jQuery

- jQuery is a lightweight, "write less, do more", JavaScript library.
- The purpose of jQuery is to make it much easier to use JavaScript on your website.
- jQuery takes a lot of common tasks that require many lines of JavaScript code to accomplish, and wraps them into methods that you can call with a single line of code.
- jQuery also simplifies a lot of the complicated things from JavaScript, like AJAX calls and DOM manipulation.

The jQuery library contains the following features:

- HTML/DOM manipulation
- CSS manipulation
- HTML event methods
- Effects and animations
- AJAX
- Utilities

- There are lots of other JavaScript libraries out there, but jQuery is probably the most popular, and also the most extendable.
- Many of the biggest companies on the Web use jQuery, such as:
    - Google
    - Microsoft
    - IBM
    - Netflix

Adding jQuery to Your Web Pages

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

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

```
<head>
<script src="jquery-3.4.1.min.js"></script>
</head>
```

Tip: Place the downloaded file in the same directory as the pages where you wish to use it.

jQuery Syntax

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

Basic syntax is: **$(*selector*).*action*()**

- A $ sign to define/access jQuery
- A (*selector*) to "query (or find)" HTML elements
- A jQuery *action*() to be performed on the element(s)

Examples:

`$(this).hide()` - hides the current element.

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

`$(".test").hide()` - hides all elements with class="test".

`$("#test").hide()` - hides the element with id="test".

The Document Ready Event

all jQuery methods are inside a document ready event:

```
$(document).ready(function(){

  // jQuery methods go here...

});
```

- This is to prevent any jQuery code from running before the document is finished loading (is ready).
- It is good practice to wait for the document to be fully loaded and ready before working with it. This also allows you to have your JavaScript code before the body of your document, in the head section.
- Here are some examples of actions that can fail if methods are run before the document is fully loaded:
    - Trying to hide an element that is not created yet
    - Trying to get the size of an image that is not loaded yet

jQuery Selectors

- jQuery selectors allow you to select and manipulate HTML element(s).
- jQuery selectors are used to "find" (or select) HTML elements based on their **name, id, classes, types, attributes, values of attributes and much more**. It's based on the existing CSS Selectors, and in addition, it has some own custom selectors.
- All selectors in jQuery start with the dollar sign and parentheses: **$()**.

The element Selector

The jQuery element selector selects elements based on the element name. You can select all `<p>` elements on a page like this:

```
$("p")
```

```
$(document).ready(function(){
  $("button").click(function(){
    $("p").hide();
  });
});
```

The #id Selector `$("#test")`

The .class Selector `$(".test")`

Syntax	Description
$("*")	Selects all elements
$(this)	Selects the current HTML element
$("p.intro")	Selects all <p> elements with class="intro"
$("p:first")	Selects the first <p> element
$("ul li:first")	Selects the first element of the first
$("ul li:first-child")	Selects the first element of every
$("[href]")	Selects all elements with an href attribute

jQuery DOM Manipulation

Get 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

Get Attributes - attr()

The jQuery `attr()` method is used to get attribute values.

jQuery - Set Content and Attributes

```
$("#btn1").click(function(){
  $("#test1").text("Hello world!");
});
$("#btn2").click(function(){
  $("#test2").html("<b>Hello world!</b>");
});
$("#btn3").click(function(){
  $("#test3").val("Dolly Duck");
});
```

A Callback Function for text(), html(), and val()

All of the three jQuery methods above: `text()`, `html()`, and `val()`, also come with a callback function.

The callback function has two parameters: the index of the current element in the list of elements selected and the original (old) value.

You then return the string you wish to use as the new value from the function.

```
$("#btn1").click(function(){
  $("#test1").text(function(i, origText){
    return "Old text: " + origText + " New text:
Hello world!
    (index: " + i + ")";
  });
});

$("#btn2").click(function(){
  $("#test2").html(function(i, origText){
    return "Old html: " + origText + " New html:
Hello <b>world!</b>
    (index: " + i + ")";
  });
});
```

Set Attributes - attr()

```
$("button").click(function(){
  $("#w3s").attr("href", "https://www.w3schools.com/jquery/");
});
```

The `attr()` method also allows you to set multiple attributes at the same time

```
$("button").click(function(){
  $("#w3s").attr({
    "href" : "https://www.w3schools.com/jquery/",
    "title" : "W3Schools jQuery Tutorial"
  });
});
```

A Callback Function for attr()

```
$("button").click(function(){
  $("#w3s").attr("href", function(i, origValue){
    return origValue + "/jquery/";
  });
});
```

jQuery - AJAX Introduction

- AJAX = Asynchronous JavaScript and XML.
- In short; AJAX is about loading data in the background and display it on the webpage, without reloading the whole page.
- Examples of applications using AJAX:
    - Gmail, Google Maps, Youtube, and Facebook tabs

jQuery and AJAX?

- jQuery provides several methods for AJAX functionality.
- With the jQuery AJAX methods, you can request text, HTML, XML, or JSON from a remote server using both HTTP Get and HTTP Post -
- And you can load the external data directly into the selected HTML elements of your web page!

jQuery load() Method

The jQuery `load()` method is a simple, but powerful AJAX method.

The `load()` method loads data from a server and puts the returned data into the selected element.

Syntax:

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

The required URL parameter specifies the URL you wish to load.

The optional data parameter specifies a set of querystring key/value pairs to send along with the request.

The optional callback parameter is the name of a function to be executed after the `load()` method is completed.

Here is the content of our example file: "demo_test.txt":

```
<h2>jQuery and AJAX is FUN!!!</h2>
<p id="p1">This is some text in a paragraph.</p>
```

```
$("#div1").load("demo_test.txt");
```

It is also possible to add a jQuery selector to the URL parameter.

The following example loads the content of the element with id="p1", inside the file "demo_test.txt", into a specific `<div>` element:

```
$("#div1").load("demo_test.txt #p1");
```

The optional callback parameter specifies a callback function to run when the `load()` method is completed. The callback function can have different parameters:

- `responseTxt` - contains the resulting content if the call succeeds
- `statusTxt` - contains the status of the call
- `xhr` - contains the XMLHttpRequest object

The following example displays an alert box after the load() method completes. If the `load()` method has succeeded, it displays "External content loaded successfully!", and if it fails it displays an error message:

```
$("button").click(function(){
  $("#div1").load("demo_test.txt", function(responseTxt,
statusTxt, xhr){
    if(statusTxt == "success")
      alert("External content loaded successfully!");
    if(statusTxt == "error")
      alert("Error: " + xhr.status + ": " + xhr.statusText);
  });
});
```

jQuery AJAX

The jQuery library includes various methods to send Ajax requests. These methods internally use XMLHttpRequest object of JavaScript. The following table lists all the Ajax methods of jQuery.

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

Syntax:

$.ajax(url,[options])

$.post(url,data,callback_function,type)

$.get(url,data,callback_function)

$.load(url,data)

jQuery ajax() Method

The jQuery ajax() method provides core functionality of Ajax in jQuery. It sends asynchronous HTTP requests to the server.

Syntax:

```
$.ajax(url);

$.ajax(url,[options]);
```

Parameter description:

- url: A string URL to which you want to submit or retrieve the data
- options: Configuration options for Ajax request. An options parameter can be specified using JSON format. This parameter is optional.

The following table list all the options available for configuring Ajax request.

Options	Description
data	A data to be sent to the server. It can be JSON object, string or array.
dataType	The type of data that you're expecting back from the server.
error	A callback function to be executed when the request fails.
success	A callback function to be executed when Ajax request succeeds.
timeout	A number value in milliseconds for the request timeout.
type	A type of http request e.g. POST, PUT and GET. Default is GET.
url	A string containing the URL to which the request is sent.

Send Ajax Request

The ajax() methods performs asynchronous http request and gets the data from the server. The following example shows how to send a simple Ajax request.

Example: jQuery Ajax Request

```
$.ajax('getHelloWorld.php',   // request url
    {
        success: function (data, status, xhr) {// success callback
function
            $('p').html(data);
    }
});

<p></p>
```

getHelloWorld.php

```
<?php
    echo "Hello World";
?>
```

ajax_demo.html

```
<!DOCTYPE html>
<html>
<head>
	<title>jQuery</title>
	<script src="jquery.min.js"></script>
	<script type="text/javascript">
		$(document).ready(function(){

		  $("#button2").click(function(){
		      var regno = $("#regno").val();
		      var name = $("#name").val();
		    $.ajax('ajax.php',{
		      type:"POST",
		      data:{"regno":regno,"name":name},
		      success: function (data,status,xhr) {   // success callback function
		                $('p').html(data+status);
		              },
		              error: function (jqXhr, textStatus, errorMessage) { // error callback
		                $('p').html('Error: ' + errorMessage);
		              }
		    });
		  });
		});

	</script>
</head>
```

```
<body>
	Regno<input type="text" id="regno">
	<br>
	Name<input type="text" id="name"><br>
	<button type="button" id="button2">Ajax_load2</button>
	<p></p>
</body>
</html>
```

ajax.php

```
<?php

	$regno = $_POST["regno"];
	$name = $_POST["name"];
	echo "Hello world<br>".$regno."-->".$name;
?>
```

```
$(document).ready(function(){


    $("#submit").click(function(){

        //get data from the form

        $.ajax(url,{
            type:"POST",
            data:{key:value.....},
            success: function(){
            },
            error: function(){

            }
        });

    });

});
```

jQuery - AJAX get() and post() Methods

The jQuery get() and post() methods are used to request data from the server with an HTTP GET or POST request.

HTTP Request: GET vs. POST

Two commonly used methods for a request-response between a client and server are: GET and POST.

- **GET** - Requests data from a specified resource
- **POST** - Submits data to be processed to a specified resource

GET is basically used for just getting (retrieving) some data from the server. **Note:** The GET method may return cached data.

POST can also be used to get some data from the server. However, the POST method NEVER caches data, and is often used to send data along with the request.

jQuery $.get() Method

The `$.get()` method requests data from the server with an HTTP GET request.

Syntax:

```
$.get(URL,callback);
```

The required URL parameter specifies the URL you wish to request.

The optional callback parameter is the name of a function to be executed if the request succeeds.

The following example uses the `$.get()` method to retrieve data from a file on the server:

```
$("button").click(function(){
  $.get("demo_test.php", function(data, status){
    alert("Data: " + data + "\nStatus: " + status);
  });
});
```

- The first parameter of `$.get()` is the URL we wish to request ("demo_test.asp").
- The second parameter is a callback function. The first callback parameter holds the content of the page requested, and the second callback parameter holds the status of the request.

jQuery $.post() Method

The `$.post()` method requests data from the server using an HTTP POST request.

Syntax:

```
$.post(URL,data,callback);
```

The required URL parameter specifies the URL you wish to request.

The optional data parameter specifies some data to send along with the request.

The optional callback parameter is the name of a function to be executed if the request succeeds.

The following example uses the `$.post()` method to send some data along with the request:

```
$("button").click(function(){
  $.post("demo_test_post.php",
  {
    name: "Donald Duck",
    city: "Duckburg"
  },
  function(data, status){
    alert("Data: " + data + "\nStatus: " + status);
  });
});
```