What is jQuery?

jQuery is a lightweight, "write less, do more", JavaScript library.

The purpose of jQuery is to make it much easier to use JavaScript on your website.

jQuery takes a lot of common tasks that require many lines of JavaScript code to accomplish, and wraps them into methods that you can call with a single line of code.

jQuery also simplifies a lot of the complicated things from JavaScript, like AJAX calls and DOM manipulation.

The jQuery library contains the following features:

* HTML/DOM manipulation
* CSS manipulation
* HTML event methods
* Effects and animations
* AJAX
* Cross Browser Support

## How to use jQuery?

There are two ways to use jQuery.

* **Local Installation** − You can download jQuery library on your local machine and include it in your HTML code.
* **CDN Based Version** − You can include jQuery library into your HTML code directly from Content Delivery Network (CDN).

## Local Installation

* Go to the <https://jquery.com/download/> to download the latest version available.
* Now put downloaded **jquery-2.1.3.min.js** file in a directory of your website, e.g. /jquery.

## jQuery CDN

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

Both Google and Microsoft host jQuery.

To use jQuery from Google or Microsoft, use one of the following:

<head>

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

</head>

<head>

<script src="https://ajax.aspnetcdn.com/ajax/jQuery/jquery-3.2.1.min.js"></script>

</head>

## 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 $ 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".

## Selectors:

## 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:

$(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:

$(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:

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

## The 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...*  
  
});

window.onload = **function**() {

alert( "welcome" );

};

To ensure that their code runs after the browser finishes loading the document, many JavaScript programmers wrap their code in an onload function.

Unfortunately, the code doesn't run until all images are finished downloading, including banner ads. To run code as soon as the document is ready to be manipulated, jQuery has a statement known as the ready event.

The jQuery library exposes its methods and properties via two properties of the window object called jQuery and $. $ is simply an alias for jQuery and it's often employed because it's shorter and faster to write.

**jQuery and Other JavaScript Frameworks**

As you already know; jQuery uses the $ sign as a shortcut for jQuery.

There are many other popular JavaScript frameworks like: Angular, Backbone, Ember, Knockout, and more.

What if other JavaScript frameworks also use the $ sign as a shortcut?

If two different frameworks are using the same shortcut, one of them might stop working.

The jQuery team have already thought about this, and implemented the noConflict() method.

You can also create your own shortcut very easily. The noConflict() method returns a reference to jQuery, that you can save in a variable, for later use.

Here is an example:

var jq = $.noConflict();  
jq(document).ready(function(){  
    jq("div").hide();  
 });

**Getting and setting content [text(), html() and val()]**

## jQuery html() method:

The jQuery html() method gets or sets html content to the specified DOM element(s).

Syntax:

$('selector expression').html('content');

Example: jQuery html() method

$('#myDiv').html(); //returns innerHtml of #myDiv

//add <p>This is paragraph.</p> to #emptyDiv

$('#emptyDiv').html('<p>This is paragraph.</p>');

<divid="myDiv"class="yellowDiv">

<pstyle="font-size:16px;font-weight:bold">

This is paragraph.

</p>

</div>

<divid="emptyDiv">

</div>

## jQuery text() method

The jQuery text() method gets or sets the text content to the specified DOM element(s).

Syntax:

$('selector expression').text('content');

Example: jQuery text() method

$('#myDiv').text(); //returns "This is paragraph."

$('p').text(); //returns "This is paragraph."

//removes all the content from #emptyDiv and inserts "This is some text." to it

$('#emptyDiv').text('This is some text.');

<divid="myDiv"class="divCls">

<pstyle="font-size:16px;font-weight:bold">

This is paragraph.

</p>

</div>

<divid="emptyDiv">

</div>

## jQuery val() method:

The jQuery val() method gets or sets value property to the specified DOM element(s).

Syntax:

$('selector expression').val('value');

Example: jQuery val() method

$('input:Submit').val(); //returns "Save"

//set value of input text to "Steve"

$('input:text').val('Steve');

$('input:text').val(); //returns "Steve"

<div>

<label>Name:</label><inputtype="text"/>

</div>

<div>

<inputtype="Submit"value="Save"/>

</div>

**Get Value**

<divid="divTest">  
        <b>Test</b>  
        <inputtype="text"id="txtTest"name="txtTest"value="Input field"/>  
</div>  
  
<scripttype="text/javascript">  
$(function()  
{  
        alert("Text: "+ $("#divTest").text());  
        alert("HTML: "+ $("#divTest").html());  
        alert("Value: "+ $("#divTest").val());  
          
        alert("Text: "+ $("#txtTest").text());  
        alert("HTML: "+ $("#txtTest").html());  
        alert("Value: "+ $("#txtTest").val());  
});  
</script>

**Set Value**

<divid="divText"></div>  
<divid="divHtml"></div>  
<inputtype="text"id="txtTest"name="txtTest"value="Input field"/>  
  
<scripttype="text/javascript">  
$(function()  
{  
        $("#divText").text("A dynamically set text");  
        $("#divHtml").html("<b><i>A dynamically set HTML string</i></b>");  
        $("#txtTest").val("A dynamically set value");  
});  
</script>

| **Method** | **Description** |
| --- | --- |
| append() | Inserts content to the end of element(s) which is specified by a selector. |
| before() | Inserts content (new or existing DOM elements) before an element(s) which is specified by a selector. |
| after() | Inserts content (new or existing DOM elements) after an element(s) which is specified by a selector. |
| prepend() | Insert content at the beginning of an element(s) specified by a selector. |
| remove() | Removes element(s) from DOM which is specified by selector. |
| replaceAll() | Replace target element(s) with specified element. |
| wrap() | Wrap an HTML structure around each element which is specified by selector. |

## jQuery after() method:

The jQuery after() method inserts content (new or existing DOM elements) after target element(s) which is specified by a selector.

$('selector expression').after('content');

$('#div1').after('<div style="background-color:yellow"> New div </div>');

<divid="div1">div 1

</div>

<divid="div2">div 2

</div>

<divid="div1">div 1

</div>

<divstyle="background-color:yellow"> New div </div>

<divid="div2">div 2

</div>

jQuery before() method:

The jQuery before() method inserts content (new or existing DOM elements) before target element(s) which is specified by a selector.

Syntax:

$('selector expression').before('content');

Example: jQuery before() method

$('#div1').before('<div style="background-color:yellow"> New div </div>');

<divid="div1">div 1

</div>

<divid="div2">div 2

</div>

Output:

<divstyle="background-color:yellow"> New div </div>

<divid="div1">div 1

</div>

<divid="div2">div 2

</div>

## jQuery append() method:

The jQuery append() method inserts content to the end of target element(s) which is specified by a selector.

Syntax:

$('selector expression').append('content');

Example: jQuery append() method

$('p').append('World!');

<p>Hello </p>

Output:

<p>Hello World!</p>

## jQuery prepend() method:

The jQuery prepend() method inserts content at the beginning of an element(s) specified by a selector.

Syntax:

$('selector expression').prepend('content');

Example: jQuery prepend() method

$('div').prepend('<p>This is prepended paragraph</p>');

<div>

<label>This is div.</label>

</div>

Output:

<div>

<p>This is prepended paragraph</p>

<label>This is div.</label>

</div>

## jQuery remove() method:

The jQuery remove() method removes element(s) as specified by a selector.

Syntax:

$('selector expression').remove();

Example: jQuery remove() method

$('label').remove();

<div>This is div.

<label>This is label.</label>

</div>

Output:

<div>

This is div.

</div>

## jQuery replaceAll() method:

The jQuery replaceAll() method replaces all target elements with specified element(s).

Syntax:

$('content string').replaceAll('selector expression');

Example: jQuery replaceAll() method

$('<span>This is span</span>').replaceAll('p');

<div>

<p>This is paragraph.</p>

</div>

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

Output:

<div>

<span>This is span</span>

</div>

<span>This is span</span>

## jQuery wrap() method:

The jQuery wrap() method wrap each target element with specified content element.

Syntax:

$('selector expression').wrap('content string');

Example: jQuery wrap() method

$('span').wrap('<p></p>');

<div>

<span>This is span.</span>

</div>

<span>This is span.</span>

Output:

<div>

<p><span>This is span.</span></p>

</div>

<p><span>This is span.</span></p>

# **Manipulate HTML Attributes using jQuery:**

The following table lists jQuery methods to get or set value of attribute, property, text or html.

| **jQuery Method** | **Description** |
| --- | --- |
| attr() | Get or set the value of specified attribute of the target element(s). |
| prop() | Get or set the value of specified property of the target element(s). |
| html() | Get or set html content to the specified target element(s). |
| text() | Get or set text for the specified target element(s). |
| val() | Get or set value property of the specified target element. |

## jQuery attr() method:

jQuery attr() method is used to get or set the value of specified attribute of DOM element.

Syntax:

$('selector expression').attr('name','value');

Example: jQuery attr() method

$('p').attr('style'); // returns "font-size:16px;font-weight:bold"

$('div').attr('class','yellowDiv'); // adds class='divCls' to each div element

<div>

This is div.

</div>

<p style="font-size:16px;font-weight:bold">

This is paragraph.

</p>

<div>

This is div.

</div>

<p>

This is paragraph.

</p>

## jQuery prop() method:

The jQuery prop() method gets or sets the value of specified property to the DOM element(s).

Syntax:

$('selector expression').prop('name','value');

Example: jQuery prop() method

var style = $('p').prop('style');

style.fontWeight; // returns "bold"

$('div').prop('class','yellowDiv'); // add class="yellowDiv" to all div elements

<div>

This is div.

</div>

<p style="font-size:16px;font-weight:bold">

This is paragraph.

</p>

<div>

This is div.

</div>

<p>

This is paragraph.

</p>

# **Manipulate DOM Element's Dimensions using jQuery:**

The jQuery library includes various methods to manipulate DOM element's dimensions like height, width, offset, position etc.

The following table lists all the jQuery methods to get or set DOM element's dimensions.

| **jQuery Method** | **Description** |
| --- | --- |
| height() | Get or set height of the specified element(s). |
| innerHeight() | Get or set inner height (padding + element's height) of the specified element(s). |
| outerHeight() | Get or set outer height (border + padding + element's height) of the specified element(s). |
| offset() | Get or set left and top coordinates of the specified element(s). |
| position() | Get the current coordinates of the specified element(s). |
| width() | Get or set the width of the specified element(s). |
| innerWidth() | Get or set the inner width (padding + element's width) of the specified element(s). |
| outerWidth() | Get or set outer width (border + padding + element's width) of the specified element(s). |

## jQuery offset() method:

The jQuery offset() method gets or sets coordinates of the specified element(s).

Syntax:

$('selector expression').offset(options);

Example: jQuery offset() method

varofs = $('#myDiv').offset();

alert('left:' + ofs.left + ', top: ' + ofs.top);

$('p').offset({ left:100, top:200});

<divid="myDiv">

This is div.

</div>

<p>

This is paragraph.

</p>

# **Traversing DOM Elements using jQuery:**

The jQuery library includes various methods to traverse DOM elements in a DOM hierarchy.

The following table lists jQuery methods for traversing DOM elements.

| **jQuery Methods** | **Description** |
| --- | --- |
| children() | Get all the child elements of the specified element(s) |
| each() | Iterate over specified elements and execute specified call back function for each element. |
| find() | Get all the specified child elements of each specified element(s). |
| first() | Get the first occurrence of the specified element. |
| next() | Get the immediately following sibling of the specified element. |
| parent() | Get the parent of the specified element(s). |
| prev() | Get the immediately preceding sibling of the specified element. |
| siblings() | Get the siblings of each specified element(s) |

## jQuery each() method:

The jQuery each() method iterates over each specified element (specified using selector) and executes callback function for each element.

Syntax:

$('selector expression').each(callback function);

Example: jQuery each() method

$('p').each(function (index) {

alert('index' + index + ', text: ' + $(this).text());

});

<div>

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

</div>

<divid="myDiv">

<p>

This is second paragraph.

</p>

<divid="inrDiv">

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

</div>

<div>

<ul>

<li>First</li>

<li>Second</li>

<li>Third</li>

</ul>

</div>

</div>

<div>

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

</div>

## jQuery next() method:

The jQuery next() method gets the immediately following sibling of the specified element.

Syntax:

$('selector expression').next();

Example: jQuery next() method

alert('Next element to #myDiv: ' + $('#myDiv').next().html());

alert('Next element to #inrDiv: ' + $('#inrDiv').next().html());

<div>

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

</div>

<divid="myDiv">

<p>

This is second paragraph.

</p>

<divid="inrDiv">

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

</div>

<div>

<ul>

<li>First</li>

<li>Second</li>

<li>Third</li>

</ul>

</div>

</div>

<div>

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

</div>

## jQuery parent() method:

The jQuery parent() method gets the immediate parent element of the specified element.

Syntax:

$('selector expression').parent();

Example: jQuery parent() method

alert('Parent element of #inrDiv: ' + $('#inrDiv').parent().html());

<div>

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

</div>

<divid="myDiv">

<p>

This is second paragraph.

</p>

<divid="inrDiv">

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

</div>

<div>

<ul>

<li>First</li>

<li>Second</li>

<li>Third</li>

</ul>

</div>

</div>

<div>

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

</div>

# **jQuery Traversing –**Filtering

The first(), last(), eq(), filter() and not() Methods

The most basic filtering methods are first(), last() and eq(), which allow you to select a specific element based on its position in a group of elements.

Other filtering methods, like filter() and not() allow you to select elements that match, or do not match, a certain criteria.

The first() method returns the first element of the specified elements.

The following example selects the first <div> element:

### **Example**

$(document).ready(function(){  
    $("div").first();  
});

The last() method returns the last element of the specified elements.

The following example selects the last <div> element:

### **Example**

$(document).ready(function(){  
    $("div").last();  
});

The eq() method returns an element with a specific index number of the selected elements.

The index numbers start at 0, so the first element will have the index number 0 and not 1. The following example selects the second <p> element (index number 1):

### **Example**

$(document).ready(function(){  
    $("p").eq(1);  
});

The filter() method lets you specify a criteria. Elements that do not match the criteria are removed from the selection, and those that match will be returned.

The following example returns all <p> elements with class name "intro":

### **Example**

$(document).ready(function(){  
    $("p").filter(".intro");  
});

The not() method returns all elements that do not match the criteria.

**Tip:** The not() method is the opposite of filter().

The following example returns all <p> elements that do not have class name "intro":

### **Example**

$(document).ready(function(){  
    $("p").not(".intro");  
});

# **CSS Manipulation using jQuery:**

The jQuery library includes various methods to manipulate style properties and CSS class of DOM element(s).

The following table lists jQuery methods for styling and css manipulation.

| **jQuery Methods** | **Description** |
| --- | --- |
| css() | Get or set style properties to the specified element(s). |
| addClass() | Add one or more class to the specified element(s). |
| hasClass() | Determine whether any of the specified elements are assigned the given CSS class. |
| removeClass() | Remove a single class, multiple classes, or all classes from the specified element(s). |
| toggleClass() | Toggles between adding/removing classes to the specified elements |

# **jQuery Animation:**

jQuery includes methods which give special effects to the elements on hiding, showing, changing style properties, and fade-in or fade-out operation. These special effect methods can be useful in building an interactive user interface.

The following table lists jQuery methods for adding special effects to the DOM elements.

|  |  |
| --- | --- |
| fadeIn() | Display specified element(s) by fading them to opaque. |
| fadeOut() | Hides specified element(s) by fading them to transparent. |
| fadeTo() | Adjust the opacity of the specified element(s) |
| fadeToggle() | Display or hide the specified element(s) by animating their opacity. |
| hide() | Hide specified element(s). |
| show() | Display specified element(s). |
| toggle() | Display hidden element(s) or hide visible element(s). |
| slideUp() | Hide specified element(s) with sliding up motion. |
| slideDown() | Display specified element(s) with sliding down motion. |
| slideToggle() | Display or hide specified element(s) with sliding motion. |

## jQuery Event Methods:

The jQuery library provides methods to handle DOM events. Most jQuery methods correspond to native DOM events.

The following table lists all jQuery methods and corresponding DOM events.

| **Category** | **jQuery Method** | **DOM Event** |
| --- | --- | --- |
| Form events | Blur | onblur |
|  | Change | onchange |
|  | Focus | onfocus |
|  | Focusin | onfocusin |
|  | Select | onselect |
|  | submit | onsubmit |
| Keyboard events | keydown | onkeydown |
|  | keypress | onkeypress |
|  | keyup | onkeyup |
|  | focusout |  |
| Mouse events | click | onclick |
|  | dblclick | ondblclick |
|  | focusout |  |
|  | hover |  |
|  | mousedown | onmousedown |
|  | mouseenter | onmouseenter |
|  | mouseleave | onmouseleave |
|  | mousemove | onmousemove |
|  | mouseout | onmouseout |
|  | mouseover | onmouseover |
|  | mouseup | onmouseup |
|  | Toggle |  |
| Browser events | Error | onerror() |
|  | Resize | onresize |
|  | Scroll | onscroll |
| Document loading | Load | onload |
|  | Ready |  |
|  | Unload | onunload |

# **jQuery Ajax Introduction:**

Ajax stands for "Asynchronous JavaScript and XML".

JavaScript includes features of sending asynchronous http request using XMLHttpRequest object. Ajax is about using this ability of JavaScript to send asynchronous http request and get the xml data as a response (also in other formats) and update the part of a web page (using JavaScript) without reloading or refreshing entire web page.

Advantages of jQuery Ajax:

1. Cross-browser support
2. Simple methods to use
3. Ability to send GET and POST requests
4. Ability to Load JSON, XML, HTML or Scripts

The following table lists all the Ajax methods of jQuery.

| **jQuery Ajax Methods** | **Description** |
| --- | --- |
| ajax() | Sends asynchronous http request to the server. |
| get() | Sends http GET request to load the data from the server. |
| Post() | Sends http POST request to submit or load the data to the server. |
| getJSON() | Sends http GET request to load JSON encoded data from the server. |
| getScript() | Sends http GET request to load the JavaScript file from the server and then executes it. |
| load() | Sends http request to load the html or text content from the server and add them to DOM element(s). |

The jQuery library also includes following events which will be fired based on the state of the Ajax request.

| **jQuery Ajax Events** | **Description** |
| --- | --- |
| ajaxComplete() | Register a handler function to be called when Ajax requests complete. |
| ajaxError() | Register a handler function to be called when Ajax requests complete with an error. |
| ajaxSend() | Register a handler function to be called before Ajax request is sent. |
| ajaxStart() | Register a handler function to be called when the first Ajax request begins. |
| ajaxStop() | Register a handler function to be called when all the Ajax requests have completed. |
| ajaxSuccess() | Register a handler function to be called when Ajax request completes successfully. |

1. $.ajax() method allows to send asynchronous http requests to submit or retrieve data from the server without reloading the whole page.
2. $.ajax() can be used to send http GET, POST, PUT, DELETE etc. request. It can retrieve any type of response from the server.
3. Syntax: $.ajax(url,[options])
4. Use option parameter to customize ajax request as per your need

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

<div id="resultarea"></div>

**Script:**

var myusername = $("#username").val();

$.ajax({

type: "GET",

url: "serverscript.xxx",

data: myusername,

cache: false,

success: function(data){

$("#resultarea").text(data);

}

});

* accepts: The content type sent in the request header that tells the server what kind of response it will accept in return
* async: Set this options to false to perform a synchronous request
* beforeSend: A pre-request callback function that can be used to modify the jqXHR object before it is sent
* cache: Set this options to false to force requested pages not to be cached by the browser
* complete: A function to be called when the request finishes (after success and error callbacks are executed)
* contentType: The content type of the data sent to the server
* data: The data to send to the server when performing the Ajax request
* dataType: The type of data expected back from the server
* error: A function to be called if the request fails
* headers: An object of additional headers to send to the server
* jsonp: A string to override the callback function name in a JSONP request
* mimeType: A string that specifies the mime type to override the XHR mime type
* password: A password to be used with XMLHttpRequest in response to an HTTP access authentication request
* statusCode: An object of numeric HTTP codes and functions to be called when the response has the corresponding code
* success: A function to be called if the request succeeds
* timeout: A number that specifies a timeout (in milliseconds) for the request
* type: The type of request to make, which can be either “POST” or “GET”
* url: A string containing the URL to which the request is sent

**Real time question:**

Tooltip:

**function**openTooltip(ele, type) {

ele = ele.getBoundingClientRect();

**var**topValue=0;

**var**leftValue = ele.left - 100;

**if** (ele.top>300 ){

topValue= ele.top - 310;

} **else** {

topValue=ele.top + 15;

}

**if** (ele.left> 250){

leftValue = ele.left - 250;

} **else** {

leftValue = ele.left + 15;

}

$(".tooltipOverlay").css("top",topValue);

$(".tooltipOverlay").css("left",leftValue);

$('.tooltipoverlay').hide();

$("#"+type+"Div").show();

}

**function**closeTooltip(type) {

$("#"+type+"Div").hide();

## Removing an item from a select box using jQuery

<selectid="selectBox"name="selectBox">

<optionvalue="option1"> option1 </option>

<optionvalue="option2"> option2 </option>

<optionvalue="option3"> option3 </option>

<optionvalue="option4"> option4 </option>

</select>

## Remove From Select Drop Down

$("#selectBox option[value='option1']").remove();

## Add Option To Select Drop Down

$("#selectBox").append('<option value="option6">option6</option>');

## Remove All Options Except The First

$('select').children('option:not(:first)').remove();

**Add Options From Array**

$.each(selectValues, function(key, value) {

$('#mySelect')

.append($("<option></option>")

.attr("value",key)

.text(value));

});

**References:**

1. <https://api.jquery.com>
2. <https://www.tutorialspoint.com/jquery>
3. <http://www.tutorialsteacher.com/jquery>
4. <http://www.jquery-tutorial.net>