jQuery

Contents

[1 Introduction 2](#_Toc156996312)

[2 How to use jQuery 3](#_Toc156996313)

[3 .ready(handler) Method 3](#_Toc156996314)

[4 Example – HelloWorld and Hide/Show 4](#_Toc156996315)

[5 Example – TextField & TextArea 5](#_Toc156996316)

[6 Example – Checkboxes 6](#_Toc156996317)

[7 Example – RadioButtons 6](#_Toc156996318)

[8 Example – Dropdown 6](#_Toc156996319)

[9 Example – Dropdown Multi-Select 7](#_Toc156996320)

[10 Example – Chaining 7](#_Toc156996321)

[11 Example – Form Processing 8](#_Toc156996322)

[12 More Examples 10](#_Toc156996323)

[13 Expectations 10](#_Toc156996324)

[Appendix 1 W3Schools Tutorials 11](#_Toc156996325)

[Appendix 2 Examples of Selectors 12](#_Toc156996326)

[Appendix 3 Events 12](#_Toc156996327)

[Appendix 4 Effects 12](#_Toc156996328)

[Appendix 5 Examples 12](#_Toc156996329)

[Appendix 6 Callback Functions 13](#_Toc156996330)

[Appendix 7 Get and Set Contents 13](#_Toc156996331)

[Appendix 8 Add/Remove Contents 13](#_Toc156996332)

[Appendix 9 Manipulating CSS 13](#_Toc156996333)

[Appendix 10 References 13](#_Toc156996334)

[Appendix 11 External Examples 13](#_Toc156996335)

# Introduction

jQuery is a library that is written on top of the JavaScript DOM and makes it easier to manipulate HTML elements. This allows developers to write simpler code to provide an enhanced user experience. Anything you can do in jQuery, you can do in JavaScript, but in many cases, it would take much more code.

As discussed previously, the DOM is an API to access and manipulate HTML elements. Consider several different ways to select elements using JavaScript and jQuery.

|  |  |  |
| --- | --- | --- |
| **Select Element(s) by** | **JavaScript** | **jQuery** |
| Id | *getElementById("myId")* | *$("#myId")* |
| Name attribute | *getElementsByName("myName")* | *$("input[name=’myName’]")* |
| Class name | *getElementsByClassName("myClass")* | *$(".myClass")* |
| Tag | *getElementsByTagName("p")* | *$(“p”)* |

However, jQuery allows many more ways to select elements, as we will see later. Some notes about jQuery:

* The “$” is an alias for “jQuery”. For example, these two statements are identical: $("h2") and jQuery("h2").
* *$* (or *jQuery*) is simply a JavaScript function that takes different types of input.
* The argument for a call to *$* (jQuery) is called a *selector*, which are very similar to CSS selectors.
* The JavaScript *getElement…* functions return an HTML element or an array of HTML elements. The jQuery selectors return a jQuery object that wraps the element(s) that were selected. A jQuery object has over 300 methods (actions) for manipulating the elements. This is what makes jQuery so powerful.

The basic syntax of a jQuery statement is:

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

Where:

* $ – Used to access jQuery
* *selector –* Used to find HTML element(s)
* *action –* Action to be performed on the element(s). More specifically, it is a jQuery method. As stated above, the *selector* returns HTML objects wrapped as a jQuery object; thus, we can call jQuery methods on this object.

As an example, the *text* method sets or returns the text content of selected elements. For example, this statement changes the text of all level 2 headers to the value shown:

$("h2").text("Header is changed");

Some resources:

|  |  |
| --- | --- |
| Resource | Description |
| [jQuery API](https://api.jquery.com/) | jQuery methods documentation |
| [TutorialsTeacher](https://www.tutorialsteacher.com/jquery/jquery-selectors) | Selectors tutorial |
| [freeCodeCamp](https://www.freecodecamp.org/news/the-best-jquery-examples/) | Thorough, simple jQuery tutorial |

# How to use jQuery

You can use jQuery in one of two ways:

1. Include it from a Content Delivery Network (CDN) by adding the statement below in the *head* section. Google is a [CDN](https://developers.google.com/speed/libraries#jquery) for jQuery.

<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.7.1/jquery.min.js"></script>

1. Download it from: <http://jquery.com/download/> and then put it in your folder where your pages are. Then, reference it by adding this in the *head* section:

<script src="jquery-3.7.1.min.js"></script>

# .ready(handler) Method

We first consider the:

.ready(function() {...})

jQuery method. The *ready* method accepts a *function,* which is executed when the DOM is fully loaded into memory. The *function* jQuery method calls (and/or JavaScript). One common application of this is to define and attach event handlers to various controls on the page. The general syntax for this approach is:

$(document).ready(function() {

// jQuery functions go here...

});

This statement selects the entire HTML document and calls the *ready* method on it. Once the DOM is loaded, the *function* is executed.

|  |
| --- |
| Note: The syntax above is depreciated[[1]](#footnote-1); however, you will find it all over the web and even in the official jQuery tutorials. These notes will continue to use the deprecated syntax. The new syntax is:  $(function() {  // jQuery functions go here...  }); |

For example,

|  |  |
| --- | --- |
| Code | Interpretation |
| <script>  $(document).ready(function () {  $("p").click(function () {  $(this).hide();  });  });  </script> | * Assigns the *click* event to all paragraphs, once the DOM is loaded. * The *click* event accepts a *function* which executes when a paragraph is clicked. In this case, the clicked paragraph would be hidden. * This *function* is called an *event handler* and is implemented as an *anonymous function.* * *this* is a reference to the HTML DOM element that is the source of the event (*i.e.* a paragraph). |

|  |
| --- |
| Note: The syntax above is depreciated[[2]](#footnote-2). These notes will continue to use the deprecated syntax. The new syntax is:  $("p").on(‘click’, function () {  $(this).hide();  }); |

The example above, could be modularized (though this doesn’t seem to be done very often in practice) as shown below.

function hideParagraph() {

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

$(this).hide();

});

}

$(document).ready(hideParagraph);

However, I’m not sure how you would define additional event handlers if they were needed. I would guess:

$(document).ready(function() {

hideParagraph;

doSomethingElse;

});

# Example – HelloWorld and Hide/Show

Example – Illustrates “hello world” using jQuery. Also shows how to hide and redisplay a button. See *jq\_1.html* on the website.

|  |  |
| --- | --- |
|  |  |

<html>

<head>

<title>JS Simple Example 1</title>

<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>

<script type="text/javascript">

// jQuery functions

$(document).ready(function () {

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

alert("jQuery Hello World");

});

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

if ($(this).text() === "Hide JS Button") {

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

$(this).text("Show JS Button");

}

else if ($(this).text() === "Show JS Button") {

$("#jsButton").show();

$(this).text("Hide JS Button");

}

});

});

// JavaScript function

function helloWorld() {

window.alert("Java Script Hello World");

}

</script>

</head>

<body>

<h2>JQ/JS Simple Example 1</h2>

<form id="hello">

<button type="button" id="jsButton" onclick="helloWorld()">

Java Script Hello World</button>

<button type="button" id="jqButton">jQuery Hello World</button>

<p><button type="button" id="hide">Hide JS Button</button></p>

</form>

</body>

</html>

Note:

* In JavaScript, “===” returns *true* only when the operands are the same type and the contents are the same.

# Example – TextField & TextArea

Consider this HTML which displays a button, a text box, and a text area:

<button type="button" id="process">Process</button>

<input id="name" type="text" />

<textarea id="message"></textarea>

In the jQuery code below, the element with *id=”process”* (the button) is selected and a *click* event is attached to it. As stated earlier, the [*click*](https://api.jquery.com/click-shorthand/#click-handler)method requires a function to define the event handler, which in this case, retrieves the name from the text box, builds a message, and displays it in the text area.

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

var name = $("#name").val();

var msg = "Hello " + name;

$("#message").val(msg);

});

Note:

* The function itself is essentially a JavaScript function, but can (and usually does) contain jQuery (the highlighted lines).
* The jQuery *val* method sets or returns the value of form fields. The *text* method does not work.

|  |
| --- |
| Note: The syntax for the *click* method above is apparently depreciated[[3]](#footnote-3), which these notes will continue to use. The new syntax is:  $(selector).on(“click”, function() {  // code goes here.  }); |

# Example – Checkboxes

Consider this HTML:

<p>Activities:

<input name="activities" type="checkbox" value="Soccer" />Futball

<input name="activities" type="checkbox" value="Chess" />Chess

<input name="activities" type="checkbox" value="ACM" />ACM

</p>

The [*each*](https://api.jquery.com/each/#each-function)method performs an action on each element in the jQuery object. The *each* method requires a function to define the required action. In the jQuery below, we are selecting each selected checkboxes, and pushing the *value* of each one into the (JavaScript) *activities* array.

var activities = []; // Array to store the activities that have been checked

$("input[name='activities']:checked").each(function() {

activities.push($(this).val());

});

We could also obtain the text beside the checkbox with the statement below, using the [*next*](https://api.jquery.com/next/#next-selector)method. This works even if a Label is used for the checkboxes.

var chkText = $(this).next().text();

If the Label is on the left of the checkbox, which is more standard, I believe, you could use *.prev,* instead of .*next,* in order to access the previous node. However, if only text was used on the left, I’m not sure how to obtain that text. I tried, *.prev* which gives the previous sibling; however, it didn’t work.

# Example – RadioButtons

Radiobuttons are handled similarly to checkboxes. Consider:

<p>Mood:

<input name="mood" type="radio" value="Happy" />Happy

<input name="mood" type="radio" value="Copacetic" />Copacetic

<input name="mood" type="radio" value="Sad" />Sad

</p>

Since only one radiobutton can be selected, we don’t need *.each* as with checkboxes:

var mood = $("input[name='mood']:checked").val();

# Example – Dropdown

The HTML:

<select id="classification" class="inVals" name="D1">

<option value="1">Freshman</option>

<option value="2">Sophomore</option>

<option value="3">Junior</option>

<option value="4">Senior</option>

</select>

The jQuery:

var classificationDesc = $("#classification option:selected").text();

var classificationNum = $("#classification option:selected").val();

# Example – Dropdown Multi-Select

The HTML:

<select size="5" id="classes" multiple="multiple">

<option value="3">CS 3300</option>

<option value="2">CS 3340</option>

<option value="4">CS 3410</option>

<option value="4">CS 3520</option>

<option value="3">CS 4721</option>

</select>

The jQuery:

// Get classes from multi-select dropdown

var classes = []; // Array to store class names

var credits = []; // Array to store class credits

var totalCredits = 0; // Stores total credits

$("#classes option:selected").each(function () {

classes.push($(this).text());

credits.push($(this).val());

totalCredits += parseInt($(this).val());

});

# Example – Chaining

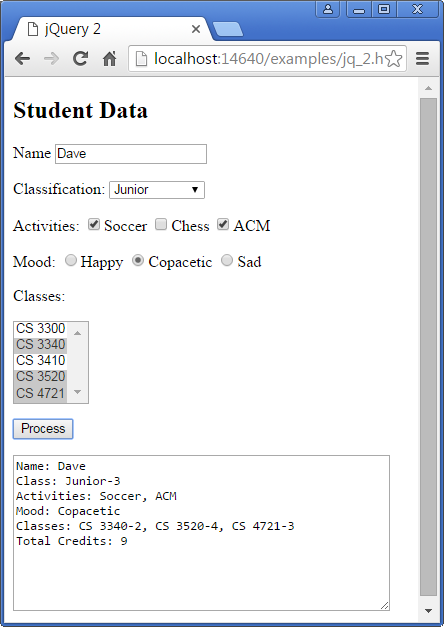
Many jQuery methods return jQuery objects which allows function calls to be chained. For example:

$("#p1").css("color", "red").slideUp(2000).slideDown(2000);

Source: [https://www.w3schools.com/jquery/jq uery\_chaining.asp](https://www.w3schools.com/jquery/jq%20%20uery_chaining.asp)

# Example – Form Processing

Illustrates processing form elements. See *jq\_2.html* on website. Consider the page below.



The code on the next page shows the jQuery to process this form. Interspersed with the jQuery is the HTML it is referencing. Of course, the HTML would be in the body; it is interspersed with the jQuery for ease of understanding the jQuery.

$(document).ready(function () {

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

HTML - TextBox

<p>Name <input id="name" class="inVals" type="text" /></p>

jQuery

var name = $("#name").val();

HTML – Dropdown, Single Selection

<select id="classification" class="inVals" name="D1">

<option value="1">Freshman</option>

<option value="2">Sophomore</option>

<option value="3">Junior</option>

<option value="4">Senior</option>

</select>

jQuery

var classificationDesc = $("#classification option:selected").text();

var classificationNum = $("#classification option:selected").val();

HTML - Checkboxes

<p>Activities:

<input name="activities" class="acts" type="checkbox" value="Soccer" />Soccer

<input name="activities" class="acts" type="checkbox" value="Chess" />Chess

<input name="activities" class="acts" type="checkbox" value="ACM" />ACM

</p>

jQuery

var activities = [];

$("input[name='activities']:checked").each(function() {

activities.push($(this).val());

});

HTML - RadioButtons

<p>Mood:

<input name="mood" type="radio" value="Happy" />Happy

<input name="mood" type="radio" value="Copacetic" />Copacetic

<input name="mood" type="radio" value="Sad" />Sad

</p>

jQuery

var mood = $("input[name='mood']:checked").val();

HTML – Dropdown, Multiple Selection

<select size="5" id="classes" multiple="multiple">

<option value="3">CS 3300</option>

<option value="2">CS 3340</option>

<option value="4">CS 3410</option>

<option value="4">CS 3520</option>

<option value="3">CS 4721</option>

</select>

jQuery

var classes = [];

var credits = [];

var totalCredits = 0;

$("#classes option:selected").each(function () {

classes.push($(this).text());

credits.push($(this).val());

totalCredits += parseInt($(this).val());

});

HTML – TextArea

<p><textarea rows="10" id="message" cols="50"></textarea></p>

jQuery

var message = "Name: " + name + "\n";

message += "Class: " + classificationDesc + "-" + classificationNum + "\n";

message += "Activities: ";

for (i = 0; i < activities.length; i++)

message += activities[i] + (i<activities.length-1 ? ", " : "");

message += "\nMood: " + mood + "\n";

message += "Classes: ";

for (i = 0; i < classes.length; i++)

message += classes[i] + "-" + credits[i] + (i < classes.length - 1 ? ", " : "");

message += "\nTotal Credits: " + totalCredits;

$("#message").val(message);

});

});

# More Examples

* See *jq\_ex1.html* to illustrate double-click, css, fade
* See *jq\_styles.html* to illustrate add css class, and toggling classes
* See *jq\_ex2.html* to illustrate loading a file from server.

# Expectations

1. Given a description of a simple client-side dynamic behavior write the HTML and jQuery to implement:

* Use a jQuery selector to select a form element based on id
* Apply jQuery *click* event
* Use jQuery to process form elements for a simple example.

Appendix

1. W3Schools Tutorials

For each tutorial, I have listed the chapters and highlighted in blue are the ones you should read.

|  |
| --- |
| [jQuery Tutorial](http://www.w3schools.com/jquery/default.asp)   1. jQuery HOME – Read carefully 2. jQuery Intro – Read carefully 3. jQuery Getting Started – Read carefully 4. jQuery Syntax – Read carefully 5. jQuery Selectors – Read carefully 6. jQuery Events – Read carefully though the ready and click events, scan the rest of them if interested.   [jQuery Effects Tutorial](http://www.w3schools.com/jquery/jquery_hide_show.asp)   1. jQuery Hide/Show – Read carefully 2. jQuery Fade – Omit 3. jQuery Slide – Omit 4. jQuery Animate – Omit 5. jQuery stop() – Omit 6. jQuery Callback – Read carefully 7. jQuery Chaining – Scan   [jQuery HTML Tutorial](http://www.w3schools.com/jquery/jquery_dom_get.asp)   1. jQuery Get – Read carefully 2. jQuery Set – Read carefully 3. jQuery Add – Omit 4. jQuery Remove – Omit 5. jQuery CSS Classes – Omit 6. jQuery css() – Read carefully 7. jQuery Dimensions – Omit   [jQuery AJAX Tutorial](http://www.w3schools.com/jquery/jquery_ajax_intro.asp)   1. jQuery AJAX Intro – Read carefully 2. jQuery Load – Read carefully 3. jQuery Get/Post – Omit |

1. Examples of Selectors

jQuery allows a much more flexible way to select elements. For example[[4]](#footnote-4)

|  |  |
| --- | --- |
| $(“\*”) | Selects all elements |
| $(this) | Selects the current HTML element |
| $("p.intro") | Selects all <p> elements with class="intro" |
| $("p:first") | Selects the first <p> element |
| $("ul li:first") | Selects the first <li> element of the first <ul> |
| $("ul li:first-child") | Selects the first <li> element of every <ul> |
| $("[href]") | Selects all elements with an href attribute |
| $("a[target='\_blank']") | Selects all <a> elements with a target attribute value equal to "\_blank" |
| $("a[target!='\_blank']") | Selects all <a> elements with a target attribute value NOT equal to "\_blank" |
| $(":button") | Selects all <button> elements and <input> elements of type="button" |
| $("tr:even") | Selects all even <tr> elements |
| $("tr:odd") | Selects all odd <tr> elements |

A good reference for selecting elements with jQuery is [learn.jQuery](https://learn.jquery.com/using-jquery-core/selecting-elements/).

1. Events

When the user interacts with a page, we want to *change* things in response. User interactions *fire events* which jQuery can respond to.

* <http://www.w3schools.com/jquery/jquery_events.asp>
* Full list of jQuery events: <http://www.w3schools.com/jquery/jquery_ref_events.asp>

1. Effects

How do we change the HTML?

* <http://www.w3schools.com/jquery/jquery_hide_show.asp>
* Full list of jQuery effects: <http://www.w3schools.com/jquery/jquery_ref_effects.asp>

1. Examples

|  |  |
| --- | --- |
| **Effect** | **Description** |
| Fading | <http://www.w3schools.com/jquery/jquery_fade.asp> |
| Sliding | http://www.w3schools.com/jquery/jquery\_slide.asp |
| animate | <http://www.w3schools.com/jquery/jquery_animate.asp> |
| Stop | http://www.w3schools.com/jquery/jquery\_stop.asp |

1. Callback Functions

Some functions take an optional argument which is a callback function. A callback function is executed after an effect is finished.

http://www.w3schools.com/jquery/jquery\_callback.asp

1. Get and Set Contents

How do we get or set the contents of elements? These methods are useful:

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

Examples:

* get: <http://www.w3schools.com/jquery/jquery_dom_get.asp>
* set: <http://www.w3schools.com/jquery/jquery_dom_set.asp>

1. Add/Remove Contents

Add/Remove the contents of a tag or tags themselves:

* <http://www.w3schools.com/jquery/jquery_dom_add.asp>
* <http://www.w3schools.com/jquery/jquery_dom_remove.asp>

1. Manipulating CSS

Add/Remove existing CSS classes or add CSS on the fly:

* <http://www.w3schools.com/jquery/jquery_css_classes.asp>
* <http://www.w3schools.com/jquery/jquery_css.asp>
* jQuery HTML/CSS Reference: <http://www.w3schools.com/jquery/jquery_ref_html.asp>

1. References
2. jQuery.com: [How jQuery Works](http://learn.jquery.com/about-jquery/how-jquery-works/), [jQuery Event Basics](http://learn.jquery.com/events/event-basics/) and [Introduction to Effects](http://learn.jquery.com/effects/intro-to-effects/).
3. jQuery Fundamentals: [jQuery Basics](http://jqfundamentals.com/chapter/jquery-basics), [Events & Event Delegation](http://jqfundamentals.com/chapter/events).
4. External Examples
5. [jQuery Examples](https://www.tutorialspoint.com/jqueryexamples/index.htm) – Tons of examples
6. [12 Examples](http://net.tutsplus.com/tutorials/javascript-ajax/20-helpful-jquery-methods-you-should-be-using/) - Examples of 12 jQuery functions
7. [jQuery.com/Learning Center](http://learn.jquery.com/) - Nicely organized Chapters.

1. <https://api.jquery.com/ready/> [↑](#footnote-ref-1)
2. <https://api.jquery.com/click-shorthand/> [↑](#footnote-ref-2)
3. <https://api.jquery.com/click-shorthand/#click-handler> [↑](#footnote-ref-3)
4. <http://www.w3schools.com/jquery/jquery_selectors.asp> [↑](#footnote-ref-4)