**Introduction**

jQuery is a javascript library. There are many javascript libraries, such as Moo Tools, Scriptaculous, Dojo, Prototype, and many more. The reason to learn about jQuery over other great libraries is due to its simplicity and flexibility.

jQuery allows us to animate elements, add/remove classes, attributes, id’s, etc. It also allows us to use Ajax in a very simple way. It is the most popular javascript library for a reason. The best part is that when you execute some jQuery, it will run the same way in all five major browsers.

Let us learn in brief about the options available on the website jQuery.com.

### Installation

There are several ways to start using jQuery on your web site. You can:

* Download the jQuery library from jQuery.com
* Include jQuery from a CDN, like Google

##### Downloading jQuery

There are two versions of jQuery available for downloading:

* Production version – this is for your live website because it has been minified and compressed
* Development version – this is for testing and development (uncompressed and readable code)

Both versions can be downloaded from [jQuery.com](http://jquery.com/download/).

The jQuery library is a single JavaScript file, and you reference it with the HTML <script> tag (notice that the <script> tag should be inside the <head> section):

<head>

<script src="jquery-3.5.1.min.js"></script>

</head>

##### jQuery CDN

If you don’t want to download and host jQuery yourself, you can include it from a CDN (Content Delivery Network).

Google is an example of someone who host jQuery:

<head>

<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>

</head>

### Syntax

With jQuery you select (query) HTML elements and perform “actions” on them.

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

Basic syntax is: **$(*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)

Examples:

$(this).hide() – hides the current element.

$("p").hide() – hides all <p> elements.

$(".test").hide() – hides all elements with class=”test”.

$("#test").hide() – hides the element with id=”test”.

**Note:** jQuery uses CSS syntax to select elements. You will learn more about the selector syntax in the next section.

##### The Document Ready Event

You might have noticed that all jQuery methods in our examples, are inside a document ready event:

$(document).ready(function(){

*// jQuery methods go here...*

});

This is to prevent any jQuery code from running before the document is finished loading (is ready).

It is good practice to wait for the document to be fully loaded and ready before working with it. This also allows you to have your JavaScript code before the body of your document, in the head section.

Here are some examples of actions that can fail if methods are run before the document is fully loaded:

* Trying to hide an element that is not created yet
* Trying to get the size of an image that is not loaded yet

**Tip:** The jQuery team has also created an even shorter method for the document ready event:

$(function(){

*// jQuery methods go here...*

});

### Selectors

jQuery selectors allow you to select and manipulate HTML element(s).

jQuery selectors are used to “find” (or select) HTML elements based on their **name, id, classes, types, attributes, values** of attributes and much more. It’s based on the existing CSS Selectors, and in addition, it has some own custom selectors.

All selectors in jQuery start with the dollar sign and parentheses: $().

##### The element Selector

The jQuery element selector selects elements based on the element name.

You can select all <p> elements on a page like this:

$("p")

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

$(document).ready(function(){

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

$("p").hide();

});

});

##### The #id Selector

The jQuery #id selector uses the id attribute of an HTML tag to find the specific element.

An id should be unique within a page, so you should use the #id selector when you want to find a single, unique element.

To find an element with a specific id, write a hash character, followed by the id of the HTML element:

$("#test")

**Example:**  
When a user clicks on a button, the element with id=”test” will be hidden:

$(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")

**Example:**

When a user clicks on a button, the elements with class=”test” will be hidden:

$(document).ready(function(){

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

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

});

});

### Event Methods

Before we move on to the event methods, its important to understand **what is Event**? All the different visitors’ actions that a web page can respond to are called events.

An event represents the precise moment when something happens.

Examples:

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

The term **“fires/fired”** is often used with events. Example: “The keypress event is fired, the moment you press a key”.

Here are some common DOM events:

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

##### jQuery Syntax For Event Methods

In jQuery, most DOM events have an equivalent jQuery method. To assign a click event to all paragraphs on a page, you can do this:

$("p").click();

The next step is to define what should happen when the event fires. You must pass a function to the event:

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

// action goes here!!

// action could be an alert/calculation or anything all written in pure JavaScript or using jQuery

});

##### Commonly Used jQuery Event Methods

**$(document).ready()**

The $(document).ready() method allows us to execute a function when the document is fully loaded. As discussed earlier in the section.

**click()**

The click() method attaches an event handler function to an HTML element.

The function is executed when the user clicks on the HTML element.

The following example says: When a click event fires on a <p> element; hide the current <p> element:

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

  $(this).hide();

});

**dblclick()**

The dblclick() method attaches an event handler function to an HTML element.

The function is executed when the user double-clicks on the HTML element:

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

  $(this).hide();

});

**mouseenter()**

The mouseenter() method attaches an event handler function to an HTML element.

The function is executed when the mouse pointer enters the HTML element:

$("#p1").mouseenter(function(){

alert("You entered p1!");

});

**mouseleave()**

The mouseleave() method attaches an event handler function to an HTML element.

The function is executed when the mouse pointer leaves the HTML element:

$("#p1").mouseleave(function(){

alert("Bye! You now leave p1!");

});

**mousedown()**

The mousedown() method attaches an event handler function to an HTML element.

The function is executed, when the left, middle or right mouse button is pressed down, while the mouse is over the HTML element:

$("#p1").mousedown(function(){

alert("Mouse down over p1!");

});

**mouseup()**

The mouseup() method attaches an event handler function to an HTML element.

The function is executed, when the left, middle or right mouse button is released, while the mouse is over the HTML element:

$("#p1").mouseup(function(){

alert("Mouse up over p1!");

});

**hover()**

The hover() method takes two functions and is a combination of the mouseenter() and mouseleave() methods.

The first function is executed when the mouse enters the HTML element, and the second function is executed when the mouse leaves the HTML element:

$("#p1").hover(function(){

alert("You entered p1!");

},

function(){

alert("Bye! You now leave p1!");

});

**focus()**

The focus() method attaches an event handler function to an HTML form field.

The function is executed when the form field gets focus:

$("input").focus(function(){

$(this).css("background-color", "#cccccc");

});

**blur()**

The blur() method attaches an event handler function to an HTML form field.

The function is executed when the form field loses focus:

$("input").blur(function(){

$(this).css("background-color", "#ffffff");

});

##### The on() Method

The on() method attaches one or more event handlers for the selected elements.

Attach a click event to a <p> element:

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

$(this).hide();

});

Attach multiple event handlers to a <p> element:

$("p").on({

mouseenter: function(){

$(this).css("background-color", "lightgray");

},

mouseleave: function(){

$(this).css("background-color", "lightblue");

},

click: function(){

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

}

});

A screenshot of a computer

Description automatically generated















