Using jQuery

jQuery can be used just like any other JavaScript library i.e., by adding it using the HTML script tag prior to writing code that requires jQuery. For example:

```
<script src="jsquery.js">
<script>
// Enter code leveraging jQuery here.
</script>
```

CORE

jQuery Object

- **jQuery()** - Returns a collection of matched elements either created by passing an HTML string or found in the DOM based on passed argument(s). It has the following variations:
- **jQuery(selector [, context])/jQuery(element)/jQuery(elementArray)/jQuery(object)/jQuery(selection)/jQuery()**
- **jQuery(html [,ownerDocument])/jQuery(html, attributes)**
- **jQuery(callback)**
- **jQuery.noConflict([removeAll])** - Returns jQuery's control of the $ variable back to some other JS library. For example:

```
<script src="someJSlibrary.js"></script>
<script src="jquery.js"></script>
<script>
jQuery.noConflict();
// Enter code leveraging $ of someJSlibrary.
</script>
```

- **jQuery.holdReady(hold)** - Allows the caller to delay jQuery's ready event. Typically used by dynamic script loaders for loading additional JS, such as jQuery plugins, prior to the occurrence of the ready event.
- **jQuery.when(deferred)** - Offers a way to execute callback functions based on none, one or more Tenable objects. Returns a resolved Promise when no arguments are passed.

Deferred Object

- **jQuery.Deferred([beforeStart])** - Returns/Creates a chainable utility object (deferred object) in the pending state. It is a factory function.

- **deferred.always(alwaysCallbacks [, alwaysCallbacks])** - Calls handlers when the Deferred object is resolved or rejected. Other methods of the Deferred object can be chained to this method. Callbacks are executed in the same order as they were added.
- **deferred.done(done callback [, doneCallbacks])** - Calls handlers when the Deferred object is resolved. Accepts one or many arguments. These can either be a single function or an array of functions.
- **deferred.fail(failCallbacks [, failCallbacks])** - Calls handlers when the Deferred object is rejected. Accepts one or many arguments, which can be either a single function or an array of functions. Executes callbacks in the same order as they are added.
- **deferred.notify(args)** - Calls the progressCallbacks added by deferred.then or deferred.progress on a Deferred object with the given arguments. The arguments are passed to each progressCallback.
- **deferred.notifyWith(context [, args])** - Calls the progressCallbacks on a Deferred object with the given arguments and context.
- **[Deprecated in jQuery 1.8] deferred.pipe([doneFilter][, failFilter])** - Utility method for filtering and/or chaining Deferred objects. Returns a new promise that filters the status and values of a Deferred object through a function.
- **deferred.progress(progressCallbacks [, progressCallbacks])** - Calls handlers when the Deferred object generates progress notifications. Other methods of the Deferred object can be chained to this method.
- **deferred.promise([target])** - Allows an asynchronous function to prevent other code from interfering with the progress or status of its internal request. When a target is provided, it attaches the methods onto it and then returns this object rather than creating a new one.
- **deferred.reject([args])** - Rejects a Deferred object and calls any failCallbacks with the given arguments.
- **deferred.rejectWith([, args])** - Rejects a Deferred object and calls any failCallbacks with the given context and arguments.
- **deferred.resolve([args])** - Resolves a Deferred object and calls any doneCallbacks with the given arguments.
- **deferred.resolveWith(context[, args])** - Resolves a Deferred object and calls any doneCallbacks with the given arguments and context.
- **deferred.state()** - Returns a string that represents the present state of a Deferred object. In other words, it determines the current state of a Deferred object. A Deferred object can be in one of the following three states:
- **Pending -** Not in a completed state.
- **Resolved -** Either deferred.resolve() or deferred.resolveWith() has been called, and the doneCallbacks have been called or are about to be called.
- **Rejected -** Either deferred.reject() or deferred.rejectWith() has been called, and the failCallbacks have been called or are about to be called.
- **deferred.then(doneFilter[, failFilter][, progressFilter])** - Calls handlers when the Deferred object is resolved, rejected, or pending. Returns a new promise that can filter the status and values of the Deferred object through a function. Allows chaining other methods of the Promise object to it.

- **.promise([type][, target])** - Returns a dynamically generated Promise object that is resolved when all actions of a certain type bound to the collection, queue or not, have ended.

Utilities

- **jQuery.contains(container, contained)** - Checks whether a DOM element is a descendant of some other DOM element or not. Supports only element nodes. The first argument can't be a jQuery object or plain JS object, it must be a DOM element. For example:

```
$.contains(document.documentElement, document.Body); // Returns true
$.contains(document.Body, document.documentElement); // Returns false
```

- **jQuery.each(array/object, callback)** - Seamlessly iterates over both arrays and objects. While arrays and array-like objects with a length property are iterated by numeric index, beginning from 0 and ending at length - 1, other objects are iterated via their named properties. Returns the first argument, i.e., the object that is iterated.
- **jQuery.extend(target, [, object1], [, object2],…., [, objectN])** - Merge the contents of two or more objects together into the target, first object. Ignores null or undefined arguments. If only one argument is supplied, i.e., the target argument is omitted, the jQuery object is assumed to be the target object. This helps in adding new functions to the jQuery namespace.
- **jQuery.globalEval(code[, options])** - Executes the mentioned JS code within the global context. Typically used for loading external scripts dynamically.
- **jQuery.grep(array, function, [, invert])** - Finds members of an array that satisfy a filter function. The original array remains unaffected. Two arguments are passed to the filter function, the current array item and its index.
- **jQuery.inArray(value, array, [, fromIndex])** - Searches and returns the index of a specified value in an array. Returns -1 if not found.
- **jQuery.isArray(object)** - Checks/Tests whether the object is an array.
- **jQuery.isEmptyObject(object)** - Checks/Tests whether the object is empty i.e. contains no enumerable properties. For example:

```
jQuery.isEmptyObject({}); // Returns true
jQuery.isEmptyObject({ not: "empty" }); // Returns false
```

- **[Deprecated in jQuery 3.3] jQuery.isFunction(value)** - Returns true if the argument passed is a callable function. False otherwise.
- **jQuery.isNumeric(value)** - Returns true if the argument passed is of type number or string. False otherwise.
- **jQuery.isPlainObject(object)** - Checks whether the provided object is a plain object or not.
- **[Deprecated in jQuery 3.3] jQuery.isWindow(object)** - Returns true if the passed argument is a window.

- **jQuery.isXMLDoc(node)** - Checks whether a DOM node is an XML document or within an XML document.
- **jQuery.makeArray(object)** - Converts an array-like object into a true JS array. Post conversion, any special features the object had will no longer be available.
- **jQuery.map(array/object, callback)** - Applies a function to each element of an array or object and maps the processed elements in a new array.
- **jQuery.merge(first, second)** - Returns an array containing all elements from the two arrays. The order of the elements is preserved and items from the second array are appended to the first array. For example,

jQuery.merge ([0, 1, 2 , 3], [4, 5, 6, 7]) // Returns [0, 1, 2, 3, 4, 5, 6, 7]

- **jQuery.noop()** - An empty function that is used when there is a need to pass a function that does nothing. Useful for jQuery plugin authors for offering optional callbacks. The function executes when no callback is given.
- **jQuery.now()** - Returns a number that represents the current time. Similar to the (new Date).getTime() expression.
- **jQuery.parseHTML(data[, context][, keepScripts])** - Uses native methods to convert (parse) a given string into an array of DOM nodes.
- **[Deprecated in jQuery 3.0] jQuery.parseJSON()** - Returns an equivalent JavaScript value from a well-formed JSON string. A JS exception is thrown if a malformed JSON string is passed.
- **jQuery.parseXML(data)** - Parses/Converts a string into an XML document using the native parsing function of the browser. The created XML document can then be passed to jQuery for creating a typical jQuery object, which can be manipulated and traversed.
- **jQuery.proxy(function, context)** - Returns a function that will always have a particular context.. Useful for attaching event handlers to an element where the context points back to a different object.
- **[Deprecated in jQuery 1.9] jQuery.support** - A set of properties representing the presence of different browser features or bugs. Intended for internal use in jQuery.
- **jQuery.trim(string)** - Removes the whitespace (newlines, spaces, and tabs) from the beginning and end of the supplied string. Whitespace characters present in other than the start and end of the string are preserved.
- **jQuery.type(object)** - Determines the internal JavaScript [[Class]] of the passed object.
- **[Deprecated in jQuery 3.0] jQuery.unique(array)** - Searches through an array of DOM elements, performs sorting, and removes duplicate elements. Unusable for arrays of numbers or strings.
- **jQuery.uniqueSort(array)** - Replacement for jQuery.unique() method from jQuery 3.0.

DOM Element Methods

- **.get(index)** - Returns an element/DOM node matched by the index. Returns undefined if the value of the index is out of bounds. If no index is supplied, the method returns an array of all the elements.

- **.index(element/selector)** - Takes a DOM node and returns an index. Returns an integer value depending on three different cases:
- **When no argument is passed** - Returns an integer representing the position of the first element within the jQuery object w.r.t. its sibling elements.
- **When the method is called on a collection of elements, and a DOM element or jQuery object is passed** - Returns an integer representing the position of the passed element w.r.t to the original collection.
- **When a selector string is passed** - Returns an integer value indicating the position of the first element within the jQuery object w.r.t. The elements matched by the selector string. Returns -1 when an element is not found.
- **.toArray()** - Returns an array of all the elements in the jQuery set.

Internals

- **.jquery** - A string containing the version number of jQuery. The property is assigned to the jQuery prototype i.e. $.fn. For example,

alert ("The jQuery version you're using is:" + $.fn.jquery); // Returns the version number of jQuery you're running.

- **jQuery.error(message)** - Accepts a string and throws an exception containing the same. They are primarily used by plugin developers for overriding the method for providing a better/more informative display for the error messages.
- **.length** - Returns an integer value representing the number of elements in the jQuery object. Similar to the .size() method.
- **.pushStack(elements)/.pushStack(elements, name, arguments)** - Adds an array of DOM elements onto the jQuery stack.

Callbacks Object

- **jQuery.Callbacks(flags)** - Used internally for providing the base functionality to the jQuery $.ajax() and $.Deferred() components. It is a multi-purpose method offering a powerful way of managing callback lists. Provides support for several methods, such as callbacks.add(), callbacks.disable(), and callbacks.remove().
  The optional flags argument determines the behaviour of the returned callback list. Supported flags are:
- **memory** - Keeps track of the previous values and calls any callback added after the list has been fired straight away with the latest memorized values.
- **once** - Ensures that the callback list is fired only once.
- **stopOnFalse** - Interrupts callings upon coming across a callback returns false.
- **unique** - Ensures that a callback is added only once.
- **callbacks.add(callbacks)** - Adds single or multiple callbacks to a callback list.
- **callbacks.disable()** - Disables a callback list from performing further.
- **callbacks.disabled()** - Checks whether the callback list is disabled.
- **callbacks.empty()** - Removes all the callbacks from a callback list.

- **callbacks.fire(arguments)** - Calls/Invokes all the callbacks from a callback list with the passed arguments.
- **callbacks.fired()** - Checks whether the callbacks from a callback list have been called at least once.
- **callbacks.fireWith([context][, arguments])** - Fires all callbacks from a callback list with the passed arguments and context.
- **callbacks.has([callback])** - Checks if the callback list has any callbacks attached. If a callback is passed as an argument, then determines whether it is in the callback list or not.
- **callbacks.lock()** - Locks a callback list in its current state. Additional functions can be added and fired after the callback list is locked, provided the Callbacks object is created with the memory flag as its argument.
- **callbacks.locked()** - Determines the lock-state of a callback list.
- **callbacks.remove(callbacks)** - Removes one or more or all callbacks from a callback list.

EFFECTS

It is possible to globally turn off all jQuery effects by setting:

jQuery.fx.off = true

Basics

- **.hide()/.hide([duration][, complete])/.hide(options)/.hide(duration[, easing][, complete])** - Hides the matching element(s).
- **.show()/.show([duration][, complete])/.show(options)/.show(duration[, easing][, complete])** - Displays the matching element(s).
- **.toggle()/.toggle([duration][complete])/.toggle(options)/.toggle(duration[, easing][, complete])/.toggle(display)** - Toggles the visibility of matching elements.

Custom

- **.animate(properties[, duration][, easing][, complete])/.animate(properties, options)** - Creates animation effects on any numeric CSS property.
- **.clearQueue(queueName)** - Removes all functions pending execution from the passed queue. Removes remaining functions from the fx, the standard effects queue, when called without specifying a queue, i.e., no argument specified.
- **.delay(duration[, queueName])** - Delays, with a timer, the pending functions in the specified queue.
- **.dequeue(queueName)** - Removes the next function from the specified queue and then executes the same.
- **jQuery.dequeue(element[, queueName])** - Same as that of .dequeue().
  Note: Low-level method. Using .dequeue() preferred.

- **.finish(queue)** - Stops the ongoing (currently-running) animation, removes all queued animations, and completes all animations by setting their CSS properties to their target values. If a string is passed, then only the animations represented by the same are stopped from the queue.
  Note: The .finish() method is the same as that of the .stop(true, true) with the exception that the former also results in setting CSS properties of all the queued animations to their end values.
- **[Deprecated in jQuery 3.0] jQuery.fx.interval** - A property for adjusting the rate, in milliseconds, at which animations are fired. The default is 13 ms.
  Note: No effect in browser supporting the requestAnimationFrame method.
- **jQuery.fx.off** - When set to true, this property disables all animations, i.e., instead of displaying an effect, all animation methods set their elements to their final state. Set the property false to turn on animations.
- **jQuery.speed([duration][, settings])/jQuery.speed([duration][, easing][, complete])/jQuery.speed(settings)** - Allows defining properties usable in a custom animation by creating an object containing the same, instantly usable in the definition of custom animations. An alternative to implementing logic dealing with default values and optional parameters for defining animation effects.
- **.queue([queueName])** - DIsplays the queue of functions pending execution on the matched elements.
- **.queue([queueName], newQueue)/.queue([queueName], callback)** - Manipulates the queue of functions pending execution, once for every matched element.
- **jQuery.queue(element[, queueName])** - Same as that of .queue([queueName]) method.
  Note: Low-level method. .queue([queueName]) recommended.
- **jQuery.queue(element, queueName, newQueue)/jQuery.queue(element, queueName, callback)** - Same as that of .queue() method for manipulating.
  Note: Low-level method. .queue() recommended.
- **.stop([clearQueue][, jumpToEnd])/.stop([queue][, clearQueue][, jumpToEnd])** - Stops ongoing animation for the matched elements.

Fading

- **.fadeIn([duration][, complete])/.fadeIn(options)/.fadeIn([duration][, easing][, complete])** - Displays matched element(s) by fading them completely opaque.
- **.fadeOut([duration][, complete])/.fadeOut(options)/.fadeOut([duration][, easing][, complete])** - Hides matched element(s) by fading them to transparent.
- **.fadeTo(duration, opacity [, complete])/.fadeTo(duration, opacity [, easing][, complete])** - Adjusts opacity of matched element(s).
- **.fadeToggle([duration][, easing][, complete])/.fadeToggle(options)** - Displays or hides matched elements by animating their opacity.

Sliding

- **.slideDown([duration][, complete])/.slideDown(options)/.slideDown([duration][, easing][, complete])** - Displays the matched elements with a sliding motion.
- **.slideToggle([duration][, complete])/.slideToggle(options)/.slideToggle([duration][, easing][, complete])** - Displays or hides the matched elements with a sliding motion.
- **.slideUp([duration][, complete])/.slideUp(options)/.slideUp([duration][, easing][, complete])** - Hides the matched elements with a sliding motion.

EVENTS

Browser Events

- **.resize(handler)/.resize([eventData], handler)/.resize()** - Triggers the resize JS event on an element. Can also be used for binding an event handler to the resize event.
- **.scroll(handler)/.scroll([eventData], handler)/.scroll()** - Triggers the scroll JS event on an element. Also used for binding an event handler to the scroll event.

Document Loading

- **.ready(handler)** - Specifies a function to be executed once the DOM is fully loaded. It offers a way to run the JS code as soon as the web page's DOM becomes safe to manipulate.

Event Handler Attachment

- **[Deprecated in jQuery 3.0] .bind(eventType [, eventData], handler)/.bind(eventType [, eventData] [,preventBubble])/.bind(events)** - Attaches a handler to one or more events for the elements.
- **[Deprecated in jQuery 3.0] .delegate(selector, eventType, handler)/.delegat(selector, eventType, eventData, handler)/.delegate(selector, events)** - Attaches a handler to a single or several events for all matching elements, instantly or some time later on the basis of a specific set of root elements.
- **.off()/.off(event)/.off(events [, selector])/.off(events [, selector][handler])** - Removes specified event handler(s) attached with .on(). Removes all handlers attached to the elements when no arguments are specified.
- **.on(events [, selector][, data], handler)/.on(events [, selector][, data])** - Attaches one or more event handlers for a single or several events to the selected elements.
- **.one(events [, data], handler)/.on(events [, selector][, data], handler)/.one(events [, selector][, data])** - Almost identical to .on() method with the exception that the handler for an element and event type is unbound after the first invocation.
- **.trigger(eventType [, extraParameters])/.trigger(event [, extraParameters])** - Executes all event handlers and behaviors attached to matched elements for an event. The order of the execution of event handlers is retained, i.e., the event handlers are executed in the same order as they were to be executed if triggered naturally by the user.

- **.triggerHandler(eventType [, extraParameters])/.triggerHandler(event [, extraParameters])** - Executes all event handlers attached to the matching element for an event.
- **[Deprecated in jQuery 3.0] .unbind()/.unbind(event)/.unbind(eventType, false)/.unbind(eventType, [, handler])** - Removes a event handler(s) from the matched elements.
- **[Deprecated in jQuery 3.0] .undelegate()/.undelegate(namespace)/.undelegate(selector, eventType)/.undelegate(selector, eventType, handler)/.undelegate(selector, events)** - Removes event handlers bound using .delegate().

Form Events

- **.blur()/.blur(handler)/.blur([eventData], handler)** - Binds or triggers an event handler to the blur JS event on an element.
- **.change()/.change(handler)/.change([eventData], handler)** - Binds or triggers an event handler to the change JS event on an element.
- **.focus()/.focus(handler)/.focus([eventData], handler)** - Binds or triggers an event handler to the focus JS event on an element.
- **.focusin()/.focusin(handler)/.focusin([eventData], handler)** - Binds or triggers an event handler to the focusin JS event on an element.
- **.focusout()/.focusout(handler)/.focusout([eventData], handler)** - Binds or triggers an event handler to the focusout JS event on an element.
- **.select()/.select(handler)/.select([eventData], handler)** - Binds or triggers an event handler to the select JS event on an element.
- **.submit()/.submit(handler)/.submit([eventData], handler)** - Binds or triggers an event to the submit JS event on an element.

Keyboard Events

- **.keydown()/.keydown(handler)/.keydown([eventData], handler)** - Binds or triggers an event handler to the keydown JS event on an element.
- **.keypress()/.keypress(handler)/.keypress([eventData], handler)** - Binds or triggers an event handler to the keypress JS event on an element.
- **.keyup()/.keyup(handler)/.keyup([eventData], handler)** - Binds or triggers an event handler to the keyup JS event on an element.

Mouse Events

- **.click()/.click(handler)/.click([eventData], handler)** - Binds or triggers an event handler to the click JS event on an element.
- **.contextmenu()/.contextmenu(handler)/.contextmenu([eventData], handler)** - Binds or triggers an event handler to the contextmenu JS event on an element.
- **.dblclick()/.dblclick(handler)/.dblclick([eventData], handler)** - Binds or triggers an event handler to the dblclick JS event on an element.

- **.hover(handlerIn, handlerOut)** - Binds two event handlers to the matched elements that execute when the mouse pointer enters and leaves the elements. For binding one event handler, use .hover(handlerInOut).
- **.mousedown()/.mousedown(handler)/.mousedown([eventData], handler)** - Binds or triggers an event handler to the mousedown JS event on an element.
- **.mouseenter()/.mouseenter(handler)/.mouseenter([eventData], handler)** - Binds or triggers an event handler when the mouse enters an element.
- **.mouseleave()/.mouseleave(handler)/.mouseleave([eventData], handler)** - Binds or triggers an event handler when the mouse leaves an element.
- **.mousemove()/.mousemove(handler)/.mousemove([eventData], handler)** - Binds or triggers an event handler to the mousemove JS event on an element.
- **.mouseout()/.mouseout(handler)/.mouseout([eventData], handler)** - Binds or triggers an event handler to the mouseout JS event on an element.
- **.mouseover()/.mouseover(handler)/.mouseover([eventData], handler)** - Binds or triggers an event handler to the mouseover JS event on an element.
- **.mouseup()/.mouseup(handler)./mouseup([eventData], handler)** - Binds or triggers an event handler to the mouseup JS event on an element.

Event Object

- **event.currentTarget** - The present DOM element within the event bubbling phase.
- **event.delegateTarget** - The element where the presently-called jQuery event handler was attached.
- **event.data** - An optional object of data passed to an event method when the present executing handler is bound.
- **event.isDefaultPrevented()** - Returns a Boolean value representing whether the event.preventDefault() method was called or not.
- **event.isImmediatePropagationStopped()** - Checks whether the event.stopImmediatePropagation() method was called or not.
- **event.isPropagationStopped()** - Checks whether the event.stopPropagation() method was called or not.
- **event.metakey** - Checks whether the META key was fired during the time the event was fired.
  Note: The META key for Windows-based keyboards is the Windows key and Command key for Mac-based keyboards.

- **event.namespace** - Determines the event namespace used when the specified event was triggered. Used primarily by jQuery plugin authors requiring handling tasks differently on the basis of the event namespace used.
- **event.pageX** - Displays the mouse position relative to the left edge of the document.
- **event.pageY** - Displays the mouse position relative to the top edge of the document.
- **event.preventDefault()** - Prevents the default action of the specified event.
- **event.relatedTarget** - Indicates:
  -> The element being entered for mouseout
  -> The element exiting for mouseover.

- **event.result** - Returns the last value returned by an event handler that was triggered by the specified event.
- **event.stopImmediatePropagation()** - Prevents execution of the rest of the event handlers and prevents the specified event from bubbling up the DOM tree.
- **event.stopPropagation()** - Prevents the specified event from bubbling up the DOM tree
- **event.target** - Returns the DOM element that initiated the specified event.
- **event.timeStamp** - Returns a number that represents the difference (in milliseconds) between the time the browser created the specified event and January 1, 1970.
- **event.type** - Describes the nature of the specified event.
- **event.which** - Indicates the key or button that was pressed for key or mouse events, respectively.

SELECTORS

Basics

- **jQuery("*")** - Selects all elements. Known as the All or Universal Selector.
  Note: Extremely slow, unless used by itself.
- **jQuery(".class")** - Selects all the elements of the specified class. Known as the Class Selector.
- **jQuery("element")** - Selects all the elements with the specified tag name. Known as the Element Selector.
- **jQuery("#id")** - Selects an element with the passed id attribute. Known as the ID Selector.
- **jQuery("selector1, selector2,...,selectorN)** - Selects the combined results of all the specified selectors. Known as the Multiple Selector.

Hierarchy

- **jQuery("parent>child")** - Selects all the direct child elements specified by “child” of elements specified by “parent”. Known as the Child Selector.
- **jQuery("ancestor-descendant")** - Selects all the descendants of the specified ancestor. Known as the Descendant Selector.
- **jQuery("prev + next")** - Selects all the next elements matching the next selector that are immediately preceded by a sibling prev selector. Known as the Next Adjacent Selector.
  Note: Elements on either side of the combinator must have the same parent.

- **jQuery("prev ~ siblings")** - Selects all sibling elements that follow after the prev selector, have the same parent, and match the filtering sibling's selector—known as the Next Siblings Selector.
  Note: Elements on either side of the combinator must have the same parent.

Basic Filters

- **jQuery(": animated")** - Selects all the in-progress elements of animation during the time the selector is run, known as the Animated Selector.
  Note: The filter throws an error when used without the effects module.

- **[Deprecated in jQuery 3.4] jQuery(":eq(index)")/jQuery(":eq(-index)")** - Selects the element at specified index within the matching set.
- **[Deprecated in jQuery 3.4] jQuery(":even")** - Selects even index elements.
- **[Deprecated in jQuery 3.4] jQuery(":first")** - Selects the first matched DOM element.
- **[Deprecated in jQuery 3.4] jQuery(":gt(index)")/jQuery(":gt(-index)")** - Selects all the elements at an index greater than the specified index within the matching set.
- **jQuery(":header")** - Selects all the header elements.
- **jQuery(":lang(language)")** - Selects all the elements of the specified language.
- **[Deprecated in jQuery 3.4] jQuery(":last")** - Selects the last matching element.
- **[Deprecated in jQuery 3.4] jQuery(":lt(index)")/jQuery(":lt(-index)")** - Selects all the elements at an index less than the specified index within the matching set.
- **jQuery(":not(selector)")** - Selects all the elements that don't match the specified selector.
- **[Deprecated in jQuery 3.4] jQuery(":odd")** - Selects odd index elements.
- **jQuery(":root")** - Selects the root element of the document.
- **jQuery(":target")** - Selects the target element based on the fragment identifier of the document's URI.

Content Filters

- **jQuery(":contains(text)")** - Selects all the elements containing the specified text.
- **jQuery(":empty")** - Selects all the elements having no children, including text nodes.
- **jQuery (":has(selector)")** - Selects all the elements containing at least one element matching the specified selector.
- **jQuery(": parent")** - Selects all the elements having a minimum of one child node, which can be either an element or text.

Visibility Filters

- **jQuery(":hidden")** - Selects all the hidden elements.
- **jQuery(":visible")** - Selects all the visible elements.

Attribute

- **jQuery("[attribute|=' value']")** - Selects all the elements that have the specified attribute with a value equal to the specified string or starting with the same followed by a -.
- **jQuery("[attribute*='value']")** - Selects all the elements that have the specified attribute with a value containing the specified substring.
- **jQuery("[name~='value']")** - Selects elements having the specified attribute with a value containing the specified word, delimited by spaces.

- **jQuery("[name$='value']")** - Selects elements having the specified attribute with a value ending exactly with the specified string. Makes case-sensitive comparison.
- **jQuery("[name='value']")** - Selects elements having the specified attribute with a value exactly equal to the specified value.
- **jQuery("[name!=' value']")** - Selects elements that don't have the specified attribute as well as those that have the specified attribute, not the specified value.
- **jQuery("[name^='value']")** - Selects elements having the specified attribute with a value beginning exactly with the specified string.
- **jQuery("[name]")** - Selects elements having the specified attribute, regardless of the value they are having.
- **jQuery("[attributeFilter1][attributeFilter2]...[attributeFilterN]")** - Selects elements that match all the specified attributed filters. Known as Multiple Attribute Selector.

Child Filters

- **jQuery(":first-child")** - Selects all the elements that are the first child of their parent.
- **jQuery(":first-of-type")** - Selects all the elements that are the first among all the siblings of the same element name.
- **jQuery(":last-child")** - Selects all the elements that are the last child of their parent.
- **jQuery(":last-of-type")** - Selects all the elements that are the last among all the siblings of the same element name.
- **jQuery(":nth-child(index/even/odd/equation)")** - Selects all the elements that are the nth-child of their parent.
- **jQuery(":nth-last-child(index/even/odd/equation)")** - Selects all the elements that are the nth-child of their parent, starting from the last element to the first element.
- **jQuery(":nth-of-type(index/even/odd/equation)")** -Selects all the elements that are the nth-child of their parent with respect to siblings with the same element name.
- **jQuery(":nth-last-of-type(index/even/odd/equation)")** - Selects all the elements that are the nth-child of their parent with respect to siblings with the same element name. It starts counting towards the last element to the first element.
- **jQuery(":only-child")** - Selects all the elements that are the only child of their parent element.
- **jQuery(":only-of-type")** - Selects all the elements that have no siblings with the same element name.
  Note: Matches nothing if the parent has other child elements with the same element name.

Forms

- **jQuery(":button")** - Selects all the button elements as well as elements of the type button.
- **jQuery(":checkbox")** - Selects all the elements of the type checkbox.
- **jQuery(":checked")** - Selects all the elements that are checked (or selected).
- **jQuery(":disabled")** - Selects all the elements that are disabled.

- **jQuery(":enabled")** - Selects all the elements that are enabled.
- **jQuery(":focus")** - Selects an element if it is currently focused.
- **jQuery(":file")** - Selects all the elements of the type file.
- **jQuery(":image")** - Selects all the elements of the type image.
- **jQuery(":input")** - Selects all the button, input, select, and textarea elements.
- **jQuery(":password")** - Selects all the elements of the type password.
- **jQuery(":radio")** - Selects all the elements of the type radio.
- **jQuery(":reset")** - Selects all the elements of the type reset.
- **jQuery(":selected")** - Selects all the elements that are selected.

That completes part - I of jQuery Cheat Sheet. We explained Core, Effects, Events, and Selectors here. In the second part, we'll focus on AJAX, Attribute/CSS, Manipulation, and Traversing in jQuery.