**CS 325 - Introduction to Web Programming**

authored by Zadia and Dale

**Topic 7 – JQuery**

**Part 1**

* jQuery is a JavaScript library, to modify elements on a webpage and improve user interaction.
* JQuery philosophy: “write less, do more”
* The jQuery library contains the following features:
* HTML/DOM manipulation
* CSS manipulation
* HTML event handling
* Special effects and animations
* AJAX handling
* Utility methods
* Why jQuery?

There are lots of other JavaScript frameworks out there, but jQuery seems to be the most popular, and also the most extendable.

Many of the biggest companies on the Web use jQuery, such as:

* Google
* Microsoft
* IBM
* Netflix

That's why I’m teaching it, even though it isn't in the textbook. The textbook teaches two other less popular libraries, that you are welcome to use, but I won't go over in class. If you use other libraries in your labs and projects, make sure that I have a copy so that I can grade them.

* **Adding jQuery to your pages**

**Option 1**: Installing JQuery

Use the Google-hosted content delivery network (CDN) to include a version of

jQuery in a script in the head section.

<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>

**Option 2:** Download your own copy of jQuery from jQuery.com and host it on your own server or local filesystem.

1. Go to jquery.com.

2. Download the current version.

3. Store the resulting .js file to your working directory.

(jQuery-3.2.1.min.js is the current file as of December 2017)

To incorporate the local copy of the library in your pages, simply add it as an external JavaScript file in the head section:

<script src="jquery-3.1.1.min.js"></script>

**jQuery Syntax has two forms:**

**$(*selector*).*action*()**

* A $ sign to define/access jQuery
* A (*selector*) to "query" (or find) HTML elements
* A jQuery *action*() to be performed on the element(s)

**$(*selector*).*event*(*action*)**

* A $ sign to define/access jQuery
* A *selector* to "query" (or find) HTML elements
* An *event* that triggers the action
* An *action* to be performed on the element(s)

**Example**

//first form -- hides all <p> elements immediately

$("p").hide()

//second form -- hides all <p> elements when any one is clicked

$("p").click(function() { $("p").hide(); })

* **The Document Ready Event**

All jQuery methods are inside a document ready event.

$(document).ready(function(){  
   *// jQuery methods go here...*  
});

You can also write it this way:

$(function(){  
   *// jQuery calls go here...*  
});

This is to prevent any jQuery code from running before the document is finished loading ("is ready") – replaces *window.onload = pageLoad;*

**jQuery Selectors**

jQuery selectors allow you to select and manipulate HTML element(s). HTML elements based on their name, id, classes, types, attributes, values of attributes, etc

**Example 1**

Selecting all paragraphs: $("p")

When a user clicks on a button, all <p> elements will be hidden:

$(document).ready(function(){  
    $("button").click(function(){  
        $("p").hide();  
    });  
});

**Example 2**

// the code executes a browser alert( ) that informs us that the page contains six <a> elements

$(function(){

alert('Page contains ' + $('a').length + ' <a> elements!');

});

* Alternatively, you can pass the jQuery function an actual JavaScript reference to the DOM elements:

$(function(){

alert('Page contains ' +

$(document.getElementsByTagName('a')).length + ' <a>

Elements');

});

* **The ID selector**
* The jQuery #id selector uses the id attribute of an HTML tag to find the specific element. The id should be unique.

**Example 3**

$(document).ready(function(){

$("button").click(function(){

$("#test").hide();

});

});

* **The Class Selector**
* The jQuery class selector finds elements with a specific class.

To find elements with a specific class, write a period character, followed by the name of the class:

$(".test") //period instead of hash

**Example 4**

$(document).ready(function(){

$("button").click(function(){

$(".test").hide();

});

});

See <https://www.w3schools.com/jquery/jquery\_ref\_selectors.asp> for a list of all the different selectors possible with jQuery.

**Part 2**

**Class Exercise**

Hide the button named “Click Me” when it is clicked.

Hint: use $(this)

"this" refers to the current element (same as in Java)

Solution: $( "button:contains('Click Me')" ).click(function() { $(this).hide(); })

**jQuery Events**

Events are the main method of communication between a user and a website or web

application. Most of our JavaScript/jQuery coding will be run in response to a variety

of user and browser events.

Examples:

* moving a mouse over an element
* selecting a radio button
* clicking on an element

**Example 5**

click(*function*) **-** The function is executed when the user clicks on the HTML element.

$(document).ready(function(){

$("p").click(function(){

$(this).hide();

});

});

Here are some common DOM events:

|  |  |  |  |
| --- | --- | --- | --- |
| **Mouse Events** | **Keyboard Events** | **Form Events** | **Document/Window Events** |
| click | keypress | submit | load |
| dblclick | keydown | change | resize |
| mouseenter | keyup | focus | scroll |
| mouseleave |  | blur | unload |

List of events: <https://developer.mozilla.org/en-US/docs/Web/Events>

**Class Exercise**

Change the background color of an h2 element when your mouse hovers over and change it again when your mouse leaves it.

* **jQuery Actions**

List of actions in jQuery: <http://api.jquery.com/>

1. **Hide/Show**

* Hide and show HTML elements with the hide() and show() methods

1. **Animate() -** create custom animations.

## Example 12 - Manipulate Multiple Properties

$(document).ready(function(){

$("#para2").click(function(){

$("#para2").animate({

left: '250px',

opacity: '0.5',

height: '150px',

width: '150px'

});

});

});

You can also use relative values (the value is then relative to the element's current value). This is done by putting += or -= in front of the value.

e.g.

left: '250px',  
        height: '+=150px',  
        width: '+=150px'

You can even specify a property's animation value as "show", "hide", or "toggle":

e.g. height: 'toggle'

1. **stop() -** used to stop animations or effects before it is finished.
2. **Callback -** executed after the current effect is 100% finished

**Example 13 - with callback**

$(document).ready(function(){

$("button").click(function(){

$("#callbk").hide("slow", function(){

alert("The paragraph is now hidden");

});

});

});

**Example 14 - without callback**

$(document).ready(function(){

$("button").click(function(){

$("p").hide(1000);

alert("The paragraph is now hidden");

});

});

1. **Chaining -** Chaining allows us to run multiple jQuery methods (on the same element) within a single statement.

**=========================================================**

e.g. $("#p1").slideUp(2000).slideDown(2000);

**Part 3 – Using jQuery to Manipulate DOM Elements**

**Get Content**

Methods for DOM manipulation are:

* text() - the text content of selected elements
* html() - the content of selected elements (including HTML markup)
* val() - the value of form fields

**Example 15**

$(document).ready(function(){

$("#btn1").click(function(){

alert("Text: " + $("#test1").text());

});

$("#btn2").click(function(){

alert("HTML: " + $("#test1").html());

});

$("#btn3").click(function(){

alert("Value: " + $("#test2").val());

});

});

## Get Attributes

## attr() - get attribute values.

**Example 16 -  get the value of the href attribute in a link**

$(document).ready(function(){

$("#btn4").click(function(){

alert($("#cs").attr("href"));

});

});

*Similarly, you have methods for Set Content and Set Attribute.*

**Class Exercise 4**

Change the value of the alt attribute for Ollie & Roses to "cat and flowers" from "Panda".

**Add New HTML Content**

Methods that are used to add new content:

* append() - Inserts content at the end of the selected elements
* prepend() - Inserts content at the beginning of the selected elements
* after() - Inserts content after the selected elements
* before() - Inserts content before the selected elements

**Example 17**

$(document).ready(function(){

$("#btn5").click(function(){

$("ol").append("<li>Appended item</li>");

});

$("#btn6").click(function(){

$("ol").prepend("<li>Prepended item</li>");

});

$("#btn7").click(function(){

$("img").before("<b>Before</b>");

});

$("#btn8").click(function(){

$("img").after("<i>After</i>");

});

});

## Remove Elements/Content

* remove() - Removes the selected element (and its child elements)
* empty() - Removes the child elements from the selected element

**Class Exercise 5**

HTML

<p>I will stay</p>

<p class="test">I will go because my class="test".</p>

<p class="demo">I will go because my class="demo".</p>

<button id="btn9">Remove all p elements with class="test" and class="demo"</button>

Write JQuery code to remove all paragraphs with class test and demo.

**Solution: Code Example 18**

**Use jQuery to Manipulate CSS**

Methods:

* addClass() - Adds one or more classes to the selected elements
* removeClass() - Removes one or more classes from the selected elements
* toggleClass() - Toggles between adding/removing classes from the selected elements
* css() - Sets or returns the style attribute

**Example 19**

$(document).ready(function(){

$("#btn10").click(function(){

$("h1, h2, p").addClass("red");

$("div").addClass("important");

});

$("#btn11").click(function(){

$("h1, h2, p").removeClass("red");

});

$("#btn12").click(function(){

$("p").css({"background-color": "yellow", "font-size": "200%"});

});

});

**jQuery Dimension Methods**

jQuery has several important methods for working with dimensions:

* width() - sets or returns the width of an element (excludes padding, border and margin).
* height() - sets or returns the height of an element (excludes padding, border and margin).
* innerWidth() - returns the width of an element (includes padding).
* innerHeight() - returns the height of an element (includes padding).
* outerWidth() - returns the width of an element (includes padding and border).
* outerHeight() - returns the height of an element (includes padding and border).



**Example 20**

$(document).ready(function(){

$("#btn13").click(function(){

var txt = "";

txt += "Width of div: " + $("#div5").width() + "</br>";

txt += "Height of div: " + $("#div5").height()+ "</br>";

txt += "Inner width of div: " + $("#div1").innerWidth() + "</br>";

txt += "Inner height of div: " + $("#div1").innerHeight()+ "</br>";

txt += "Outer width of div: " + $("#div1").outerWidth() + "</br>";

txt += "Outer height of div: " + $("#div1").outerHeight()+ "</br>";

$("#div5").html(txt);

});

});

# Traversing the DOM

* parent() - returns the direct parent element of the selected element.
* parents() - returns all ancestor elements of the selected element, all the way up to the document's root element (<html>)
* parentsUntil() - returns all ancestor elements between two given arguments.
* children() - returns all direct children of the selected element
* find() - returns descendant elements of the selected element, all the way down to the last descendant.
* siblings() - returns all sibling elements of the selected element.
* next() - returns the next sibling element of the selected element
* nextAll() - returns all next sibling elements of the selected element.
* nextUntil() - returns all next sibling elements between two given arguments.
* prev()
* prevAll()
* prevUntil()

**DOM Tree**

<div class="container">

<h1>Hello World</h1>

<p>This is a <em>simple paragraph</em>.</p>

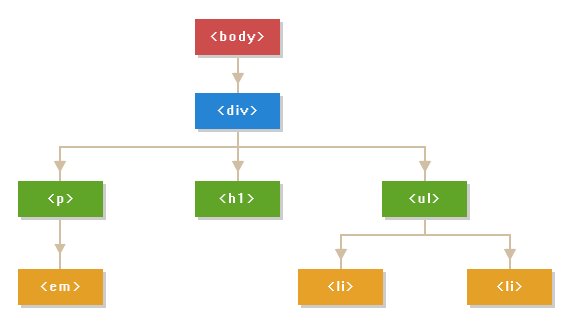
<ul>

<li>Item One</li>

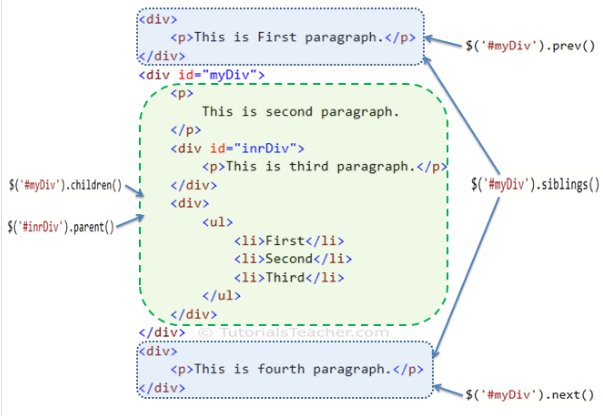
<li>Item Two</li>

</ul>

</div>



**How jQuery get DOM elements?**



**Example 21 – Traversing the DOM**

$(document).ready(function(){

returns all sibling elements between a <h2> and a <h6> element

$("h2").nextUntil("h6").css({"color": "red", "border": "2px solid red"});

returns all next sibling elements of <h2>

$("h2").nextAll().css({"color": "red", "border": "2px solid red"});

returns all sibling elements of <h2> that are <p> elements

$("h2").siblings("p").css({"color": "red", "border": "2px solid red"});

returns the direct parent element of <h2> elements

$("h2").parent().css({"color": "red", "border": "2px solid red"});

});

**jQuery Filter and Not methods**

jQuery has many selectors but you may want something like hide all ul elements that are of class .foo and also does not have an attribute named "key". There is no selector for such an element. Luckly jQuery has two methods that might help: filter and not:

$(document).ready(function() {

$("ul").filter(".foo").not("[key]").hide();

}

Why not just use classes to handle this situation? That is, why not create a class that contains exactly those UL elements that are of class foo and do not have a key attribute?

[It adds a lot of classes to the CSS code that could more easily be handled by this one line of jQuery. Also, you might forget one UL when adding the classes in the HTML.]