# The City Lit Institute

##### Department of Computing

## Keeley Street, Holborn, London WC2B 4BA

**jQuery**

(The JavaScript Library)

**LECTURER**: **ALEXANDER ADU- SARKODIE**

MSc. Telematics (IT & Telecom), MSc. Eng., Dip. Russ. Lang., Teach. Cert, AMIAEng (UK), MBCS (UK), MIfL(UK)

**Biopic:** [**http://www.goldhawk-college.com/istudio/**](http://www.goldhawk-college.com/istudio/)

**Blog:** [**http://www.blogger.com/profile/14800490193632788559**](http://www.blogger.com/profile/14800490193632788559)

**jQuery**

**jQuery** is an open source multi-browser JavaScript library, allowing you to **write less and do more.** It is designed to simplify client side scripting of HTML. It was released in January 2006 by **John Resig.**

jQuery is a fast and concise lightweight JavaScript Library with functions that simplifies HTML **document traversing**, **event handling**, **animating**, and **Ajax** interactions for rapid web development (RAD).

The jQuery library can be added to a web page with a **single line of markup**. Itis **free, open source software**.

The library has the following properties:

* About 31 kb size (Minified and Gzipped)
* Supports CSS1-3 Selectors
* Cross browser compatible :
* IE6.0+, FF 3.6+, Safari 5.0+, Opera and Chrome

Before you start studying jQuery, you should have a basic knowledge of:

* HTML
* CSS
* JavaScript

jQuery is licensed under the MIT Licence. It's syntax is designed to make it easier to navigate a document, select **DOM** elements, create **animations** , handle **events** , and **develop Ajax** applications . jQuery also provides capabilities for developers to create plug-ins on top of the JavaScript library.

The library contains the following features:

* HTML element selections
* HTML element manipulation
* CSS manipulation
* HTML event functions
* JavaScript Effects and animations
* HTML DOM traversal and modification
* AJAX
* Utilities

**jQuery Syntax**

The jQuery syntax is tailor made for selecting HTML elements and

perform some action on the element(s).

Basic syntax is: $(selector).action()

**Adding the jQuery Library to Your Pages**

The jQuery library is stored as a single JavaScript file, containing all the jQuery methods.

It can be added to a web page with the following mark-up:

< head>  
 < script type="text/javascript" src="jquery.js"></script>  
< /head>

**Snippets**

<html>

<head>

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

<script type="text/javascript">

$(document).ready(function(){

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

$(this).hide();

});

});

</script>

</head>

<body>

<p>If you click on me, I will disappear.</p>

</body>

</html>

**document.ready() :**

This is to prevent any jQuery code from running before the document is finished

loading (is ready).

Here are some examples of actions that **can fail** if functions are run before the

document is fully loaded:

•Trying to hide an element that doesn't exist

•Trying to get the size of an image that is not loaded

**Alternatives to Downloading**

If you don't want to store the jQuery library on your own computer, you can use the hosted jQuery library from **Google** or **Microsoft**.

This is a good option for **future proofing** the library and **easing maintenance**. Once they get updated by Google or Microsoft, it will **update automatically** your page. Also, you can take advantage of new **features and functionalities that they innovate**. They act as **CDN** (Content Delivery Network).

**Google**

<head>  
 <script type="text/javascript" src="http://ajax.googleapis.com/ajax/libs/jquery/1.4.2/jquery.min.js"></script>  
</head>

**Microsoft**

<head>  
 <script type="text/javascript" src="http://ajax.microsoft.com/ajax/jquery/jquery-1.4.2.min.js"></script>  
</head>

**jQuery Element Selectors**

jQuery uses **CSS selectors** to select HTML elements. Note the selectors in this case are strings.

$("p") **selects all <p> elements.**

$("p.intro") **selects all <p> elements with class="intro".**

$("p#demo") **selects <p> element with id="demo".**

**jQuery Attribute Selectors**

jQuery uses **XPath** expressions to select elements with given attributes. Note the prevalence of the square brackets inside the string quotes. The selectors are also strings in this case. Also note if the attributes are assigned values then they are quoted in single quotes. Escape properly.

$("[href]") select all elements with an href attribute.

$("[href='#']") **select all elements with an href value equal to "#".**

$("[href!='#']") **select all elements with an href attribute NOT equal to "#".**

$("[href$='.jpg']") **select all elements with an href attribute that ends with ".jpg".**

**jQuery CSS Selectors**

jQuery CSS selectors can be used to change CSS properties for HTML elements.

The following example changes the background-color of all p elements to yellow:

Example

<html>

<head>

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

<script type="text/javascript">

$(document).ready(function(){

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

$("p").css("background-color","yellow");

});

});

</script>

</head>

<body>

<h2>This is a heading</h2>

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

<p>This is another paragraph.</p>

<button>Click me</button>

</body>

</html>

**Syntax Description**

$(this) - **Maps on to the current HTML element**

$("p#intro:first") **The first <p> element with id="intro"**

$(".intro") - **All elements with class="intro"**

$("ul li:first") - **The first <li> element of the first <ul**>

$("ul li:first-child") - **The first <li> element of every <ul>**

$("[href$='.jpg']") - **All elements with an href attribute that ends with ".jpg"**

$("div#intro .head") **All elements with class="head" inside a <div> element with id="intro"**

**Basic jQuery Example**

The following example demonstrates the jQuery hide() method, hiding all <p> elements in an HTML document.

Example 1

< html>  
< head>  
< script type="text/javascript" src="jquery.js"></script>  
< script type="text/javascript">  
 $(document).ready(function(){  
 $("button").click(function(){  
 $("p").hide();  
 });  
 });  
< /script>  
< /head>  
  
<body>  
 <p>This is a paragraph.</p>  
 <p>This is another paragraph.</p>  
 <button>Click me</button>  
</body>  
</html>

**jQuery Syntax**

The jQuery syntax is tailor made for **selecting** HTML elements and performing some **action** on the element(s).

Basic syntax is: **$(selector).action()**

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

Examples:

$(this).hide() - hides current element

$("p").hide() - hides all paragraphs

$("p.test").hide() - hides all paragraphs with class="test"

$("#test").hide() - **hides the element with id="test"**

[**$(this).hide()**](http://www.w3schools.com/jquery/tryit.asp?filename=tryjquery_hide_this)  
Demonstrates the jQuery hide() method, hiding the **current** HTML element.

Example 2

<html>

<head>

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

<script type="text/javascript">

$(document).ready(function(){

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

$(this).hide();

});

});

</script>

</head>

<body>

<button>Click me</button>

</body>

</html>

[**$("#test").hide()**](http://www.w3schools.com/jquery/tryit.asp?filename=tryjquery_hide_id)  
Demonstrates the jQuery hide() method, hiding the element with **id="test".**

Example 3

<html>

<head>

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

<script type="text/javascript">

$(document).ready(function(){

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

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

});

});

</script>

</head>

<body>

<h2>This is a heading</h2>

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

**<p id="test">This is another paragraph.</p>**

<button>Click me</button>

</body>

</html>

[**$(".test").hide()**](http://www.w3schools.com/jquery/tryit.asp?filename=tryjquery_hide_class)  
Demonstrates the jQuery hide() method, hiding all elements with **class="test"**

<html>

<head>

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

<script type="text/javascript">

$(document).ready(function(){

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

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

});

});

</script>

</head>

<body>

**<h2 class="test">This is a heading</h2>**

**<p class="test">This is a paragraph.</p>**

<p>This is another paragraph.</p>

<button>Click me</button>

</body>

</html>

## **Summary**

|  |  |
| --- | --- |
| **Syntax** | **Description** |
| $(this) | Current HTML element |
| $("p") | All <p> elements |
| $("p.intro") | All <p> elements with class="intro" |
| $("p#intro") | All <p> elements with id="intro" |
| $("p#intro:first") | The first <p> element with id="intro" |
| $(".intro") | All elements with class="intro" |
| $("#intro") | The first element with id="intro" |
| $("ul li:first") | The first <li> element of the first <ul> |
| $("ul li:first-child") | The first <li> element of every <ul> |
| $("[href$='.jpg']") | All elements with an href attribute that ends with ".jpg" |
| $("div#intro .head") | All elements with class="head" inside a <div> element with id="intro" |

## **jQuery Event Functions**

The jQuery event handling methods are core functions in jQuery.

Event handlers are method that are called when "something happens" in HTML. The term "**triggered (or "fired") by an event**" is often used.

It is common to put jQuery code into event handler methods in the <head> section:

< html>  
< head>  
 < script type="text/javascript" src="jquery.js"></script>  
 < script type="text/javascript">  
 $(document).ready(function(){  
 $("button").click(function(){  
 $("p").hide();  
 });  
 });  
< /script>  
< /head>  
  
< body>  
 < h2>This is a heading</h2>  
 < p>This is a paragraph.</p>  
 < p>This is another paragraph.</p>  
 < button>Click me</button>  
< /body>

</html>

**jQuery Hide and Show**

With jQuery, you can hide and show HTML elements with the hide() and show() methods:

<html>

<head>

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

<script type="text/javascript">

$(document).ready(function(){

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

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

});

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

$("p").show();

});

});

</script>

</head>

<body>

<p>If you click on the "Hide" button, I will disappear.</p>

<button id="hide">Hide</button>

<button id="show">Show</button>

</body>

</html>

**jQuery Toggle**

The jQuery toggle() method toggles the visibility of HTML elements using the show() or hide() methods.

Shown elements are hidden and hidden elements are shown.

Syntax:

$(selector).toggle(speed,callback)

The speed parameter can take the following values as string: "slow", "fast", "normal", or milliseconds.

<html>

<head>

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

<script type="text/javascript">

$(document).ready(function(){

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

$("p").toggle();

});

});

</script>

</head>

<body>

<button>Toggle</button>

<p>This is a paragraph with little content.</p>

<p>This is another small paragraph.</p>

</body>

</html>

The jQuery slide methods gradually change the height for selected elements.

jQuery has the following slide methods:

$(selector).slideDown(speed,callback)

$(selector).slideUp

(speed,callback)

$(selector).slideToggle(speed,callback)

The speed parameter can take the following values: "slow", "fast", "normal", or milliseconds.

The callback parameter is the name of a function to be executed after the function completes.

**slide toggle**

<html>

<head>

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

<script type="text/javascript">

$(document).ready(function(){

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

$(".panel").slideToggle("slow");//alternatively use slideUp/slideDown

});

});

</script>

<style type="text/css">

div.panel,p.flip {

margin:0px;

padding:5px;

text-align:center;

background:#e5eecc;

border:solid 1px #c3c3c3;

}

div.panel

{

height:120px;

display:none;

}

</style>

</head>

<body>

<div class="panel">

<p>Because time is valuable, we deliver quick and easy learning.</p>

<p>At The City Lit, you can study everything you need to learn, in an accessible and handy format.</p>

</div>

<p class="flip">Show/Hide Panel</p>

</body>

</html>

## **Functions in a Separate File**

If your website contains a lot of pages, and you want your jQuery functions to be easy to maintain, put your jQuery functions in a **separate** .js file.

When we demonstrate jQuery here, the functions are added directly into the <head> section, However, sometimes it is preferable to place them in a separate file, like this (refer to the file with the src attribute):

## Example 8

< head>  
< script type="text/javascript" src="jquery.js"></script>  
< script type="text/javascript" src="my\_jquery\_functions.js"></script>  
< /head>

## **jQuery Name Conflicts**

jQuery uses the $ sign as a shortcut for jQuery. It serves as a wrapper.

Some other JavaScript libraries also use the dollar sign for their functions.

The jQuery **noConflict()** method specifies a custom name (like jq), instead of using the dollar sign.

## Example 9

<html>

<head>

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

<script type="text/javascript">

var jq=jQuery.noConflict();

jq(document).ready(function(){

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

jq("p").hide();

});

});

</script>

</head>

<body>

<h2>This is a heading</h2>

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

<p>This is another paragraph.</p>

<button>Click me</button>

</body>

</html>

Here are some examples of event methods in jQuery:

|  |  |
| --- | --- |
| **Event Method** | **Description** |
| $(document).ready(function) | Binds a function to the ready event of a document (when the document is finished loading) |
| $(*selector*).click(function) | Triggers, or binds a function to the click event of selected elements |
| $(*selector*).dblclick(function) | Triggers, or binds a function to the double click event of selected elements |
| $(*selector*).focus(function) | Triggers, or binds a function to the focus event of selected elements |
| $(*selector*).mouseover(function) | Triggers, or binds a function to the mouseover event of selected elements |

**jQuery Effects**

Hide, Show, Toggle, Slide, Fade, and Animate.

Both hide() and show() can take the two optional parameters: speed and callback.

Syntax:

$(selector).hide(speed,callback)

$(selector).show(speed,callback)

The speed parameter specifies the speed of the hiding/showing, and can take the following values: "slow", "fast", "normal", or milliseconds:

The callback parameter is the name of a function to be executed after the hide (or show) method completes.

## jQuery Slide - slideDown, slideUp, slideToggle

The jQuery slide methods gradually change the **height** for selected elements.

jQuery has the following slide methods:

**$(selector).slideDown(speed,callback)**

**$(selector).slideUp(speed,callback)**

**$(selector).slideToggle(speed,callback)**

The speed parameter can take the following values: "slow", "fast", "normal", or milliseconds.

The callback parameter is the name of a function to be executed after the function completes

## **jQuery Fade - fadeIn, fadeOut, fadeTo**

The jQuery fade methods gradually changes the **opacity** for selected elements.

jQuery has the following fade methods:

$(selector).fadeIn(speed,callback)

$(selector).fadeOut(speed,callback)

$(selector).fadeTo(speed,opacity,callback)

The speed parameter can take the following values: "slow", "fast", "normal", or milliseconds.

The opacity parameter in the fadeTo() method allows fading to a given opacity.

The callback parameter is the name of a function to be executed after the function completes.

## **jQuery Custom Animations**

The syntax of jQuery's method for making custom animations is:

**$(selector).animate({params},[duration],[easing],[callback])**

The key parameter is **params**. It defines the CSS properties that will be animated. Many properties can be animated at the same time:

animate({width:"70%",opacity:0.4,marginLeft:"0.6in",fontSize:"3em"});

The second parameter is **duration**. It specifies the speed of the animation. Possible values are "fast", "slow", "normal", or milliseconds.

Note:

HTML elements are positioned **static** by default and cannot be moved.  
To make elements **moveable**, set the CSS **position** property to **fixed**, **relative** or **absolute**.

**jQuery** **.addClass( className )**

**function (index, currentClass)** A function returning one or more space-separated class names to be added to the existing class name(s). Receives the index position of the element in the set, and the existing class name(s) as arguments. Within the function, this refers to the current element in the set.

It's important to note that this method does not replace a class. It simply adds the class, appending it to any which may already be assigned to the elements.

More than one class may be added at a time, separated by a space, to the set of matched elements, like so:

$("p").addClass("myClass yourClass");

This method is often used with .removeClass() to switch elements' classes from one to another, like so:

$("p").removeClass("myClass noClass").addClass("yourClass");

Here, the myClass and noClass classes are removed from all paragraphs, while yourClass is added.

**Adds the class "selected" to the matched elements.**

<!DOCTYPE html>  
<html>  
<head>  
 <style>  
 p { margin: 8px; font-size:16px; }  
 .selected { color:blue; }  
 .highlight { background:yellow; }  
 </style>  
 <script src="http://code.jquery.com/jquery-latest.js"></script>  
</head>  
<body>  
   
 <p>Hello</p>  
 <p>and</p>  
 <p>Goodbye</p>  
   
<script>  
 $("p:last").addClass("selected");  
 </script>  
  
</body>  
</html>

**Pass in a function to .addClass() to add the "green" class to a div that already has a "red" class.**

<!DOCTYPE html>  
<html>  
<head>  
 <style>  
 div { background: white; }  
 .red { background: red; }  
 .red.green { background: green; }  
 </style>  
 <script src="http://code.jquery.com/jquery-latest.js"></script>  
</head>  
<body>  
   
 <div>This div should be white</div>  
 <div class="red">This div will be green because it now has the "green" and "red" classes.  
 It would be red if the addClass function failed.</div>  
 <div>This div should be white</div>  
 <p>There are zero green divs</p>  
  
<script>  
 $("div").addClass(function(index, currentClass) {  
 var addedClass;  
  
 if ( currentClass === "red" ) {  
 addedClass = "green";  
 $("p").text("There is one green div");  
 }  
   
 return addedClass;  
 });  
</script>  
  
</body>  
</html>

**jQuery Callback Functions**

A callback function is executed after the current animation is 100% finished.

JavaScript statements are executed line by line. However, with animations, the next line of code can be run even though the animation is not finished. This can create errors.

To prevent this, you can create a callback function.

A callback function is executed after the current animation (effect) is finished.

Typical syntax: **$(selector).hide(speed,callback)**

The callback parameter is a function to be executed after the hide effect is completed:

**jQuery html**

jQuery contains powerful methods (functions) for changing and manipulating HTML elements and attributes.

## Changing HTML Content

### $(selector).html(content)

The html() method changes the contents (innerHTML) of matching HTML elements.

## **Adding HTML content**

$(selector).append(content)

The append() method appends content to the **inside** of matching HTML elements.

$(selector).prepend(content)

The prepend() method "prepends" content to the inside of matching HTML elements.

**$(selector).after(content)**

The after() method inserts HTML content after all matching elements.

$(selector).before(content)

The before() method inserts HTML content before all matching elements.

**jQuery css() Method**

jQuery has one important method for CSS manipulation: css()

The css() method has three different syntaxes, to perform different tasks.

* css(name) - Return CSS property value
* css(name,value) - Set CSS property and value
* css({properties}) - Set multiple CSS properties and values as an object.

## **Set CSS Property and Value**

Use css(name,value) to set the specified CSS property for ALL matched elements:

$("p").css("background-color","yellow");

## **Set Multiple CSS Property/Value Pairs**

Use css({properties}) to set one or more CSS property/value pairs for the selected elements. Takes an object form:

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

## **jQuery height() and width() Methods**

jQuery has two important methods for size manipulation.

* height()
* width()

The height() method sets the height of all matching elements:

$("#div1").height("200px");

The width() method sets the width of all matching elements:

$("#div2").width("300px");

**Building jQuery Applications**

Slides

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">

<html>

<head>

<title>jQuery slideshow</title>

</head>

<link type="text/css" rel="stylesheet" href="slideshow.css" media="print" />

<script language="javascript" src="slideshow.js" type="text/javascript"></script>

<body>

<div id="slideshow">

<div id="slidesContainer">

<div class="slide">

<!-- Content for slide 1 goes here -->

</div>

<div class="slide">

<!-- Content for slide 2 goes here. -->

</div>

<div class="slide">

<!-- Content for slide 3 goes here. -->

</div>

<div class="slide">

<!-- Content for slide 4 goes here. -->

</div>

</div>

</div>

</body>

</html>

Notes:

In this example, you’ll see how **text-based** browsers, and browsers **incapable** of **rendering CSS and JavaScript,** will **see our slideshow**. It’s important to note that all of our content is **easily accessible**; we’ve hidden nothing from the user – ensuring that everyone will have the ability to view our content. The widget will degrade gracefully.

There is also no markup for the left and right arrow controls, which we will insert into the DOM later on using **JavaScript**. Having them in the content structure level would be **confusing** to individuals without CSS or JavaScript capabilities.

For #**slidesContainer**, we set the **overflow** property to **auto** so that scroll bars appear when our content overflows over the set height of 263px (the height of our slides). This is a requirement

We have to **reduce** the width of the .slide div class by **20px** to accommodate **the right hand scroll bar that will appear when JavaScript is turned off**.

**Without JavaScript**, our content is still accessible; users can **scroll up and down** to view our slides. Alternatively, you can give the **.slide style rule a float:left**; property so that instead of scrolling **verticall**y, users can scroll **horizontally** when JavaScript is turned off.

**Left and right arrow CSS** : To save some JavaScript rendering resources, we’ll declare style rules for the left and right arrow controls that we will insert in the DOM via **jQuery**. The elements will be span elements, so we declare a cursor property and assign it the value of pointer to change the mouse pointer when the user hovers over the controls. We use the **text-indent** property to hide **the text out of sight, a CSS background image replacement method**.

Next, **undo the styles** that we declared in CSS that deals with the **JavaScript off** scenario. This involves declaring CSS styles in JavaScript for #slidesContainer to remove its scroll bar. Additionally, we have to **resize** our .slide divs to **560px**, which we reduced by **20px** in to accommodate the scroll bar. We also want to **float the divs to the left so that they are displayed side by side horizontally instead of stacked on top of each other vertically**.

Then, by DOM manipulation, we insert a div called #**slideInner** (invinsible in the HTML code) that **wraps around all of our slides** that has a width equal to the total width of all the .slide div.

Now, we insert **left** and **right** controls (with class names of .control) for **user navigation**; we do this in JavaScript so that browser clients that don’t have JavaScript enabled won’t see the controls. Now we insert the JavaScript in the head.

We **insert the controls** by **manipulating the DOM**; this way, users using **JavaScript-disabled browsers and screen readers won’t have an invalid HTML structure with controls that take them nowhere which would be confusing because clicking on them would not work without JavaScript.**

We do this using the **.prepend()** and **.append()** method which inserts an HTML string inside the selected object/s (in this case, the #slideshow div is selected). The text inside the span elements don’t matter because they were hidden

To **manage** visibility of our controls, we create a function called **manageControls** **that hides and shows the left and right arrow controls based on the current position of the slideshow**.

If it’s on the **first slide**, we hide the **left control** because there is **no preceding slide**. On the **last slide**, **we hide the right control because the user has reached the end of the slideshow**. We do this by using the **.hide()** and **.show()** jQuery methods that hides/shows the selected DOM element/s that precedes it.

Managing the **current position** using, manageControls, and positions as argument: Mind the use of ternary operators for the bind operation could be either +1 to the right when the control is clicked or -1 for the same operation but to the left.

Finally , we **move #slideInner to the left or right by animating it’s** **margin-left CSS property** value. The left margin is the negative of the width of our slides multiplied by our current position. For example, if we’re moving to Slide 3, then our left margin is equal to -1120px [-([580-20]q\*2)].

**Form Validation with AJAX response**

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">

<html>

<head>

<title>jQuery Form submission</title>

<!-- Begin Adding jQuery and jQuery Validation -->

<script type="text/javascript" src="js/jquery.min.js"></script>

<script type="text/javascript" src="js/ajax-form.js"></script>

<link href="css/tutorial.css" media="all" type="text/css" rel="stylesheet">

</head>

<body>

<!-- Process our Form Submission with jQuery’s AJAX Function without page refresh

http://demos.net.tutsplus.com/contactform/-->

<div id="contact\_form">

<!-- Build the HTML Form -->

<form name="contact" action="">

<fieldset>

<label for="name" id="name\_label">Name</label>

<input type="text" name="name" id="name" size="30" value="" class="text-input" />

<label class="error" for="name" id="name\_error">This field is required.</label>

<label for="email" id="email\_label">Email</label>

<input type="text" name="email" id="email" size="30" value="" class="text-input" />

<label class="error" for="email" id="email\_error">This field is required.</label>

<label for="phone" id="phone\_label">Phone</label>

<input type="text" name="phone" id="phone" size="30" value="" class="text-input" />

<label class="error" for="phone" id="phone\_error">This field is required.</label>

<br />

<input type="submit" name="submit" class="button" id="submit\_btn" value="Send" />

</fieldset>

</form>

</div>

</body>

</html>