What You, as an ASP.NET Developer, Need to Know About jQuery

E4D Solutions

Experts for Design Develop Debug Deploy

What we'll be looking at...

- Why jQuery?
- jQuery fundamentals
- Creating and manipulating elements
- Events
- Animations and effects
- Talking to the server
- jQuery UI
- Writing plugins
- Breaking news around new releases
- Using the CDN

ter these topics,
You'll say...

Throughout the session...

- You’ll see some

- Goal: show a particular place where jQuery really stands out

Hi, jQuery

- jQuery is
  - Most popular, cross-browser JavaScript library
  - Focusing on making client-side scripting of HTML simpler
    - Easy navigating the DOM
    - Handling events
    - Working with Ajax
  - Open-source, released in 2006

Why jQuery?

- Many JavaScript frameworks try bending the language out of its natural form
- jQuery aims at leveraging CSS, HTML and JavaScript
- Advantages
  - Lightweight
  - Easy to learn using familiar CSS syntax and intuitive
  - Many plugins available
  - Easy to extend and compatible
  - It’s on Microsoft’s radar
  - Rich community

You can read this, right?

```
$('#something').hide().css('background',
'red').fadeIn();
```

E4D Solutions

Experts for Design Develop Debug Deploy

You are not alone!

- Many LARGE companies use jQuery for their sites, including:

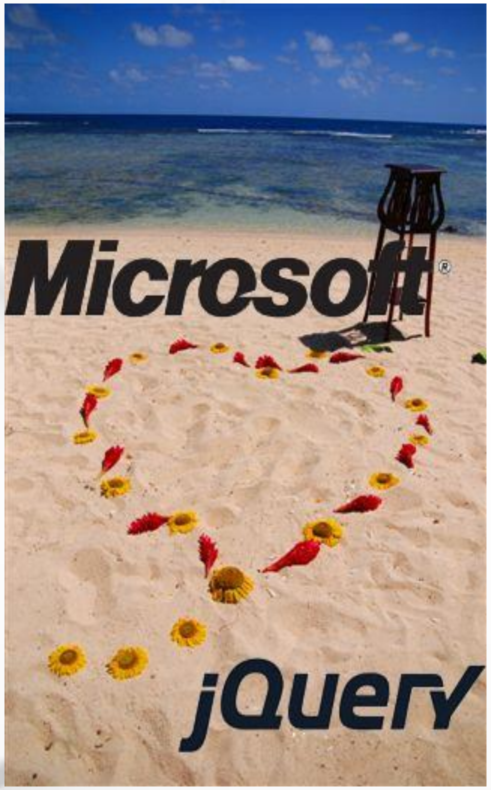
More at http://docs.jquery.com/Sites_Using_jQuery

E4D Solutions

Experts for Design Develop Debug Deploy

Microsoft and jQuery

- Included with Visual Studio in both WebForms and MVC projects
  - Can be used with or without ScriptManager
  - ScriptManager can be used to compress and combine scripts
  - IntelliSense available
  - CDN support (more later)
- Microsoft is contributor to jQuery
  - Proposed (and accepted) templating, data linking and globalization



Script, don’t get in my way!

- jQuery helps us writing *Unobstrutive JavaScript* code
- You don’t want to mingle style with HTML
- Why would you want to mingle *behavior* with HTML?

This code is downloaded every

```
<script type="text/javascript">
  window.onload = function() {
    document.getElementById('testButton').onclick = functio
      document.getElementById('xyz').style.color = 'red';
    };
  };
</script>
```

- This will become a heavy job without jQuery!

E4D Solutions

Experts for Design Develop Debug Deploy

jQuery fundamentals: $

- $ function (aka jQuery() function) returns
  - A JavaScript object containing an array of DOM elements
  - In the order they were found in the document
  - Matching a specified selector (for example a CSS selector)
  - Known to mankind as a wrapper or wrapped set

```
$("div.someClass").show();
```

Finds all D
Class some
Sets them v

- It returns the same group of elements, can be chained

```
$("div.someClass").show().addClass("SomeOtherClass");
```

To the same set, this
adds another class

jQuery fundamentals: the ready handler

- Script execution should wait until DOM elements are ready
  - You say: window.onload?
  - Sadly, this waits for everything to be loaded, including images etc
  - Script execution is *too late*
- Instead, we need to wait only until the DOM tree is created
  - Can be difficult in cross-browser situations
  - Easy-peasy with jQuery

I ♥ jQuery

```
$(document).ready(function(
) {

$("div.someClass").show();
});
```

```
$(function() {

$("div.someClass").sho
w();
});
```

jQuery fundamentals: selectors

- At the core of jQuery lies its selector engine
  - Can be used to select elements based on names, attribute, position...
- $() is heavily overloaded
  - Making a selection
  - Creating new HTML elements

jQuery fundamentals: selectors

- Most basic: CSS selectors

```
$("p a.someClass")
```

Selects all a's with someClass applied within a paragraph

  – Can be combined

```
$("p a.someClass, div")
```

Also includes all DIVs on the page

- Child selector

```
$("ul.someList > li > a")
```

Selects all links, directly in an LI, within an UL with someList class

E4D Solutions

Experts for Design Develop Debug Deploy

jQuery fundamentals: selectors

- Attribute selector

```
$("a[href*='http://www.snowball.be']")
```

Selects all links that contain a reference to my site

```
$("span[class^='some']")
```

Select all SPANs whose class attribute starts with some

```
$("span[class]")
```

Select all SPANs that have a class

E4D Solutions

Experts for Design Develop Debug Deploy

jQuery fundamentals: selectors

- Position

```
$("a:first")
```

Selects first A we can find on the page

```
$("div:even")
```

Selects the "even" DIVs on a page

```
$("table#someTable td:first-child")
```

Selects the first cells within a tab named someTable

- Psuedo-classes (CSS filter selectors & custom selectors)

```
$("input:checked")
```

Selects checked inputs (including the ones that weren't checked initially)

```
$(":password")
```

Selects all INPUTs of type password

```
$("input:not(:checked)")
```

Selects all 'not-checked' inputs

Experts for Design Develop Debug Deploy

More selectors

- Full list at http://www.w3.org/TR/css3-selectors/

Pattern	Meaning
*	any element
E	an element of type E
E[foo]	an E element with a "foo" attribute
E[foo^="bar"]	an E element whose "foo" attribute value begins exactly with the string "bar"
E:nth-child(n)	an E element, the n-th child of its parent
E:first-child	an E element, first child of its parent
E:empty	an E element that has no children (including text nodes)
E:link E:visited	an E element being the source anchor of a hyperlink of which the target is not yet visited (:link) or already visited (:visited)
E > F	an F element child of an E element
E + F	an F element immediately preceded by an E element

DEMO

Selecting elements using selectors

E4D Solutions

Experts for Design Develop Debug Deploy

jQuery fundamentals: creating elements

- $('…') selects an element <> $('<li>') creates an element
  - Attributes can be passed using JavaScript object

```
 $(function(){
        $('<div>I’m
off</div>')
       .appendTo('body');
```

```
 $(function(){
        $('<img>', {
            src: 'someImage.jpg',
            alt: 'Some nice image'
          })
      .appendTo('body');
```

E4D Solutions

Experts for Design Develop Debug Deploy

DEMO

Creating elements using $

E4D Solutions

Experts for Design Develop Debug Deploy

Working with the result of $

- Once we have a wrapped set, we can go wild with it!
  - Handle the set as a whole
  - Work with individual elements

Working with the result of $ (2)

- A wrapped set is like an array of elements, normal "array operations" can be done on it
  - Check the size

```
$('a').size();
```

  - Access an indiviual element

```
$('a') [0];
$('a').get(0);
```

  - Loop over the elements

```
$('img').each(function(n){
  this.alt='image['+n+']';
});
```

E4D Solutions

Experts for Design Develop Debug Deploy

Working with the result of $ (3)

- Set operations (continued)
  - Add and remove elements

```
$("a[class]").add("a[href]");
```

  - Filter elements

```
$("a").filter("[href^='http://']");
```

- Remember that we are always returning the set
  - Chaining is always possible!

```
$("a[class]")
      .add("a[href]")
      .filter("[href^='http://']")
      ...;
```

DEMO

Working with the set

Attributes

- When we want to change how an element looks, we can change its attributes
- jQuery provides the attr() method
  - 2 variations based on number and types of parameters
    - Read a specified property from first element in wrapped set

```
$("#myImage").attr("alt");
```

    - Set a property on all elements in the wrapped set (0 or more)

```
$('#myImage').attr('alt', ‘Me in Paris');
```

      - Can also accept a function
- Attr() helps us dealing with browser-dependencies (again)
  - jQuery float attribute refers to styleFloat in IE, cssFloat in others

```
$('a[href^=http://']).attr('target ', 'blank‘);
```

Attributes (2)

- jQuery makes it easy to apply and remove CSS classes
  - addClass(), removeClass(), toggleClass() and hasClass()
- Changing indiviual CSS elements is supported
  - css() can be used to get or set CSS on an element

```
$('#mydiv').css("background-color","yellow");
```

Working with elements

- html() can be used to get or set the content of an element

```
$('#mydiv').html();
```

Retrieves the HTML content Of mydiv

  - text() can retrieve combined textual content of all elements, including their children

- If the elements are form elements, we need to use val()

```
$('input:checkbox:checked').val();
```

Retrieves the value from a checked checkbox

E4D Solutions

Experts for Design Develop Debug Deploy

DEMO

Working with attributes

Events

- A bit of history
  - Once upon a time, a browser called Netscape introduced an event model, often referred to as DOM Level 0 Event Model
    - Creates event handlers as references to a function on a property
    - Not what we need if we want to create Unobtrusive JavaScript
    - Only one event handler per element for specific event
  - Only got standardized until DOM Level 2 Event Model
    - Based on a system of event listeners (addEventListener)
    - IE decided to go its own way (attachEvent)
  - Using event was a real mess because of browser dependencies

  - jQuery comes to the rescue

jQuery events

- bind() is where it all starts
  - Binds a function to any event on any DOM element

```
$('#myimg')
      .bind('click',
            function(event){alert(Hello TechEd!');}
      );
```

  - Works in any browser, jQuery hides the details for us
  - Possible to bind more than one event handler f

    event on on element
- one() removes itself after event handler executed

Live and let die

- bind() is OK for existing elements
- live() allows us to create event handlers for elements that don’t exist (yet)

```
$('.someClass')
      .live('click',
            function() {
            //do something
        });
```

If new elements match the selector,
they get this event handler as well

- die() removes the live()-created event handlers

```
$(".someClass").die("click")
```

DEMO

Events

Animations and effects

- Core jQuery has some basic effects
  - More are available in jQuery UI
  - Should be used with caution!
- Most basic ‘animation’ is hiding/showing an element
  - hide(): sets display:none on the element
  - show(): sets display to inline/block
  - toggle(): sets visible is hidden and vice-versa
- Methods are overloaded, accepting
  - Speed
  - Callback

Animations and effects (2)

- Elements can also be gradually added/removed
  - slideDown() and slideUp()
- Fading in is supported as well
  - fadeIn() and fadeOut()
- animate() is mother of all animations
  - Using ‘target values’ for style properties, jQuery will animate the transition

```
$('.someClass').animate({opacity:0.25},'slow');
```

Change the opacity with a slow animation

DEMO

Animations

E4D Solutions

Experts for Design Develop Debug Deploy

Ajax in the past

- When we were all young (in 1998), Microsoft introduced the ability to perform asynchronous requests from script (ActiveX)
- Later, other browsers implemented a standard, the XMLHttpRequest
  - IE6 uses an ActiveX object
- Result is that we need to do checks

Looks ugly, doesn't it?

```
if(window.ActiveXObject) {
  xhr = new
ActiveXObject("Microsoft.XMLHTT
P");
   }
else if (window.XMLHttpRequest)
{
  xhr = new XMLHttpRequest();
}
```

- Again... jQuery to the rescue!

Ajax with jQuery

- Basic functionality to load content from a server-side resource:
  - load()
    - url
    - parameters: data to be passed (string, object...). If provided, a POST is executed, otherwise a GET
    - callback (optional)

```
$('#someDiv')
      .load('test.html',
            function() {
                  alert('Load was performed.');
      });
```

Loads the HTML page

- Next to load, we can also use $.get()/$.getJson() or $.post()

DEMO

Basic Ajax request with load()

E4D Solutions

Experts for Design Develop Debug Deploy

Ajax with jQuery(2)

- If we need all control over the Ajax request we can get:
  - $.ajax()
    - options: defines an object containing all the properties for the Ajax request
- List of options is huge, therefore
  - $.ajaxSetup
    - options: defines an object containing all the properties for the Ajax request, becoming the default for Ajax requests

```
$.ajaxSetup({
  type: 'POST',
  timeout: 5000,
  dataType: 'html'
});
```

Ajax with jQuery(3)

- Throughout the Ajax request, we can get feedback
  - Local events from the $.ajax() call (callbacks)
  - Global events
    - Are broadcast to every element within the DOM, can be attached on any element
      - ajaxStart
      - ajaxSend
      - ajaxSuccess
      - ajaxError
      - ajaxComplete

DEMO

More control with ajax()

E4D Solutions

Experts for Design Develop Debug Deploy

jQuery Ajax, ASP.NET MVC and WebForms

- jQuery can work in harmony with ASP.NET MVC and WebForms
  - Sample ajax() call for WebForms

```
$.ajax({
      type: "post",
      contentType: "application/json; charset=
      dataType: "json",
      url: "/Default.aspx/AddTask",
      data: JSON.stringify(dto)
});
```

E4D Solutions

Experts for Design Develop Debug Deploy

DEMO

ASP.NET WebForms with jQuery

DEMO

ASP.NET MVC with jQuery

jQuery UI

- Huge extension of jQuery, providing more UI capabilities
- Contains number of UI features we'd logically need
- Includes
  - Effects: more advanced than core effects
  - Interactions: drag and drop
  - Widgets (aka controls): date picker...
  - All can be themed
- jqueryui.com contains tool to configure download and "ThemeRoller" tool
- Code included in jquery-ui.js

Effects

- jQuery core contains some basic effects
- Based on the effect(type, options, speed, callback) method
  - Has several animation types such as puff, highlight and shake (even explode exists)
  - Also allows to do animations with colors (not possible with animate())
    - backgroundColor, color...
- Visibility methods (show()...) are extended
- Class methods (addClass()...) are extended
- position() method is added for advanced positioning

```
$('#someElement').position({
  my: 'top center',
  at: 'bottom right',
  of: '#someOtherElement'});
```

DEMO

Effects

Interactions

- Interactions focus on allowing users to directly interact with elements, which isn't possible with standard HTML controls
  - They add advanced behaviors to our pages related to mouse interactions
- Available interactions:
  - Dragging
  - Dropping
  - Sorting
  - Resizing
  - Selecting

Dragging

- Easy-peasy (again) with jQuery
- draggable() is your friend (heavily overloaded once again)
  - Allows making elements draggable, possible with options (opacity...)

```
$('#someDiv').draggable();
```

  - Overloaded so it also support enabling, disabling... Draggable

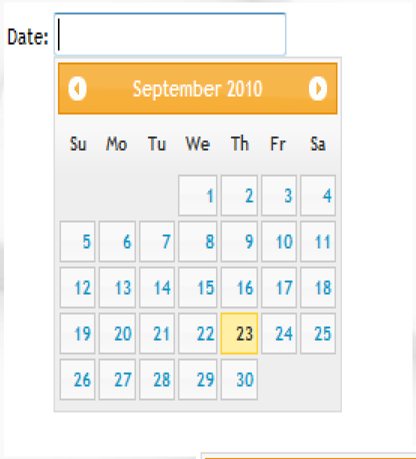
```
$('.disableMe').draggable('disable');
```

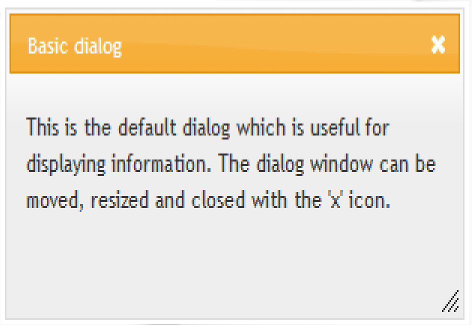
DEMO

Dragging, dropping and other interactions

Widgets: controls on steroids

- New controls (based on existing ones)
- Contents
  - Buttons
  - Sliders
  - Progress bars
  - Autocompletion
  - Date picker
  - Tabs
  - Accordion
  - Dialog box





This is the default dialog which is useful for displaying information. The dialog window can be moved, resized and closed with the 'x' icon.

DEMO

Widgets in action

Something missing in jQuery?

- 2 options:
  - Use an existing plugin
    - jQuery plugin repository: plugins.jquery.com
    - Google code: code.google.com
    - SourceForge: sourceforge.net
    - GitHub: github.com
  - Write a plugin yourself
    - Custom utility function
    - Create wrapper functions

Writing your own plugins

- Write a plugin to add it yourself!
  - Possible to write your own utility functions and wrapper methods
- Creating new wrapper methods:
  - Add the method as a property on the fn object in the $ namespace

```
$.fn.wrapperFunctionName =
function(params){function-body};
```

```
(function($){
  $.fn.setToRed = function() {
    return this.css('color','red');
  };
})(jQuery);
```

We are passing jQuery to a function that has $ as parameter. This way, $ will inside this function, always be the jQuery $

DEMO

Writing a plugin

Breaking news!

- **October 4th 2010**: jQuery has accepted 3 plugins from Microsoft
  - jQuery Templates
  - jQuery Data Link
  - jQuery Globalization
- Are now official plugins
- Templates were supposed to be standard part of next major jQuery version
  - Didn’t happen though...

jQuery Templates

- Template is HTML markup (containing tags)

```
<script id="movieTemplate" type="text/x-jquery-tmpl"
   <li><b>${Name}</b> (${ReleaseYear})</li>
</script>
```

- 3 plugins:
  - .tmpl(): renders the template
  - .tmplItem(): find the template item
  - .template(): compile the template

```
$("#movieTemplate").tmpl(movies)
                .appendTo("#movieList");
```

jQuery Templates (2)

- Container for the template can be
  - Inline markup
  - String (computed or downloaded)
- Can be retrieved using

```
$( selector ).tmpl( data )
```

  - Selector is container (div...)
    - Can result in invalid HTML (due to tags)
    - Browser may start to load in this HTML
  - Best to place it within script tag

```
<script id="myContainer" type="text/x-jquery-tmpl"
```

.tmpl()

- Take the first element in the matched set and render its content as a template, using the specified data
- .tmpl([data], [options])
  - Data: The data to render. Can be any JavaScript type, including Array or Object
  - Options: An optional map of user-defined key-value pairs

```
$( "#myTemplate" ).tmpl( myData )
```

- Can be used with local data, mostly remote data (ajax)

DEMO

jQuery Templates

jQuery Data Link

- Data linking is the data binding engine of jQuery
  - Allows linking a field of an object to a field of another object
- Available through the .link() function

```
var person = {};
$("form").link(person);
```

  - When a value of a field changes, the property on the object is changed as well
    - Links are two-way by default

DEMO

jQuery Data Link

jQuery Globalization

- Includes globalization information for over 350 (!) cultures
  - Currencies
  - Date formatting (month names)
  - Number formatting
- Uses the standardized culture identifications
  - Ex. nl-BE ← a great culture ;-)

DEMO

jQuery Globalization

Where to get your stuff?

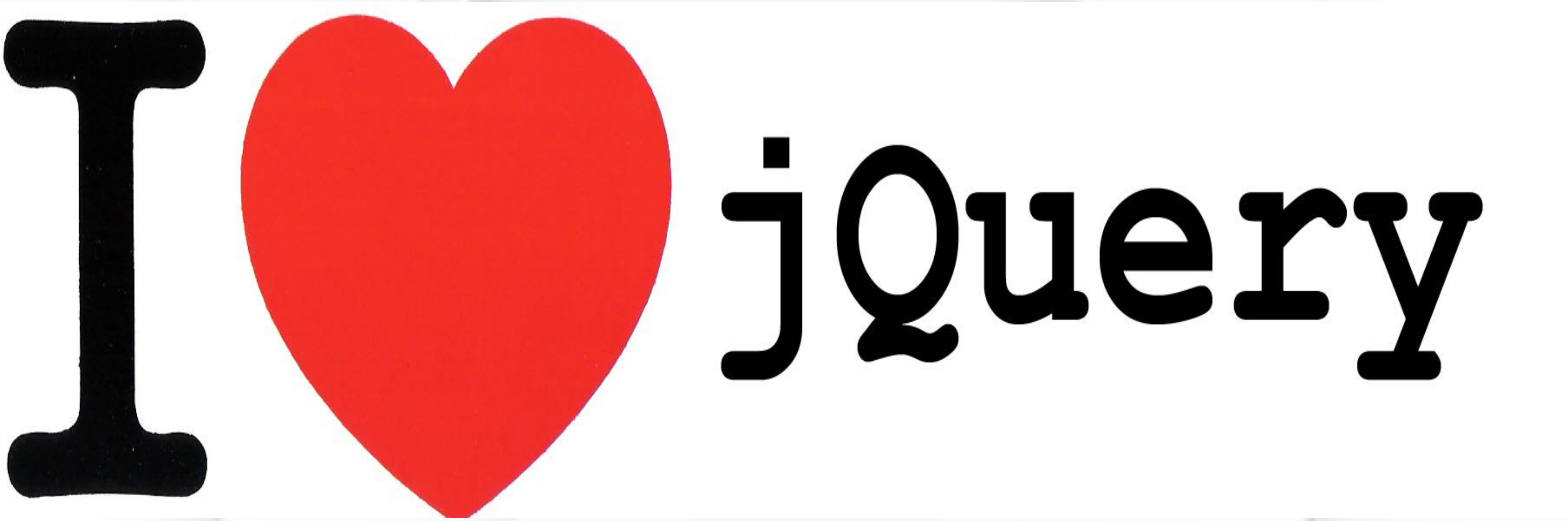
- Use a CDN?
  - Microsoft
  - Google
- Put scripts locally as well with a fallback mechanism

```
<script type="text/javascript"
      src="http://ajax.microsoft.com/ajax/jquery/jquery-1.4.2.mi
</script>
<script type="text/javascript">
if (typeof jQuery == 'undefined')
{
    document.write(unescape("%3Cscript src='/Scripts/jquery-1.4.2
      type='text/javascript'%3E%3C/script%3E"));
}
</script>
```

Summary

- Where does all the (l) for jQuery come from?
  - Light-weight library that uses JavaScript as JavaScript, relying on CSS
  - Cross-browser compatible, hides the details (ready handler)
  - Easy eventing model
  - Can work with MVC & WebForms
  - Easily extensible to fit your needs, tons of plugins already available

So I hope you now say too...



Thanks!

E4D Solutions

Experts for Design Develop Debug Deploy