

# JavaScript Workshop

---

jQuery Basics pt. 1

# jQuery Library

- Created by John Resig; similar to the Prototype library, but smaller and not as extensive.
- Emphasizes short, efficient coding techniques.
- Can access parts of a web page, modify elements on a page, respond to user events, create animation effects and contains Ajax functionality.
- Ability to extend its functionality with plugins.
- jQuery UI (**Sept 2007**), built on top of jQuery to provide widgets, mouse interactions, and mouse effects.

# Getting started

- Download the latest version:  
[http://docs.jquery.com/Downloading\\_jQuery](http://docs.jquery.com/Downloading_jQuery)

- Include the jquery.js in your web page:

```
<script type="text/javascript" src="jquery-1.2.1.js" script>
```

- Like many libraries, jQuery is built around the `$()`. There are many argument types that it can accept:

*`$("#id")` – finds element with by ID*

*`$("p")` – finds all `<p>` elements*

*`$("div > p")` – finds all `<p>` within a `<div>`*

*`$("span:contains(text)")` – finds a `<span>` with specific text*

# Chainability (The Magic of jQuery)

- jQuery uses an interesting concept to make its code short and simple. In a nutshell: Every method within jQuery returns the jQuery object itself, allowing you to 'chain' upon it.

1) adds a class "test" to every <a>

```
$("#a").addClass("test");
```

2) Adds CSS styles to a element with id of #myDIV

```
$("#myDiv").css("border", "3px solid red");
```

3) Adds a class of "red" to the 3<sup>rd</sup> <div>

```
$("#div").eq(2).addClass("red");
```

4) Changes text of the first paragraph

```
$("#p").eq(0).text("<strong>Some</strong> new text.");
```

# Launching Code on Document Ready

- The first thing that most Javascript programmers end up doing is adding some code to their program, similar to this:

```
window.onload = function() { ... }
```

- Problematically, however, the Javascript code isn't run until all images are finished downloading (this includes banner ads). The reason for using `window.onload` in the first place is due to the fact that the HTML 'document' isn't finished loading yet, when you first try to run your code. jQuery has a simple statement that you can use, known as the ready event:

```
$(document).ready (function() { // Your code here });
```

# The `$()` Factory Function

- No matter what type of selector you want to use; it always starts with the `$()` function. The `$()` function removes the need to do a for loop to access a group of elements since whatever you put inside the parentheses will be looped through automatically and stored as a jQuery object.

**A tag name:** `$("p")` all paragraphs in the document

**An ID:** `$("#some-id")` element in document that has the ID

**A class:** `$(".come-class")` elements in document that have that class

# CSS Selector Expressions

- jQuery offers a powerful set of selector expressions for matching a set of elements in a document. Some examples are:

**Element:** `$("#div")` have a tag name of `<div>`

**ID:** `("#myID")` have an ID of `"#myID"`

**Class:** `(".myclass")` have a class name of `".myclass"`

**Descendant:** `("#container p")` `<p>` descendants of `"#container"`

**Child:** `E > F` `("li > ul")` `<ul>` children of a matched `<li>`

**Adjacent Sibling:** `("ul + p")` `<p>` immediately following `<ul>`

**Multiple Elements:** `("em strong p")` match all elements

**First Child:** `("li:first-child")` `<li>` that is first child

**Last Child:** `("li:last-child")` `<li>` that is first child

Descendant - can be a child, grandchild or great-grandchild.

Adjacent – has same parent but not child

# Attribute Selector Expressions

- You can match on attribute values beginning with the @ symbol:

**Has Attribute: \$("a[@rel]")**

matched by <a> that has an rel attribute

**Value Equals: \$("a[@rel=frank]")**

<a> with rel attribute = "frank"

**Not Equal: \$("a[@rel!=frank]")**

<a> with rel attribute not containing "frank"

**Value Begins: \$("a[@rel^=address]")**

rel attribute value beginning with "address"

**Value Ends: \$("a[@rel\$=address]")**

rel attribute value ends with "address"

**Value Contains: \$("p[@class=\*frank]")**

<p> with class value containing "frank"

# Custom Selector Expressions

- These selectors were added as an attempt to address common DOM traversal needs not met by other expressions.

## **Even Element (:even) Odd Element (:odd):**

`$("li:even")` – matched `<li>` that have an even index value

`$("tr:odd")` – matched `<tr>` that have an odd index value

## **Nth Element (:eq(n), :nth(n)):**

`$("li:eq(2)")` – selects the third `<li>` element

`$("p:nth(1)")` – selects the second `<tr>` element

## **Greater Than :gt(n):**

`$("li:gt(1)")`: selects all `<li>` after the second one

## **Contains :contains(text):**

`$("p:contains(frunk)")`: selects all `<p>` containing "frunk" text

- JavaScript Workshop website:  
<http://www.javascriptworkshop.com>
- Any questions?  
[frank@javascriptworkshop.com](mailto:frank@javascriptworkshop.com) - Frank Stepanski