

FRONTEND ADVANCED

THE ADVANCED PROGRAMMING WORLD

JQUERY

WRITE LESS, DO MORE

WHAT IS JQUERY?

jQuery is a fast, lightweight, and feature-rich JavaScript library that is based on the principle *"write less, do more"*

Easy-to-use APIs makes the things like HTML document traversal and manipulation, event handling, adding animation effects to a web page

jQuery was originally created by John Resig in early 2006. The jQuery project is currently run and maintained by a distributed group of developers as an open-source project

jQuery also gives you the ability to create an Ajax based application in a quick and simple way

WHAT WE CAN DO WITH JQUERY?

Select elements to perform manipulation.

Create effect like show or hide elements, sliding transition, and so on

Create complex CSS animation with fewer lines of code

Manipulate DOM elements and their attributes

Implement Ajax to enable asynchronous data exchange between client and server

Traverse all around the DOM tree to locate any element

ADVANTAGES OF USING JQUERY

Save lots of time

Simplify common JavaScript tasks

Easy to use

Compatible with browsers

Absolutely Free

LETS DO OUR HANDS DIRTY

Ways of including jQuery in your Web App

DOWNLOAD A COPY OF JQUERY

- Download a copy of jQuery and include it in your document
- *Compressed or uncompressed*
- <https://jquery.com/download/>

INCLUDING JQUERY FROM CDN


- You can include jQuery in your document through freely available CDN (Content Delivery Network) links
- You can also include jQuery through Google and Microsoft CDN's

JQUERY SYNTAX


A jQuery statement typically starts with the dollar sign (\$) and ends with a semicolon (;)

The dollar sign (\$) is just an alias for jQuery

This statement is typically known as ready event.
Where the handler is basically a function that is passed to the ready() method to be executed safely as soon as the document is ready



```
$(document).ready(function(){  
    $("p").text("Hello World!");  
    alert("Hello World!");  
});
```



Inside an event handler function you can write the jQuery statements to perform any action following the basic syntax, like: \$(selector).action();

You should place the jQuery code inside the document ready event so that your code executes when the document is ready to be worked on

SELECTORS

CSS SELECTORS AND MORE

SELECTING ELEMENTS WITH JQUERY

Selecting Elements by	Description	Syntax	example
ID	ID selector to select a single element with the unique ID on the page	<code>\$("#id")</code>	<code>\$("#mark").css("background", "yellow");</code>
Class Name	Class selector can be used to select the elements with a specific class	<code>\$(".className")</code>	<code>\$(".mark").css("background", "yellow");</code>
Name	Element selector can be used to select elements based on the element name	<code>\$("elementName")</code>	<code>\$("p").css("background", "yellow");</code>
Attribute	Select an element by one of its HTML attributes, such as a link's target attribute or an input's type attribute, etc	<code>\$("[attr=value]")</code>	<code>\$('input[type="text"]').css("background", "yellow")</code>
Compound CSS Selector	Combine the CSS selectors to make your selection even more precise	<code>\$("elementSelector.classNameSelector")</code>	<code>\$("p.mark").css("background", "yellow");</code>
Custom Selector	jQuery provides its own custom selector to further enhancing the capabilities of selecting elements on a page	<code>\$("pseudoSelector")</code>	<code>\$("tr:odd").css("background", "yellow");</code>

EVENTS

MOUSE, KEYBOARD, FORM AND MORE

JQUERY EVENTS

MOUSE EVENTS

- click()
- dblclick()
- hover()
- mouseenter()
- mouseleave()

FORMS EVENTS

- change()
- focus()
- blur()
- submit()

KEYBOARD EVENTS

- keypress()
- keydown()
- keyup()

WINDOW EVENTS

- ready()
- resize()
- scroll()

EFFECTS

Let's make it effectful

EFFECTS

METHODS	DESCRIPTION
show() / hide()	Hide() simply sets the inline style display: none and show() method restores the display properties
toggle()	Show or hide the elements in such a way that if the element is initially displayed
fadeIn() / fadeOut()	Display or hide the HTML elements by gradually increasing or decreasing their opacity
fadeToggle()	Display or hide the selected elements by animating their opacity
fadeTo()	Lets you fade in the elements to a certain opacity level
slideUp() / slideDown()	Used to hide or show the HTML elements by gradually decreasing or increasing their height (i.e. by sliding them up or down)
slideToggle()	Show or hide the selected elements by animating their height in such a way that if the element is initially displayed
animate()	Used to create custom animations. Typically used to animate numeric CSS properties
stop()	Used to stop the jQuery animations or effects currently running on the selected elements before it completes

MANIPULATION

GETTERS & SETTER

Some jQuery methods can be used to either assign or read some value on a selection. A few of these methods are `text()`, `html()`, `attr()`, and `val()`

When these methods are called with no argument, it is referred to as a getters, because it gets (or reads) the value of the element

When these methods are called with a value as an argument, it's referred to as a setter because it sets (or assigns) that value

GETTERS & SETTER

text()

The jQuery `text()` method is either used to get the combined text contents of the selected elements, including their descendants, or set the text contents of the selected elements

html()

The jQuery `html()` method is used to get or set the HTML contents of the elements

attr()

You can use the jQuery `attr()` method to either get the value of an element's attribute or set one or more attributes for the selected element

val()

The jQuery `val()` method is mainly used to get or set the current value of the HTML form elements such as `<input>`, `<select>` and `<textarea>`

JQUERY INSERT CONTENT

jQuery provides several methods, like `append()`, `prepend()`, `html()`, `text()`, `before()`, `after()`, `wrap()` etc. that allows us to insert new content inside an existing element

`append()`

Used to insert content to the end of the selected elements

`prepend()`

Used to insert content to the beginning of the selected elements

`before()`

Used to insert content before the selected elements

`after()`

Used to insert content after the selected elements

`wrap()`

Used to wrap an HTML structure around the selected elements

JQUERY REMOVE ELEMENTS & ATTRIBUTE

jQuery provides handful of methods, such as `empty()`, `remove()`, `unwrap()` etc. to remove existing HTML elements or contents from the document

`empty()`

Removes all child elements as well as other descendant elements and the text content within the selected elements from the DOM

`remove()`

Removes the selected elements from the DOM as well as everything inside it. In addition to the elements themselves, all bound events and jQuery data associated with the elements are removed

`unwrap()`

Removes the parent elements of the selected elements from the DOM. This is typically the inverse of the `wrap()` method

`removeAttr()`

Removes an attribute from the selected elements

JQUERY ADD AND REMOVE CSS CLASSES

jQuery provides several methods, such as `addClass()`, `removeClass()`, `toggleClass()`, etc. to manipulate the CSS classes assigned to HTML elements

`addClass()`

adds one or more classes to the selected elements

`removeClass()`

remove the classes from the elements

`toggleClass()`

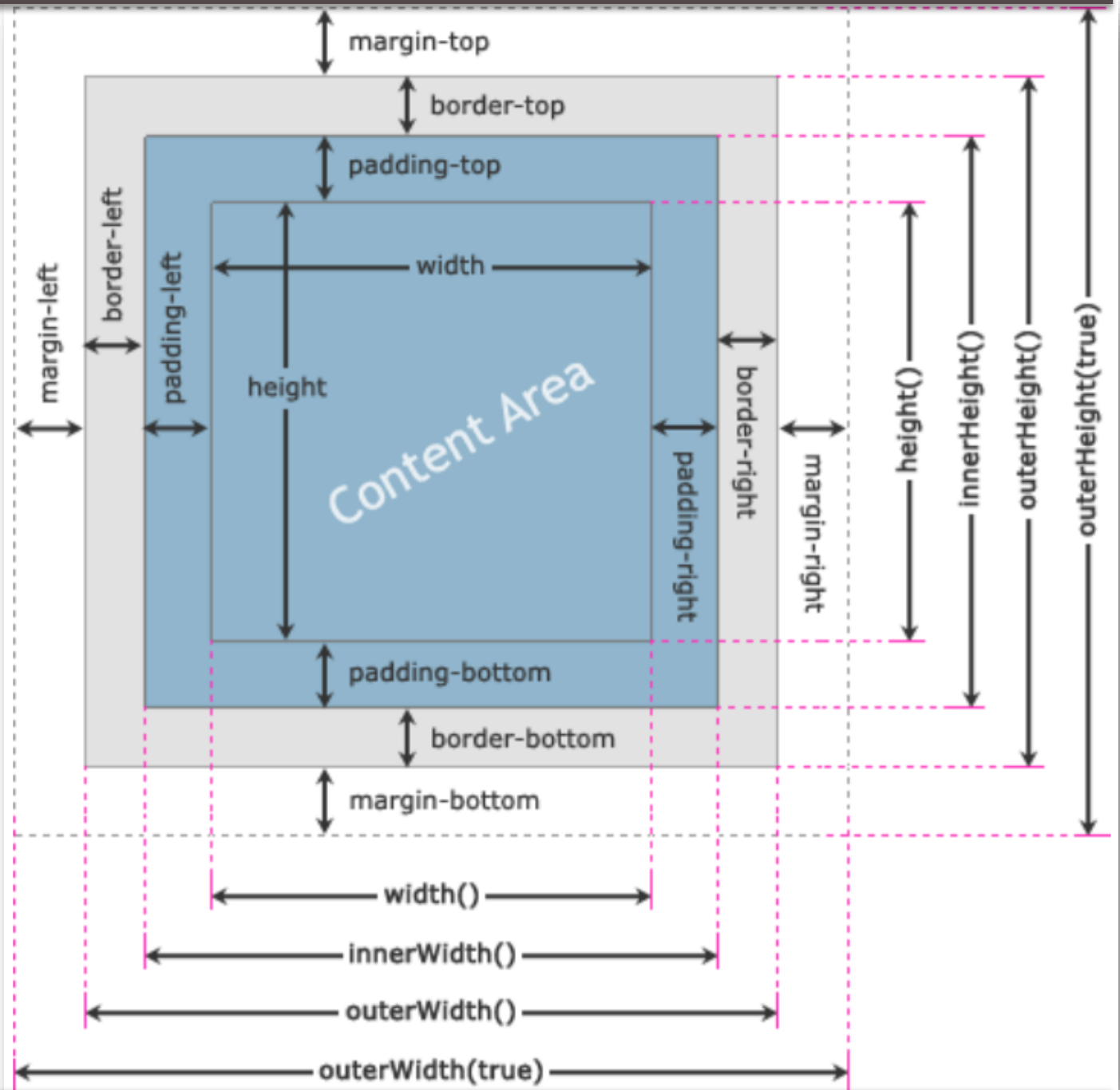
add or remove one or more classes from the selected elements by toggling them

`css()`

used to get the computed value of a CSS property or set one or more CSS properties for the selected elements

UNDERSTANDING THE JQUERY DIMENSIONS

jQuery provides several methods, such as `height()`, `innerHeight()`, `outerHeight()`, `width()`, `innerWidth()` and `outerWidth()` to get and set the CSS dimensions for the elements



TRAVERSING

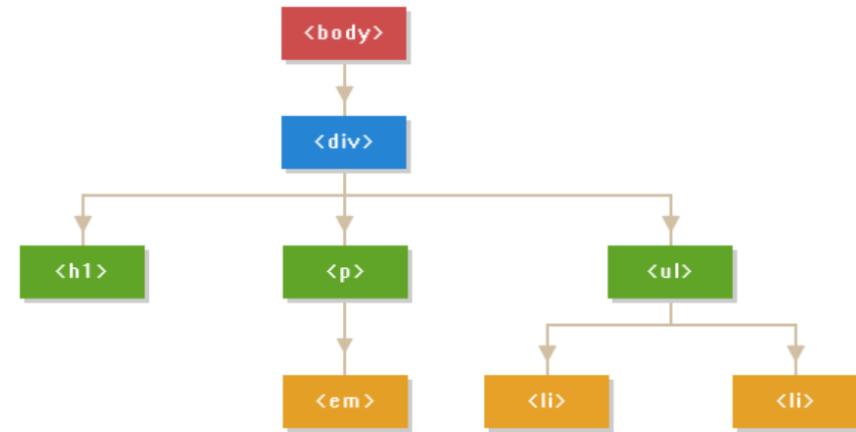
TRAVERSE THROUGH HTML DOM USING JQUERY

WHAT IS TRAVERSING?

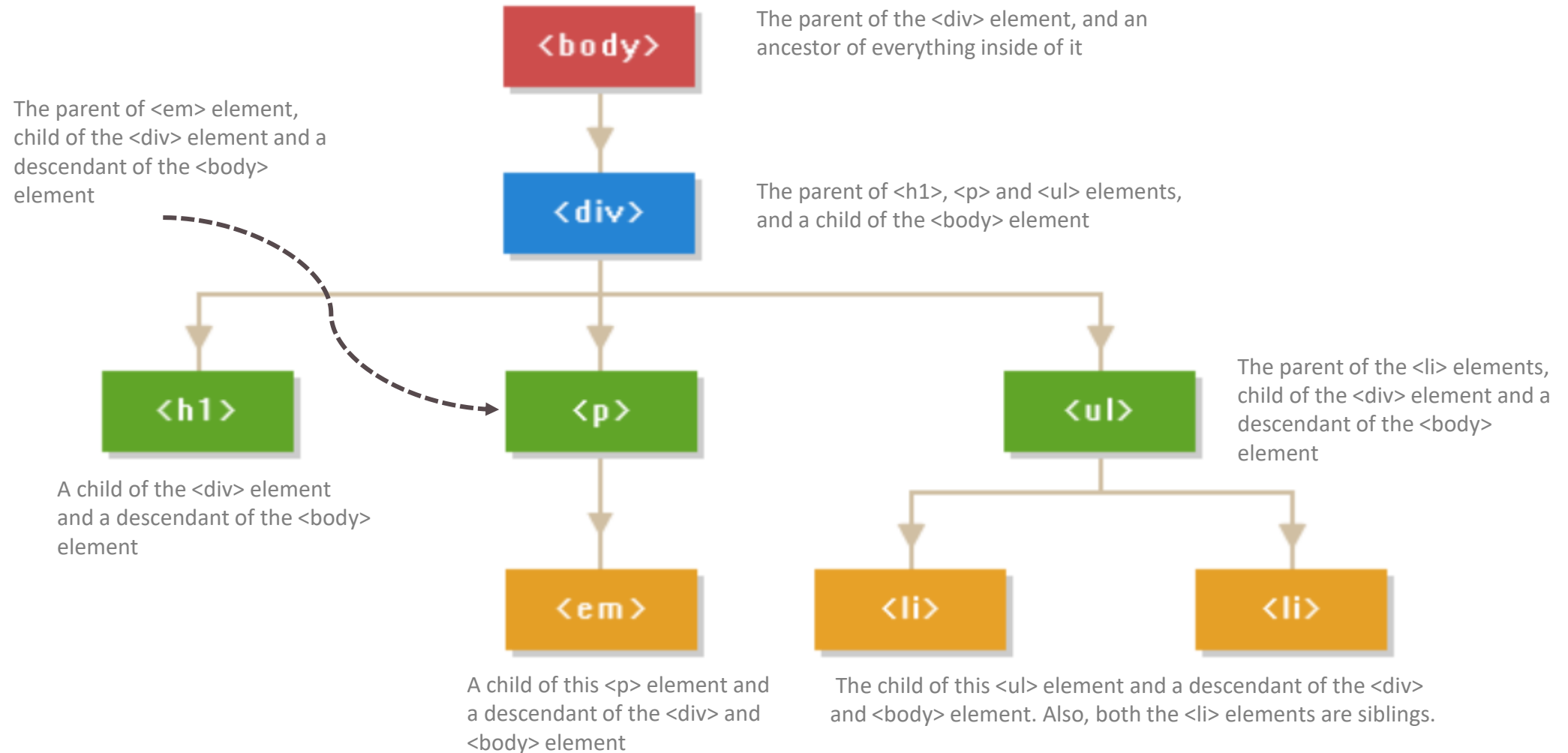
When you need to select a parent or ancestor element; that is where jQuery's DOM traversal methods come into play

With the traversal methods, we can go up, down, and all around the DOM tree very easily

To make the most of it you need to understand the relationships between the elements in a DOM tree



HOW TRAVERSING WORKS?



TRAVERSING UP THE DOM TREE

jQuery provides the useful methods such as `parent()`, `parents()` and `parentsUntil()` that you can use to traverse up in the DOM tree either single or multiple levels to easily get the parent or other ancestors of an element in the hierarchy

`parent()`

Gets the direct parent of the selected element

`parents()`

Gets the ancestors of the selected element

`parentsUntil()`

Gets all the ancestors up to but not including the element matched by the selector

TRAVERSING DOWN THE DOM TREE

jQuery provides the useful methods such as `children()` and `find()` that you can use to traverse down in the DOM tree either single or multiple levels to easily find or get the child or other descendants of an element in the hierarchy

`children()` Gets the direct children of the selected element

<code>find()</code>	Gets the descendant elements of the selected element
---------------------	--

TRAVERSING SIDEWAYS IN DOM TREE

jQuery provides several methods such as `siblings()`, `next()`, `nextAll()`, `nextUntil()`, `prev()`, `prevAll()` and `prevUntil()` that you can use to traverse sideways in the DOM tree

<code>siblings()</code>	Gets the sibling elements of the selected element
<code>next()</code>	Gets the immediately following sibling i.e. the next sibling element of the selected element
<code>nextAll()</code>	Gets all following siblings of the selected element
<code>nextUntil()</code>	Gets all the following siblings up to but not including the element matched by the selector
<code>prev()</code>	Gets the immediately preceding sibling i.e. the previous sibling element of the selected element
<code>prevAll()</code>	get all preceding siblings of the selected element

Filtering the Elements Selection

jQuery provides several methods such as `filter()`, `first()`, `last()`, `eq()`, `slice()`, `has()`, `not()`, etc. that you can use to narrow down the search for elements in a DOM tree

<code>first()</code>	Filters the set of matched elements and returns the first element from the set
<code>last()</code>	Filters the set of matched elements and returns the last element from the set
<code>eq()</code>	Filters the set of matched elements and returns only one element with a specified index number
<code>filter()</code>	Can take the selector or a function as its argument to filters the set of matched elements based on a specific criteria
<code>has()</code>	Filters the set of matched elements and returns only those elements that has the specified descendant element
<code>not()</code>	Filters the set of matched elements and returns all elements that does not met the specified conditions
<code>slice()</code>	Filters the set of matched elements specified by a range of indices

AJAX

ASYNCHRONOUS JAVASCRIPT AND XML

What is Ajax?

Ajax is just a means of loading data from the server to the web browser without reloading the whole page

AJAX make use of the JavaScript-based XMLHttpRequest object to send and receive information to and from a web server asynchronously, without interfering with the user's experience

jQuery simplifies the process of implementing Ajax by taking care of those browser differences and offers simple methods such as `load()`, `$.get()`, `$.post()`

Ajax requests are triggered by the JavaScript code; your code sends a request to a URL, and when the request completes, a callback function can be triggered to handle the response.

Since the request is asynchronous, the rest of your code continues to execute while the request is being processed

JQUERY LOAD() METHOD

```
$(selector).load(URL, data, complete);
```

The jQuery load() method loads data from the server and place the returned HTML into the selected element

The required **URL** parameter specifies the URL of the file you want to load

The optional **data** parameter specifies a set of query string (i.e. key/value pairs) that is sent to the web server along with the request

The optional **complete** parameter is basically a callback function that is executed when the request completes. The callback is fired once for each selected element

JQUERY AJAX GET AND POST REQUESTS

```
$.get(URL, data, success);  
$.post(URL, data, success);
```

The jQuery's \$.get() and \$.post() methods provide simple tools to send and retrieve data asynchronously from a web server

The required **URL** parameter specifies the URL of the file you want to load

The optional **data** parameter specifies a set of query string (i.e. key/value pairs) that is sent to the web server along with the request

The optional **complete** parameter is basically a callback function that is executed when the request completes. The callback is fired once for each selected element

REFERENCES

READING MATERIAL

- <https://api.jquery.com/>
- <https://www.tutorialrepublic.com/jquery-reference/jquery-selectors.php>

VIDEO LINKS

- https://www.youtube.com/watch?v=a59kOE2Ma1Q&list=PL6n9fhu94yhVDV697uvHpavA3K_eWGQap
- https://www.youtube.com/watch?v=Rkvn_MA04fo