$('#mydiv').css('color', 'green')  // Set
```

JQuery Traversing

Traversing

```
$("#myList").children().each( function(index, element) {
        $(this).css("color", "red");
});
```

Dissection

- JQuery will find all children to “myList”-element and traverse the list
- Callback method passed to each() will execute for each child element, will set color for each child
- $(this) is the wrapped child element ("this" only is the current DOM element)

Wrapped or Not

Sometimes the underlying DOM nodes are returned (non wrapped)

```
// Get will unwrap and return underlying DOM node
var firstChild = $("#myList").children().get(0);

// Can't use JQuery methods, not a wrapped object
firstChild.children()   // BAD!!!
```

When do we have a wrapped set, when do we have to original DOM node(s)?

- Check documentation!

HTML Events

A great many, see http://www.w3.org/TR/DOM-Level-3-Events/

Event flow (Same for all browsers???)
http://www.w3.org/TR/DOM-Level-3-Events/#event-flow

This has been very messy, different "levels", incompatible standards and more...

Event Bubbling

Some knowledge of HTML event capturing and bubbling can possible be useful

- Same listener for all children of an element

http://www.w3.org/TR/DOM-Level-3-Events/#dom-event-architecture

JQuery DOM Events

We avoid the DOM event mess by using JQuery

- Use JQuery registration of event handlers
- Event names similar DOM level 0, some examples

// DOM level 0 names (skip "on" to get JQuery name)
onclick, The event occurs when the user clicks on an element
ondblclick, The event occurs when the user double-clicks on an element
onmousedown, The event occurs when a user presses a mouse button over an element
onmousemove, The event occurs when the pointer is moving while it is over an element
onmouseover, The event occurs when the pointer is moved onto an element
onmouseout, The event occurs when a user moves the mouse pointer out of an element
...many more...

JQuery Event Setup

Static elements

```
// No () for listener function!!!
$(function(){
    $("#btn1").on("click", listeners.click);
});
```

Dynamic elements (myBtn created in future)

```
// Connect (in future)
$(function(){
 $(document).on("dblclick","#myBtn", listeners.dblclick);
});
```

JQuery Event Handling

Events handled by callback functions (similar to Java)

```
// Setup
$("#txt").on("mouseenter", listeners.mouseEnter).
    on("mouseleave", listeners.mouseLeave ).
    on("focus", listeners.focus);

// Handling
var listeners = (function(){
    var color;
    mouseEnter: function(){
                color = $(this).css("background-color");
                $(this).css("background-color", "gray");
    },
    mouseLeave: function(){
                $(this).css("background-color", color);
    },
    focus: function(){
       ...
    },
})();
```

So Far ...

JQuery simplifies a lot but still...

- To manipulate the GUI we have to manipulate the DOM …
- .. this is like we only had the paint-method in swing. Have to build everything by hand …
-... far too much work. We would like to have components like JButton etc...

… and of course (surprise)

- .. there are many, many JS component collection out there
- We'll try JQueryUI, not perfect, but feasible, simple

JQuery UI

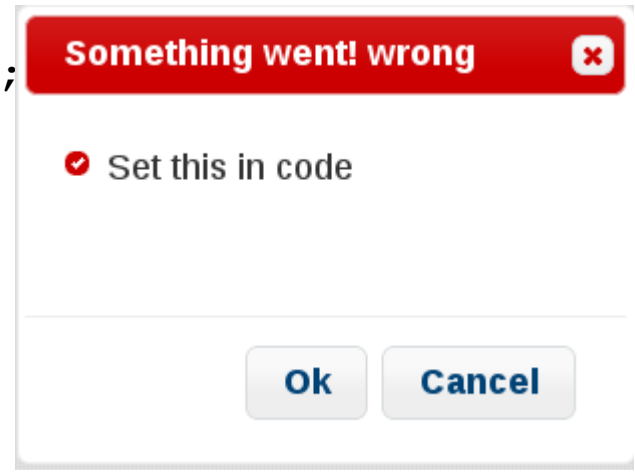
Built on top of Jquery http://jqueryui.com/

- A component is a combination of JS and CSS
- Themeroller (on line app), to generate CSS
- Download zip file instal into Web Pages/
- Event handling even simpler than JQuery

JQuery UI Component Example

```
// Create a dialog box
var myDialog = $("#dialog-message").dialog({
            autoOpen: false,
            modal: true,
            stack: true,
            buttons: {
                Ok: function() {
                    $(this).dialog("close");
                },
                Cancel: function() {
                    $(this).dialog("close");
                }
            }
        });

// Show it
myDialog.dialog("open");
```



Some dialog text from HTML page.
CSS auto generated

JavaScript Object Notation, JSON

Text-based open standard designed for human-readable data interchange

- Text based representations of JavaScript objects, http://tools.ietf.org/html/rfc4627 ,http://www.json.org/
- Language independent (but derived from JavaScript)
- Lightweight (compare XML)

Normally automatic conversion from/to JSON / JavaScript object in browser at request and response

- If not converted, use the below

```
// Have JSON (text) get an object
var obj = $.parseJSON('{"name":"John"}');
```

Example JSON

```
{
    "firstName": "John",                        // Note " not '
    "lastName": "Smith",
    "age": 25,
    "address": {
        "streetAddress": "21 2nd Street",
        "city": "New York",
        "state": "NY",
        "postalCode": "10021"
    },
    "phoneNumber": [
        {
            "type": "home",
            "number": "212 555-1234"
        },
        {
            "type": "fax",
            "number": "646 555-4567"
        }
    ]
}
```

AJAX

"Asynchronous JavaScript and XML"

- Possible to update small parts of DOM by calling server asynchronously (i.e. method call will return instantly, non blocking)
- Can't call synchronously, JavaScript single threaded, would freeze application
- Possible: Single page applications, much higher user interactivity

XMLHttpRequest API

W3C Specification for asynchronous HTTP requests http://www.w3.org/TR/XMLHttpRequest/

In JavaScript we can access the XMLHttpRequest object (but we don't, we use JQuery)
- Object presented as XHR in Chrome debugger

Attributes
- onreadystatechange // event handler callback
- responseText/responseXML // return value

Methods
- Open, send, abort,

XMLHttpRequest Details

Calls will progress through “ready states” (setting readyState of XMLHttpRequest object)

- Open -> readyState = 1
- After send -> readyState = 2
- Response content begins to load -> readyState = 3
- Finished loading -> readyState = 4

Each change will trigger call to handler function assigned to onreadystatechange attribute

Browser handles limited numbers of XMLHttpRequest objects (4?)

jqXHR API

JQuery API wrapping the low level XmlHttpRequest

jqXHR exposes some of parts of XmlHttpRequest and more...

-readyState
-status
-statusText
-responseXML and/or responseText when the underlying request responded with xml and/or text, respectively
-setRequestHeader(name, value) which departs from the standard by replacing the old value with the new one rather than concatenating the new value to the old one
-getAllResponseHeaders()
-getResponseHeader()
-abort()

Asynchronous Programming

AJAX implies **asynchronous programming** having callback functions as parameters

- Callback function called when response ready

Problematic program structure (callback possible have callback, possible have...severe scaling problems)

- Code spread out over potentially many functions
- Solution: Use a return object representing the result in the future, the "deferred" object, more to come ...

AJAX Return Types

AJAX call can return
- XML
- JSON
- ...

JSON normally easier to use if implementing a JavaScript client

AJAX with JQuery

Many variations, shorthand's

```
// GET
$.getJSON("...someURL...");

//POST
$.ajax( { type: 'POST',     // Note: Using JSON as parameter
          url: "...someURL...",
          data: person} );


// Using deferred object (calls above will return deferred object)
var deferred = $.getJSON(...);
deferred.done(function( data ){ // Executed later
    // Incoming JSON converted to JS object
     alert("Done" + data); // Asynch. callback
});
deferred.fail(function(){
      alert("Fail");       // Asynch. callback
 });
// Program continues directly here...
```

Multiple AJAX Calls

If one AJAX call depends on a previous one?
- How do we serialize the calls so that data arrives and actions are performed in order

Solution: Serialize it using JQuery when … then

```
// Deferred as result from AJAX call like $.getJson()
$.when( deferred ).then( successCallback, failCallback);
```

```
Note: $.when( deferred ) also will return a deferred, then() must be
called on a deferred
```

Example: when ... then

```
// getCount will return deferred object (from AJAX call)
me.container.getCount().then(function(result) {
        // Do some calculations using result from AJAX call
        m = ...
        // Return m as parameter to next “then callback”
        return m;
}, fail).then( function(m) {
        // Do next AJAX call
});
```

JavaScript Same Origin Policy

Security Issue
A script can read only the properties of windows and documents that have the same origin (i.e., that were loaded from the same host, through the same port, and by the same protocol) as the script itself.
- Untrustworthy script in one window could use DOM methods to read the contents of documents in other browser windows, which might contain private information.
Details: http://www.w3.org/Security/wiki/Same_Origin_Policy

Poses particular problems for large websites that use more than one server
- JSONP, trick to circumvent restrictions http://en.wikipedia.org/wiki/JSONP ...others possible better

Bookmarkability

Convenient for user to be able to bookmark pages
- If page has URL, store (as string) in history list

If using AJAX the URL want change (single page application)
- Problems with bookmarks, forward/backward, favorites

Possible solution in HTML 5 hashchange event ...
- "...event, which occurs when the user clicks an in-page link that goes somewhere else on the same page, or when a script programmatically sets the location.hash property..."

- JQuery plugin for handling http://tkyk.github.com/jquery history-plugin/ (built on hashchange event, beta??)
- ...also a HTML5 JavaScript History API