**Introduction to jQuery:**

**What is jQuery?**

jQuery is a JavaScript Library and greatly simplifies JavaScript programming. The purpose of jQuery is to make it much easier to use JavaScript on 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.

## 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

**Features of jQuery**

* jQuery is a small and lightweight JavaScript library.
* jQuery is cross-platform (cross Browser).
* jQuery means "write less do more".
* jQuery simplifies AJAX call and DOM manipulation.
* jQuery speed is high compare to Javascript.
* jQuery reduce the Complexity of the Javascript code.
* jQuery execution of script is faster than Javascript.

**Syntax of jQuery:**

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".

## The Document Ready Event

**1.Syntax is:**

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

**OR**

**Syntax :**

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

**Note:** While writing the jQuery syntax, instead of $ we can put jQuery, the only difference is that $ supports higher version browsers (firefox) where as jQuery supports lower version browsers (internet explorer 1.4,1.8).

The above syntax 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 us to have our JavaScript code before the body of our 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

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

**2.Syntax:**

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

**Note:** It’s better practice to use 1.syntax because, 2.Syntax unnecessary loads the functions.

# jQuery Selectors

jQuery Selectors are used to select and manipulate HTML elements. They are very important part of jQuery library.

With jQuery selectors, you can find or select HTML elements based on their id, classes, attributes, types and much more from a DOM.

In simple words, you can say that selectors are used to select one or more HTML elements using jQuery and once the element is selected then you can perform various operation on that.

All jQuery selectors start with a dollar sign and parenthesis e.g. $(). It is known as the factory function.

## The $() factory function

Every jQuery selector start with this sign $(). This sign is known as the factory function. It uses the three basic building blocks while selecting an element in a given document.

|  |  |  |
| --- | --- | --- |
| **S.No.** | **Selector** | **Description** |
| 1) | Tag Name: | It represents a tag name available in the DOM. For example: $('p') selects all paragraphs 'p' in the document. |
| 2) | Tag ID: | It represents a tag available with a specific ID in the DOM. For example: $('#real-id') selects a specific element in the document that has an ID of real-id. |
| 3) | Tag Class: | It represents a tag available with a specific class in the DOM. For example: $('real-class') selects all elements in the document that have a class of real-class. |

## How to use Selectors

The jQuery selectors can be used single or with the combination of other selectors. They are required at every steps while using jQuery. They are used to select the exact element that you want from your HTML document.

|  |  |  |
| --- | --- | --- |
| **S.No.** | **Selector** | **Description** |
| 1) | Name: | It selects all elements that match with the given element name. |
| 2) | #ID: | It selects a single element that matches with the given id. |
| 3) | .Class: | It selects all elements that matches with the given class. |
| 4) | Universal(\*) | It selects all elements available in a DOM. |
| 5) | Multiple Elements A,B,C | It selects the combined results of all the specified selectors A,B and C. |

## Different jQuery Selectors

|  |  |  |
| --- | --- | --- |
| **Selector** | **Example** | **Description** |
| \* | $("\*") | It is used to select all elements. |
| #id | $("#firstname") | It will select the element with id="firstname" |
| .class | $(".primary") | It will select all elements with class="primary" |
| class,.class | $(".primary,.secondary") | It will select all elements with the class "primary" or "secondary" |
| element | $("p") | It will select all p elements. |
| el1,el2,el3 | $("h1,div,p") | It will select all h1, div, and p elements. |
| :first | $("p:first") | This will select the first p element |
| :last | $("p:last") | This will select he last p element |
| :even | $("tr:even") | This will select all even tr elements |
| :odd | $("tr:odd") | This will select all odd tr elements |
| :first-child | $("p:first-child") | It will select all p elements that are the first child of their parent |
| :first-of-type | $("p:first-of-type") | It will select all p elements that are the first p element of their parent |
| :last-child | $("p:last-child") | It will select all p elements that are the last child of their parent |
| :last-of-type | $("p:last-of-type") | It will select all p elements that are the last p element of their parent |
| :nth-child(n) | $("p:nth-child(2)") | This will select all p elements that are the 2nd child of their parent |
| :nth-last-child(n) | $("p:nth-last-child(2)") | This will select all p elements that are the 2nd child of their parent, counting from the last child |
| :nth-of-type(n) | $("p:nth-of-type(2)") | It will select all p elements that are the 2nd p element of their parent |
| :nth-last-of-type(n) | $("p:nth-last-of-type(2)") | This will select all p elements that are the 2nd p element of their parent, counting from the last child |
| :only-child | $("p:only-child") | It will select all p elements that are the only child of their parent |
| :only-of-type | $("p:only-of-type") | It will select all p elements that are the only child, of its type, of their parent |
| parent > child | $("div > p") | It will select all p elements that are a direct child of a div element |
| parent descendant | $("div p") | It will select all p elements that are descendants of a div element |
| element + next | $("div + p") | It selects the p element that are next to each div elements |
| element ~ siblings | $("div ~ p") | It selects all p elements that are siblings of a div element |
| :eq(index) | $("ul li:eq(3)") | It will select the fourth element in a list (index starts at 0) |
| :gt(no) | $("ul li:gt(3)") | Select the list elements with an index greater than 3 |
| :lt(no) | $("ul li:lt(3)") | Select the list elements with an index less than 3 |
| :not(selector) | $("input:not(:empty)") | Select all input elements that are not empty |
| :header | $(":header") | Select all header elements h1, h2 ... |
| :animated | $(":animated") | Select all animated elements |
| :focus | $(":focus") | Select the element that currently has focus |
| :contains(text) | $(":contains('Hello')") | Select all elements which contains the text "Hello" |
| :has(selector) | $("div:has(p)") | Select all div elements that have a p element |
| :empty | $(":empty") | Select all elements that are empty |
| :parent | $(":parent") | Select all elements that are a parent of another element |
| :hidden | $("p:hidden") | Select all hidden p elements |
| :visible | $("table:visible") | Select all visible tables |
| :root | $(":root") | It will select the document's root element |
| :lang(language) | $("p:lang(de)") | Select all p elements with a lang attribute value starting with "de" |
| [attribute] | $("[href]") | Select all elements with a href attribute |
| [attribute=value] | $("[href='default.htm']") | Select all elements with a href attribute value equal to "default.htm" |
| [attribute!=value] | $("[href!='default.htm']") | It will select all elements with a href attribute value not equal to "default.htm" |
| [attribute$=value] | $("[href$='.jpg']") | It will select all elements with a href attribute value ending with ".jpg" |
| [attribute|=value] | $("[title|='Tomorrow']") | Select all elements with a title attribute value equal to 'Tomorrow', or starting with 'Tomorrow' followed by a hyphen |
| [attribute^=value] | $("[title^='Tom']") | Select all elements with a title attribute value starting with "Tom" |
| [attribute~=value] | $("[title~='hello']") | Select all elements with a title attribute value containing the specific word "hello" |
| [attribute\*=value] | $("[title\*='hello']") | Select all elements with a title attribute value containing the word "hello" |
| :input | $(":input") | It will select all input elements |
| :text | $(":text") | It will select all input elements with type="text" |
| :password | $(":password") | It will select all input elements with type="password" |
| :radio | $(":radio") | It will select all input elements with type="radio" |
| :checkbox | $(":checkbox") | It will select all input elements with type="checkbox" |
| :submit | $(":submit") | It will select all input elements with type="submit" |
| :reset | $(":reset") | It will select all input elements with type="reset" |
| :button | $(":button") | It will select all input elements with type="button" |
| :image | $(":image") | It will select all input elements with type="image" |
| :file | $(":file") | It will select all input elements with type="file" |
| :enabled | $(":enabled") | Select all enabled input elements |
| :disabled | $(":disabled") | It will select all disabled input elements |
| :selected | $(":selected") | It will select all selected input elements |
| :checked | $(":checked") | It will select all checked input elements |

## What are Events?

All the different visitor's 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** | **Keyboard Events** | **Form Events** | **Document/Window Events** |
| click | keypress | submit | load |
| dblclick | keydown | change | resize |
| mouseenter | keyup | focus | scroll |
| mouseleave |  | blur | unload |

**jQuery Effects**

jQuery enables us to add effects on a web page. jQuery effects can be categorized into fading, sliding, hiding/showing and animation effects.

**jQuery Effects are:**

1. Display effects

2. Fading effects

3. Sliding effects

4. Other effects

**1. Display effects:**

* **hide() –** Hides the matched or selected elements.

Example:

 $("p").hide(); // hides the paragraph content.

* **show()** – Displays or Shows the selected elements.

Example:

 $("p").show(); // Displays the paragraph content.

* **Toggle()** - shows or hides the matched elements. In other words, it toggles between the hide() and show() methods.

Example:

 $("p").toggle(); // Hides and Displays the paragraph content.

**2. Fading effects:**

* **fadeIn() -**  method is used to fade in the element.

**Syntax**:

1. $(selector).fadeIn();

2. $(selector).fadeIn (speed, callback);

3. $(selector).fadeIn (speed, easing, callback);

**speed**: It is an optional parameter. It specifies the speed of the delay. Its possible vales are slow, fast and milliseconds.

**easing**: It specifies the easing function to be used for transition.

**callback**: It is also an optional parameter. It specifies the function to be called after completion of fadein() effect.

* **fadeOut() -**  method is used to fade out the element.

**Syntax**:

1. $(selector).fadeOut();

2. $(selector).fadeOut(speed, callback);

3. $(selector).fadeOut(speed, easing, callback);

* **fadeToggle()** - method is used to toggle between the fadeIn() and fadeOut() methods. If the elements are faded in, it will make them faded out and if they are faded out it will make them faded in.

**Syntax**:

1. $(selector).fadeToggle();

2. $(selector).fadeToggle(speed, callback);

3. $(selector).fadeToggle(speed, easing, callback);

* **fadeTo() -** method is used to fading to a given opacity.

**Syntax**:

1. $(selector).fadeTo(speed, opacity);

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

3. $(selector).fadeTo(speed, opacity, easing, callback);

**3. Sliding effects:**

The jQuery slide methods slide elements up and down.

* **slideDown()** - method is used to slide down an element.

**Syntax**:

$(selector).slideDown(speed);

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

$(selector).slideDown(speed, easing, callback);

**speed**: It specifies the speed of the delay. Its possible vales are slow, fast and milliseconds.

**easing**: It specifies the easing function to be used for transition.

**callback**: It is also an optional parameter. It specifies the function to be called after completion of slideDown() effect.

* **slideDown()** - method is used to slide up an element.

**Syntax**:

$(selector).slideUp(speed);

$(selector).slideUp(speed, callback);

$(selector).slideUp(speed, easing, callback);

* slideToggle () - method is used to toggle between slideUp() and slideDown() method. If the element is slide down, it will slide up the element and if it is slide up, it will slide down.

**Syntax**:

$(selector).slideToggle(speed);

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

$(selector).slideToggle(speed, easing, callback);

**4. Other effects**

* animate() - method provides you a way to create custom animations.

**Syntax**:

$(selector).animate({params}, speed, callback);

Here,

**params** parameter defines the CSS properties to be animated.

**speed** parameter is optional and specifies the duration of the effect. It can be set as "slow", "fast" or milliseconds.

c**allback** parameter is also optional and it is a function which is executed after the animation completes.

* **delay() -** method is used to delay the execution of functions in the queue. It is a best method to make a delay between the queued jQuery effects. The jQuery delay () method sets a timer to delay the execution of the next item in the queue.

**Syntax:**

$(selector).delay (speed, queueName)

**speed**: It is an optional parameter. It specifies the speed of the delay. Its possible vales are slow, fast and milliseconds.

**queueName**: It is also an optional parameter. It specifies the name of the queue. Its default value is "fx" the standard queue effect.

**jQuery HTML/CSS**

# jQuery - Get Content and Attributes

* jQuery contains powerful methods for changing and manipulating HTML elements and attributes.

## jQuery DOM Manipulation

* One very important part of jQuery is the possibility to manipulate the DOM.
* jQuery comes with a bunch of DOM related methods that make it easy to access and manipulate elements and attributes.

## Get (Return) and Set Content - text(), html(), and val()

Three simple, but useful, jQuery methods for DOM manipulation are:

* 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

# text() - The jQuery text() method is used to set or return the text content of the selected elements.

**To return content:** When this method is used to return content, it returns the combined text content of all matched elements without the HTML markup.

**To set content:** When this method is used to set content, it overwrites the content of all matched elements.

**Syntax**:

$(selector).text()   // **To return text content**

$(selector).text(content)   // **To set text content**

$(selector).text(function(index, currentcontent))  // **To set text content using a function**

## Parameters of jQuery text() method

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| Content | It is a mandatory parameter. It specifies the new text content for the selected elements. The special characters will be encoded in this parameter. |
| Function (index,currentcontent) | It is an optional parameter. It specifies the function that returns the new text content for the selected elements.   * **Index:** It provides the index position of the element in the set. * **Currentcontent:** It provides the current content of the selected elements. |

# html() - jQuery html() method is used to change the entire content of the selected elements. It replaces the selected element content with new contents.

**Note:** It is a very useful function but works in a limited area because of its API documentation. The API documentation of the jQuery html function consists of three method signatures.

The first method signature has no argument, so it just returns the HTML within that element. The remaining two signatures take a single argument: i.e. a string or a function that returns a string.

**Syntax**:

$(selector).html()  // **It is used to return content**

$(selector).html(content)  // **It is used to set content**

$(selector).html(function (index, currentcontent))  // **It is used to set content by calling function**

# val()

# There are two usage of jQuery val() method.

* It is used to get current value of the first element in the set of matched elements.
* It is used to set the value of every matched element.

**Syntax**:

$(selector).val()  // **It is used to get value**

$(selector).val(value)  // **It is used to set value**

$(selector).val(function(index,currentvalue))   // **It is used to set value using function**

## Parameters of jQuery val() method

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| Value | It is a mandatory parameter. It is used specify the value of the attribute. |
| Function(index, currentvalue) | It is an optional parameter. It is used to specify a function that returns the value to set. |

# attr() - The jQuery attr() method is used to set or return attributes and values of the selected elements.

There are two usage of jQuery attr() method.

1. **To return attribute value**: This method returns the value of the first matched element.
2. **To set attribute value**: This method is used to set one or more attribute/value pairs of the set of matched elements.

**Syntax**:

$(selector).attr(attribute)  // **To return an attribute's value**

$(selector).attr(attribute,value)  // **To set an attribute and value**

$(selector).attr(attribute,function(index,currentvalue))  // **To set an attribute and value by using a function**

$(selector).attr({attribute:value, attribute:value,...})   // **To set multiple attributes and values**

## Parameters of jQuery attr() method

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| Attribute | This parameter is used to specify the name of the attribute. |
| Value | This parameter is used to specify the value of the attribute. |
|  |  |
| Function (index, currentvalue) | It is a parameter to specify a function that returns an attribute value to set.   * **Index:** It is used to receive the index position of the element in the set. * **Currentvalue:** It is used to provide the current attribute value of selected elements. |

# jQuery - Add Elements

With jQuery, it is easy to add new elements/content.

## Add New HTML Content

jQuery 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

# append() - The jQuery append() method is used to insert specified content as the last child (at the end of) the selected elements in the jQuery collection.

The append () and appendTo () methods are used to perform the same task. The only difference between them is in the syntax.

**Syntax**:

$(selector).append(content, function(index, html))  // append()

$(content).appendTo(selector)   // appendTo()

## Parameters of jQuery append() method

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| Content | It is a mandatory parameter. It specifies the content which you want to insert. Its possible values are:   * HTML elements * jQuery objects * DOM elements |
| Function (index,html) | It is an optional parameter. It specifies the function that returns the content to insert.   * **Index:** It returns the index position of the element in the set. * **HTML:** It returns the current HTML of the selected element. |

# prepend() - The jQuery prepend() method is used to insert the specified content at the beginning (as a first child) of the selected elements. It is just the opposite of the jQuery append() method.

**Note:** If we want to insert the content at the end of the selected elements, we should use the append method.

**Syntax**:

$(selector).prepend(content, function (index, html))

# after() - The jQuery after() method is used to insert specified content after the selected element. It is just like jQuery append() method.

# before() - The jQuery before() method is used to insert specified content before the selected element. It is just like jQuery prepend() method.

**Syntax**:

$(selector).after(content, function(index))   // after()

$(selector).before(content, function(index))   // before()

# jQuery - Remove Elements

With jQuery, it is easy to remove existing HTML elements.

## Remove Elements/Content

To remove elements and content, there are mainly two jQuery methods:

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

# remove() - The jQuery remove() method is used to remove the selected elements out of the DOM. It removes the selected element itself, as well as everything inside it (including all texts and child nodes). This method also removes the data and the events of the selected elements.

**Syntax**:

$(selector).remove(selector)

**Parameters of jQuery remove() method:**

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| Selector | is an optional parameter. It specifies whether to remove one or more elements. If you have to remove more than one element then you should separate them with comma (,). |

# empty() - The jQuery empty() method is used to remove all child nodes and content from the selected elements. This method doesn't remove the element itself.

**Syntax**:

$(selector).empty()

# jQuery - Get and Set CSS Classes

With jQuery, it is easy to manipulate the CSS of elements.

## jQuery Manipulating CSS

jQuery has several methods for CSS manipulation. We will look at the following 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

# addClass() - The addclass() method is used to add one or more class name to the selected element. This method is used only to add one or more class names to the class attributes not to remove the existing class attributes.

**Note:** If we want to add more than one class, separate the class names with spaces.

**Syntax**:

$(selector).addClass(classname, function(index, oldclass))

## Parameters of jQuery addClass() method

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| Classname | It is a mandatory parameter. It specifies one or more class names which you want to add. |
| Function(index, currentclass) | It is an optional parameter. It specifies a function that returns one or more class names to be added.   * Index: It is used to provide the index position of the element in the set. * Currentclass: It is used to return the current class name of the selected element. |

# removeClass() - The removeclass() method is used to remove one or more class name to the selected element.

**Syntax**:

$(selector).removeClass(classname, function(index, oldclass))

# toggleClass() - The jQuery toggleCLass() method is used to add or remove one or more classes from the selected elements. This method toggles between adding and removing one or more class name. It checks each element for the specified class names. If the class name is already set, it removes and if the class name is missing, it adds.

In this way, it creates the toggle effect. It also facilitates us to specify to only add or only remove by the use of switch parameter.

**Syntax**:

$(selector).toggleClass(classname, function(index, currentclass), switch)

## Parameters of jQuery toggleClass() method

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| classname | It is a mandatory parameter. It specifies one or more class name to add or remove. If you use several classes then separate them by space. |
| function(index, currentclass) | It is an optional parameter. It specifies one or more class names that you want to add or remove.   * **Index:** It provides the index position of the element in the set. * **Currentclass:** It provides the current class name of the selected element. |
| switch | It is also an optional parameter. It is a Boolean value which specifies whether the class should be added (true) or removed (false). |

# css() - The jQuery CSS() method is used to get (return) or set style properties or values for selected elements. It facilitates us to get one or more style properties.

**jQuery CSS() method provides two ways:**

## 1) Return a CSS property

It is used to get the value of a specified CSS property.

**Syntax**:

css("propertyname");

**Example:**

<!DOCTYPE html>

<html>

<head>

<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>

<script>

$(document).ready(function(){

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

alert("Background color = " + $("p").css("background-color"));

});

});

</script>

</head>

<body>

<h2>This is a heading</h2>

<p style="background-color:#ff0000">The background-color of this paragraph is red.</p>

<p style="background-color:#00ff00">The background-color of this paragraph is green.</p>

<p style="background-color:#0000ff">The background-color of this paragraph is blue.</p>

<button>Click here to get the background-color of first matched element.</button>

</body>

</html> // **After clicking button it displays value of css property, Background color =rgb(255,0,0)**

## 2) Set a CSS property

This property is used to set a specific value for all matched element.

**Syntax**:

css("propertyname","value");

## 3) Set multiple CSS properties

It is just an extension of Set CSS property. It facilitates us to add multiple property values together.

**Syntax**:

css({"propertyname":"value","propertyname":"value",...});

# jQuery - Dimensions

With jQuery, it is easy to work with the dimensions of elements and browser window.

## jQuery Dimension Methods

jQuery has several important methods for working with dimensions:

* width()
* height()
* innerWidth()
* innerHeight()
* outerWidth()
* outerHeight()

**jQuery width()** - method is used to return or set the width of matched element.

**To return width:** When this method is used to return the width, it returns the width of first matched element.

**To set width:** When this method is used to set the width, it sets the width for every matched element.

**Syntax**:

$(selector).width()  // To return the width:

$(selector).width(value)  // To set the width:

$(selector).width(function(index, currentwidth)) // To set the width using a function:

**Parameters of jQuery width() method**

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| Value | It is a mandatory parameter. It is used for setting width. It specifies the width in px, em, pt etc. The default value of jQuery width() method is px. |
| Function(index, currentwidth) | It is an optional parameter. It specifies a function that provides the new width of selected element.   * **Index:** It provides the index position of the element in the set. * **currentwidth:**It provides the current width of the selected element. |

# jQuery height() - The jQuery height() method is used to return the current computed height for the first element or set the height of every matched element. In other words, you can say that the height() method is used for two purposes:

**To return height:** When this method is used to return height, it returns the height of first matched element.

**To set height:** When this method is used to set height, it sets height of all matched elements.

**Syntax**:

$(selector).height()  // To return the height:

$(selector).height(value) // To set the height:

$(selector).height(function(index, currentheight))  // To set the height by using a function:

## Parameters of jQuery height() method

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| Value | This is a mandatory parameter. It specifies the height in px, em, pt, etc. its defauly unit is px. |
| Function(index, currentHeight) | This is an optional parameter. This is used to specify a function that returns the new height of the selected element.   * **Index:** It provides the index position of the element in the set. * **currentHeight:** It provides the current height of the selected element. |

**jQuery innerWidth() -** method is used to return the inner width of the first matched element without including border and margin.

This method includes padding but excludes border and margin.

**Syntax**:

$(selector).innerWidth()

**jQuery innerHeight ()** - method is used to return the inner height of first matched element. It includes padding but not border and margin.

**Syntax**:

$(selector).innerHeight()

**jQuery outerWidth()** - method is used to return the outer width of the first matched element with padding and border.

The jQuery outerWidth () method works for both visible and hidden elements.

**Syntax**:

$(selector).outerWidth(includeMargin)

**jQuery outerHeight() -** method is used to return the outer height of first matched element. This method includes padding and border both.

**Syntax**:

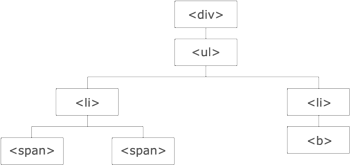
$(selector).outerHeight(includeMargin)

# jQuery Traversing

## What is Traversing?

jQuery traversing, which means "move through", are used to "find" (or select) HTML elements based on their relation to other elements. Start with one selection and move through that selection until you reach the elements you desire.

The image below illustrates a family tree. With jQuery traversing, you can easily move up (ancestors), down (descendants) and sideways (siblings) in the family tree, starting from the selected (current) element. This movement is called traversing - or moving through - the DOM.



**Illustration explained:**

* The <div> element is the**parent** of <ul>, and an **ancestor** of everything inside of it
* The <ul> element is the **parent** of both <li> elements, and a **child** of <div>
* The left <li> element is the **parent** of <span>, **child** of <ul> and a **descendant** of <div>
* The <span> element is a **child** of the left <li> and a **descendant** of <ul> and <div>
* The two <li> elements are **siblings** (they share the same parent)
* The right <li> element is the **parent** of <b>, **child** of <ul> and a **descendant** of <div>
* The <b> element is a **child** of the right <li> and a **descendant** of <ul> and <div>

# jQuery Traversing - Ancestors

An ancestor is a parent, grandparent, great-grandparent, and so on.With jQuery you can traverse up the DOM tree to find ancestors of an element.

## Traversing Up the DOM Tree

Three useful jQuery methods for traversing up the DOM tree are:

* parent()
* parents()
* parentsUntil()

## jQuery parent() Method

The parent() method returns the direct parent element of the selected element.

This method only traverse a single level up the DOM tree.

The following example returns the direct parent element of each <span> elements:

$(document).ready(function(){  
$("span").parent();  
});

## jQuery parents() Method

The parents() method returns all ancestor elements of the selected element, all the way up to the document's root element (<html>)

The following example returns all ancestors of all <span> elements:

### Example

$(document).ready(function(){  
$("span").parents();  
});

we can also use an optional parameter to filter the search for ancestors.

The following example returns all ancestors of all <span> elements that are <ul> elements:

### Example

$(document).ready(function(){  
$("span").parents("ul");  
});

## jQuery parentsUntil() Method

The parentsUntil() method returns all ancestor elements between two given arguments.

The following example returns all ancestor elements between a <span> and a <div> element:

### Example

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

# jQuery Traversing - Descendants

A descendant is a child, grandchild, great-grandchild, and so on.

With jQuery we can traverse down the DOM tree to find descendants of an element.

## Traversing Down the DOM Tree

Two useful jQuery methods for traversing down the DOM tree are:

* children()
* find()

## jQuery children() Method

The children() method returns all direct children of the selected element.

This method only traverse a single level down the DOM tree.

The following example returns all elements that are direct children of each <div> elements:

### Example

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

we can also use an optional parameter to filter the search for children.

The following example returns all <p> elements with the class name "first", that are direct children of <div>:

### Example

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

## jQuery find() Method

The find() method returns descendant elements of the selected element, all the way down to the last descendant.

The following example returns all <span> elements that are descendants of <div>:

### Example

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

The following example returns all descendants of <div>:

**Example**

$(document).ready(function(){  
$("div").find("\*");  
});

# jQuery Traversing - Siblings

Siblings share the same parent.

With jQuery we can traverse sideways in the DOM tree to find siblings of an element.

## Traversing Sideways in The DOM Tree

There are many useful jQuery methods for traversing sideways in the DOM tree:

* siblings()
* next()
* nextAll()
* nextUntil()
* prev()
* prevAll()
* prevUntil()

## jQuery siblings() Method

The siblings() method returns all sibling elements of the selected element.

The following example returns all sibling elements of <h2>:

### Example

$(document).ready(function(){  
$("h2").siblings();  
});

we can also use an optional parameter to filter the search for siblings.

The following example returns all sibling elements of <h2> that are <p> elements:

### Example

$(document).ready(function(){  
$("h2").siblings("p");  
});

## jQuery next() Method

The next() method returns the next sibling element of the selected element.

The following example returns the next sibling of <h2>:

### Example

$(document).ready(function(){  
$("h2").next();  
});

## jQuery nextAll() Method

The nextAll() method returns all next sibling elements of the selected element.

The following example returns all next sibling elements of <h2>:

### Example

$(document).ready(function(){  
$("h2").nextAll();  
});

## jQuery nextUntil() Method

The nextUntil() method returns all next sibling elements between two given arguments.

The following example returns all sibling elements between a <h2> and a <h6> element:

### Example

$(document).ready(function(){  
$("h2").nextUntil("h6");  
});

## jQuery prev(), prevAll() & prevUntil() Methods

The prev(), prevAll() and prevUntil() methods work just like the methods above but with reverse functionality: they return previous sibling elements (traverse backwards along sibling elements in the DOM tree, instead of forward).

# jQuery UI Introduction

**jQueryUI** stands for jQuery User Interface. It is a collection of animated visual effects, GUI widgets, and themes implemented with jQuery, CSS, HTML and JavaScript. These new plug-ins add a lot of new functionalities in the jQuery core library.

It is a very popular and powerful mobile first front-end framework used for faster and easier web development. According to a survey it is used on over 176000 websites, making it the second most popular JavaScript library on the Web.

### jQuery UI History

jQueryUI is a free and open source software first published and released in September 2007. It is distributed by jQuery foundation under the MIT license.

# jQuery UI Features

* jQueryUI facilitates to make highly interactive web applications.
* It is open source and free to use.
* It provides a powerful theme mechanism.
* It is very stable and maintenance friendly.
* It provides an extensive browser support
* Good documentation

## jQuery UI Categorization

We can categorize the jQueryUI into four groups.

1. Interactions
2. Widgets
3. Effects
4. Utilities

**1) Interactions:** Interactions are the set of plug-ins which facilitates users to interact with DOM elements. These are the mostly used interactions:

* Draggable
* Droppable
* Resizable
* Selectable
* Sortable

**2) Widgets:** Widgets are the jQuery plug-ins which makes you able to create user interface elements like date picker, progress bar etc. These are the mostly used widgets:

* Accordion
* Autocomplete
* Dialog
* Button
* Date Picker
* Menu
* Progress Bar
* Tabs
* Tooltip
* Slider
* Spinner

**3) Effects:**The internal jQuery effects contain a full suite of custom animation and transition for DOM elements.

* Hide
* Show
* Add Class
* Remove Class
* Switch Class
* Toggle Class
* Color Animation
* Effect
* Toggle

**4) Utilities:**Utilities are the modular tools, used by jQuery library internally.

* **Position:**It is used to set the position of the element according to the other element's alignment (position).

# jQuery UI Draggable

jQuery UI draggable() method is used to make any DOM element draggable. Once the element is made draggable, We can move it by clicking on it with the mouse and drag it anywhere within the viewport.

**Syntax:**

We can use the draggable () method in two forms:

* $(selector, context).draggable (options) Method
* $(selector, context).draggable ("action", params) Method

## First Method

The draggable (option) method specifies that an HTML element can be moved in the HTML page. Here, the option parameter specifies the behavior of the elements involved.

**Syntax:**

$(selector, context).draggable(options);

**Following is a list of different options that can be used with this method:**

|  |  |
| --- | --- |
| **Option** | **Description** |
| addclasses | If this option is set to false, it will prevent the UI-draggable class from being added in the list of selected DOM elements. By default its value is true. |
| appendto | It specifies the element in which the draggable helper should be appended to while dragging. By default its value is "parent". |
| axis | This option constrains dragging to either the horizontal (x) or vertical (y) axis. Its possible values are:"x", "y". |
| cancel | This option is used to prevent dragging from starting on specified elements. By default its value is "input, textarea, button, select, option". |
| connecttosortable | This option is used to specify a list whose elements are interchangeable. At the end of placement, the element is part of the list. By default its value is "false". |
| containment | Constrains dragging to within the bounds of the specified element or region. By default its value is "false". |
| cursor | It is used to specify the CSS property of the cursor when the element moves. It represents the shape of the mouse pointer. By default its value is "auto". |
| cursorat | It sets the offset of the dragging helper relative to the mouse cursor. Coordinates can be given as a hash using a combination of one or two keys: { top, left, right, bottom }. By default its value is "false". |
| delay | It specifies the delay in milliseconds, after which the first movement of the mouse is taken into account. The displacement may begin after that time. By default its value is "0". |
| disabled | It disables the ability to move items when set to true. Items cannot be moved until this function is enabled (using the draggable ("enable") instruction). By default its value is "false". |
| distance | The number of pixels that the mouse must be moved before the displacement is taken into account. By default its value is "1". |
| grid | It snaps the dragging helper to a grid, every x and y pixels. The array must be of the form [ x, y ]. By default its value is "false". |
| handle | If specified, restricts dragging from starting unless the mousedown occurs on the specified element(s). By default its value is "false". |
| helper | It allows for a helper element to be used for dragging display. By default its value is "original". |
| iframefix | It prevents iframes from capturing the mousemove events during a drag. By default its value is "false". |
| opacity | Opacity of the element moved when moving. By default its value is "false". |
| refreshpositions | If set to true, all droppable positions are calculated on every mousemove. By default its value is "false". |
| revert | It indicates whether the element is moved back to its original position at the end of the move. By default its value is "false". |
| revertduration | It indicates the duration of displacement (in milliseconds) after which the element returns to its original position (see options.revert). By default its value is "500". |
| scope | It is used to group sets of draggable and droppable items, in addition to droppable's accept option. By default its value is "default". |
| scroll | when set to true (the default), the display will scroll if the item is moved outside the viewable area of the window. by default its value is "true". |
| scrollsenstivity | It indicates how many pixels the mouse must exit the window to cause scrolling of the display. By default its value is "20". |
| scrollspeed | It indicates the scrolling speed of the display once scrolling begins. By default its value is "20". |
| snap | It adjusts the display of the item being moved on other elements (which are flown). By default its value is "false". |
| snapmode | It specifies how the adjustment should be made between the moved element and those indicated in options.snap. By default its value is "both". |
| snaptolerance | It specifies the maximum number of pixels in the difference in position necessary to establish the adjustment. By default its value is "20". |
| stack | It controls the z-index of the set of elements that match the selector, always brings the currently dragged item to the front. It is very useful in things like window managers. By default its value is "false". |
| zindex | z-index for the helper while being dragged. By default its value is "false". |

## Second Method

The draggable (action, params) method is used to perform an action like prevent displacement. Here action is specified as a string and one or more params can be provided based on the given action.

**Syntax:**

$(selector, context).draggable ("action", [params]);

**Following is a list of actions used with this method:**

|  |  |
| --- | --- |
| **Action** | **Description** |
| destroy() | It is used to remove drag functionality completely. The elements are no longer movable. This will return the element back to its pre-init state. |
| disable() | It is used to disable drag functionality. Elements cannot be moved until the next call to the draggable("enable") method. |
| enable() | It is used to reactivates drag management. The elements can be moved again. |
| option(optionname) | It gets the value currently associated with the specified optionname. Here optionname is name of the option to get and is of type string. |
| option() | It gets an object containing key/value pairs representing the current draggable options hash. |
| option(optionname, value) | It sets the value of the draggable option associated with the specified optionname. Here optionname is the name of the option to set and value is the value to set for the option. |
| option(options) | It sets one or more options for the draggable. Here options is a map of option-value pairs to set. |
| widget() | It returns a jQuery object containing the draggable element. |