



JQuery For Beginners

06.19.2010

Prepared for CTDOTNET Code Camp

Who am I? Who are we?

- Mike Linnetz, Director of Technology and Applications, Acsys Interactive
 - MikeL@acsysinteractive.com
- Acsys Interactive (www.acsysinteractive.com)
 - 15+ years
 - Full service agency offering strategy, marketing, creative, technology, analytics
 - Based out of Farmington, CT, with offices in NYC & Boston
 - Privately held, 50+ full time staff
- Lunch and Learns
 - Today's presentation is part of our lunch and learn series on a variety of technical topics
 - We will soon be opening up our tech lunch and learns to the public
 - Interested in updates? Email Dan Terrill dant@acsysinteractive.com or myself (take a card)
- Questions Welcome

JQuery – What and Why?

- **From jquery.com** †:
 - jQuery is a prepackaged JavaScript Library that simplifies DOM interaction, Ajax, callbacks, and animations.
 - Lightweight Footprint About 24KB in size (Minified and Gzipped)
 - CSS3 Compliant Supports CSS 1-3 selectors and more!
 - Cross-browser IE 6.0+, FF 2+, Safari 3.0+, Opera 9.0+, Chrome
- **Two Versions**
 - Development – jquery-X.X.X.js - human-readable (although I've never had the need)
 - Production – jquery-X.X.X.min.js – “minified” and not readable
 - You can either download and host locally or reference via a number of CDNs
 - (http://docs.jquery.com/Downloading_jQuery#CDN_Hosted_jQuery)

† <http://jquery.com/>

Ready, Set, Document

- `$(document).ready()`
 - JQuery's way of letting you execute code the instant that the DOM has finished loading
 - Similar to `window.onload()`, but better

```
$(document).ready(function {  
    alert('Hello World');  
});
```

OR

```
$(document).ready(doABunchOfStuff);  
function doABunchOfStuff () {  
    alert('Hello World');  
}
```

OR

```
$(function() {  
    alert('Hello World');  
});
```

OR

```
jQuery(document).ready(function() {  
    alert('Hello World');  
})
```

Selectors

- Selectors allow you to get an object reference to the specific controls on the page that you want to manipulate.

<code>\$('#porridge')</code>	Selecting by ID equivalent to <code>document.getElementById('porridge')</code>
<code>\$('tr')</code> <code>\$('input')</code> <code>\$('div')</code>	Selecting by Tag
<code>\$('.cold')</code>	Selecting by Class
<code>\$('#tr.cold')</code>	Selecting by Tag and Class
<code>\$('#div, #porridge, .cold')</code>	Mix and match

<code>\$('#div#porridge')</code>	Selecting by Tag <i>and</i> ID?! UNNECESSARY! Why?
----------------------------------	--

Selectors

- **Selecting Descendents Of**

- `$('#porridge tr')` - gives me an array of all the `<tr>` elements within the `#porridge` element

- **Selecting Descendents With a Specific Class**

- `$('#porridge tr.justright')` - gives me an array of all the `<tr>`s within the `#porridge` element which have a class of “justright”

- `$('#div.pod p')` - peas in a pod?

- **Selecting *Direct* Descendents Of**

- Consider the following HTML

- ```
<div id="fruitbowl">
 Apple
 Pear
 <div>BananaPeelBanana</div>
</div>
```

- `$('#fruitbowl strong')` - selects all the fruits

- `$('#fruitbowl > strong')` - selects only the pear and apple


# More exciting selectors

<code>\$('#myTable tr:first')</code>	returns the first <tr> in myTable
<code>\$('#myTable tr:last')</code>	returns the last <tr> in myTable
<code>\$('#myTable tr').eq(2)</code>	gives me the third <tr> (zero based index)
<code>\$('#myTable tr').nth-child(3)</code>	gives me the third <tr> (NOT zero based index)
<code>\$('#table:lt(1)')</code>	returns the first two tables (zero based index, "lt" = less than or equal to)
<code>\$('#myTable tr:not(first)')</code>	returns all <tr>s in myTable except the first
<code>\$('#myTable tr:not(first):not(last)')</code>	returns all <tr>s in myTable except the first and last
<code>\$('#porridge tr td:contains("justright")')</code>	returns the td which contains the text "justright" (case sensitive) as its contents

# Time to do something

- `.css()` – allows access to CSS properties

```
var width= $('#myDiv').css('width');
$('#myDiv').css('width','300');
$('#myTable tr:even').css({'background-color':'#000000', 'color':'#FFFFFF'});
```
- Adding/Removing Classes

```
$('#myDiv').addClass('redfish');
$('#myDiv').removeClass('vanilla strawberry');
 $('#myDiv').toggleClass('RedFish BlueFish');
```
- `.text()` – gets or sets the inner text

```
var divContents = $('#myDiv').text();
```

(only gets the raw text, including descendents)
- `.html()` – gets or sets the inner HTML

```
$('#myDiv').html('Yellow World!');
```
- `.attr()` – gets or sets the specified attribute

```
var divWidth = $('#myDiv').attr('width');
$('#img').attr('width', divWidth);
```

# Arrays of DOM Elements

- Functions called on the array affect all members

```
$('.icecream').addClass('cherry');
```

- These arrays also have properties available to you

`.length` - count of the number of elements

```
if ($("#myUnorderedList li").length > 0) { /* do something */ }
```

- More importantly, JQuery makes many functions available to you for working with the elements

- `.each (<function>)`

– Executes a function once for each member. Use the “this” keyword to refer to the current element

```
$('.img').each(function() {
 $(this).attr('alt', $(this).attr('src'));
});
```

# What else?

`.hover(<handlerIN>, <handlerOUT>)` – lets you define mouseover and mouseout handlers

```
$("#p").hover(
 function () {
 $(this).addClass("glow");
 },
 function () {
 $(this).removeClass("glow");
 }
);
```



`.click(<function>)` - defines an “on click” function, similar to the onclick attribute of any element

```
$("#a.AnchorWithAttitude").click(function { alert('Stop clicking me'); });
```

```
$("#div").click(function { alert('I don`t know why you`d ever want to have every
div on the page do something on click.') });
```

# Attribute Selectors

<code>[name]</code>	Selects elements that have the specified attribute defined.	<code>\$('img[alt]')</code>
<code>[name=value]</code>	Or defined with a value equal to a certain string.	<code>\$('input[type=text]')</code>
<code>[name!=value]</code>	Select elements that either don't have the specified attribute, or do have the specified attribute but not with a certain value.	<code>\$('input[type!=text]')</code>
<code>[name^=value]</code>	Selects elements that have the specified attribute with a value beginning with a given string.	<code>\$('img[id^=product]')</code>
<code>[name\$=value]</code>	Selects elements that have the specified attribute with a value ending with a given string.	<code>\$('img[src\$=png]')</code>
<code>[name*=value]</code>	Selects elements that have the specified attribute with a value containing the given string.	<code>\$('img[src*=goldilocks]')</code>
<code>[selector][selector]</code>	You can also string the selectors together.	<code>\$('img[src*=goldilocks][src\$=png]')</code>

# Filters

- **filter(<expression or function>)**

- Allows you to specify an expression or a function to match elements against.

```
$('#div').filter(':first, .justright')
```

- This filter expression takes all divs and filters it down to the first div plus any div with a class of “justright”

```
$('#ul').filter(function(index) {
 return $('li', this).length == 2;
})
```

- Returns only those `<ul>`s on the page which have exactly 2 `<li>`s .
- The index passed to the filter function refers to the index of each DOM element (I believe you have to pass this even if you don't use it)
- The “this” refers to the `<ul>` DOM element in the array returned by the outer expression

# Effects and Animation

- `.hide()`, `.show()`, `.toggle()`
  - Hides and shows the element, similar to “display:none” and “display:block”
- `fadeOut()`, `fadeIn()`
  - Fades elements to transparency and then back
- `slideUp()`, `slideDown()`, `slideToggle()`
  - Hides/shows elements by sliding them up and down



## Callbacks

- Some JQuery functions allow you to specify a function to call once the animation is complete.
- The callback is called once for the entire animation, not once for each element (e.g., in case multiple elements are affected by the same animation)
- `.append()`
  - Inserts an element or a string of HTML at the end of the element

```
$('#myDiv').append('<p>Tastes like chicken</p>');
```




## `.animate()`

- Allows you to define a custom animation based on *numerical* CSS properties

# setTimeout()

- `setTimeout()`

-  The javascript `setTimeout()` method calls a function after a specified number of milliseconds, same as the javascript

```
$(document).ready(function() {
 setTimeout(screwAroundNoPunIntended(), 10000);
});
```

```
function screwAroundNoPunIntended() {
 window.location = 'http://www.yourfavoritepornsite.com';
}
```

# Effects and Animation

- Draggable and droppable

- Part of the UI extension for JQuery (a separate download, jquery-ui-X.X.X.custom.min.js)
- Makes certain elements “draggable”, allowing you to easily drag them across the screen
- Makes other elements as “droppable”, whereby any draggable element which is dropped onto the droppable will trigger an event.

```
$(document).ready(function() {
 $("#divDrag").draggable();

 $("#divDrop").droppable({
 drop: function(ev, ui) {
 var drag = $(ui.draggable).attr("id");
 alert(drag + ' was just dropped onto me.'); }
 });
});
```



```
<DIV id="divDrag">Drag me around</DIV>
<DIV id="divDrop">Drop onto me</DIV>
```

# REFERENCES

- <http://jquery.com/>
- <http://www.learningjquery.com/2006/11/how-to-get-anything-you-want-part-1>
- <http://www.learningjquery.com/2006/12/how-to-get-anything-you-want-part-2>

# Tidbits

- Mixing with javascript - JQuery is just a javascript library. You can mix any JQuery statement with javascript.

```
document.getElementById("myDiv").innerText = $("#myTextbox").text();
```

- Remember, most of the things you're doing with JQuery are client side, so the changes won't show up on a View Source (which only shows you what the page looked like after the initial web request). This is just a thought when you're debugging.
- Some other UI libraries (e.g., Spry ) also make use of "\$" as a variable/reserved symbol. `JQuery.noConflict()` allows you to run other libraries that might use the \$ variable.
- † *"Many JavaScript libraries use \$ as a function or variable name, just as jQuery does. In jQuery's case, \$ is just an alias for jQuery, so all functionality is available without using \$. If we need to use another JavaScript library alongside jQuery, we can return control of \$ back to the other library with a call to \$.noConflict():"*

† <http://jquery.com/>