UNIT-IV JSON

What is JSON?

- Stands for **JavaScript Object Notation**
- Lightweight text-based open standard designed for human-readable data interchange.
- JSON is a subset of Java Script.
- JSON can be parsed by a Java Script parser.
- JSON filename extension is **.json**
- JSON Internet Media type is **application/json**

Uses of JSON

- The JSON standard is language-independent and its data structures, arrays and objects, are universally recognized.
- JSON is used when writing JavaScript based application
- JSON format is used for serializing & transmitting structured data over network connection.
- JSON is primarily used to transmit data between server and web application.
- JSON can be used with modern programming languages.

JSON Syntax :

JSON syntax is a subset of the JavaScript object notation syntax:

- Data is in name/value pairs
- Data is separated by commas
- Curly braces hold objects
- Square brackets hold arrays

JSON Name/Value Pairs

- JSON data is written as name/value pairs.
- A name/value pair consists of a field name (in double quotes), followed by a colon, followed by a value:
  "firstName" : "John“
- This is simple to understand, and equals to the **JavaScript statement**:
  firstName = "John"

Creating simple JSON Objects

```
Customer =
  {
     "name” : "John Johnson",
               "street” : "Oslo West 16",
               "age“ : 33,
               "phone” : "555 1234567”
  }
```

Creating JSON Objects (Preferred Way)

```
 {
        "customer":
         {
               "name” : "John Johnson",
               "street” : "Oslo West 16",
               "age“ : 33,
                    "phone” : "555 1234567”  }   }
```

JSON Data Types:

Type	Description
Number	double- precision floating-point format in JavaScript
String	double-quoted Unicode with backslash escaping
Boolean	true or false
Array	an ordered sequence of values
Value	it can be a string, a number, true or false, null etc
Object	an unordered collection of key:value pairs
Whitespace	can be used between any pair of tokens
null	empty

Numbers

- Similar to most programming languages
- It is a double precision floating-point format in JavaScript and it depends on implementation.

E.g.:

```
Customer =
   {
       "age": 21
   }
```

To find the age, we can use **customer.age**

Strings

A string begins and ends with quotation marks.

E.g.:

```
Customer=
   {
     "name":"Marcus",
     "language":"Portuguese"
   }
```

To find out the language,we use **customer.language**

Array

- It is an ordered collection of values.
- These are enclosed square brackets which means that array begins with .[. and ends with .]..
- The values are separated by ,(comma).

- Array indexing can be started at 0 or 1.
- Arrays should be used when the key names are sequential integers.

Eg: book.json

```
{
"book": [
{ "id":"01", "language": "Java", "edition": "third","author": "Herbert Schildt" },
{ "id":"07", "language": "C++", "edition": "second" ,"author": "E.Balagurusamy"}
]
}
```

Boolean

It includes true or false values.

syntax:

```
var json-object-name = { string : true/false, .......}
```

eg:

```
var obj = {name: 'Amit', marks: 97, distinction: true}
```

Object

- It is an unordered set of name/value pairs.
- Object are enclosed in curly braces that is it starts with '{' and ends with '}'.
- Each name is followed by ':'(colon) and the name/value pairs are separated by , (comma).
- The keys must be strings and should be different from each other.
- Objects should be used when the key names are arbitrary strings

Eg:

```
{
"id": "011A",
"language": "JAVA",
"price": 500,
}
```

null

It means empty type.

syntax: null

example:

```
var i = null;
```

JSON Simple object Creation using JavaScript Example-1

```
<html>
<body>
<h2>JSON Object Creation in JavaScript</h2>
<p>
Name: <p id="jname"></p><br>
Age: <p id="jage"></span><br>
Address: <p id="jstreet"></p><br>
Phone: <p id="jphone"></p><br>
</p>
```

```
<script>
customer= {
  "name":"John Johnson",
  "street":"Oslo West 16",
  "age":33,
  "phone":"555 1234567"};
document.getElementById("jname").innerHTML=customer.name;
document.getElementById("jage").innerHTML=customer.age;
document.getElementById("jstreet").innerHTML=customer.street;
document.getElementById("jphone").innerHTML=customer.phone;
</script>
</body>
</html>
```

JSON Array object Creation using Javascript Example-2

```
<html>
<body>
<h2>Create Object from JSON String</h2>
<p>Original name: <p id="origname"></p>
<script>
var employees = [
{ "firstName" : "John" , "lastName" : "Doe" },
{ "firstName" : "Anna" , "lastName" : "Smith" },
{ "firstName" : "Peter" , "lastName" : "Jones" }, ];
document.getElementById("origname").innerHTML=employees[0].firstName + " " +
                                        employees[0].lastName;
</script>
</body>
</html>
```

Reading an External JSON file

Example: External JSON file –Cust.json

```
{
          "customer":
           {
                    "name” : "John Johnson",
                    "street” : "Oslo West 16",
                    "age“ : 33,
                    "phone” : "555 1234567”
   }
}
```

Including the JSON object in HTML(JQuery)

```
<html>
<head>
<script type="text/javascript" src="jquery-1.10.2.min.js"> </script>
```

```
<script>
 $(function()
 {
   $.getJSON('cust.json', function(data)
     {
        document.write(data.customer.name);
    });
});
</script>
</head>
<body>
</body>
</html>
```

<u>Why Use JSON over XML</u>

- JSON objects are typed while XML data is typeless
  - **JSON types:** string, number, array, boolean,
  - **XML** data are all string.
- XML is used to describe structured data which doesn't include arrays whereas JSON include arrays
- Data is readily accessible as JSON objects in our JavaScript

This shows individual examples of XML and JSON:

<u>Eg:</u>

```
 JSON
{
  "company": Volkswagen,
   "name": "Vento",
  "price": 800000
 }
```

<u>Xml file:</u>

```
XML
<car>     <company>Volkswagen</company>
    <name>Vento</name>
    <price>800000</price>
 </car>
```

<u>Questions:</u>

1. What is JSON?
2. Give the uses of JSON
3. Give the syntax of JSON
4. How to read the JSON file in html?
5. Why use JSON over XML
6. What is the file extension for JSON?
7. What is the MIME type for JSON?
8. Give the structure of JSON
9. Explain the data types of JSON with example
10. Explain JSON objects with an example

11. Explain with an example how to read an external JSON file and display it in HTML

jQuery

Definition:

- jQuery is a lightweight, "write less, do more", JavaScript library.
- Fast .small feature rich JavaScript library
- jQuery greatly simplifies JavaScript programming and it is easy to learn

Many of the biggest companies on the Web use jQuery, such as:

- Google
- Microsoft
- IBM
- Netflix

Why jQuery?

- There are lots of other JavaScript frameworks out there, but jQuery seems to be the most popular, and also the most extendable.

The jQuery library contains the following features:

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

Adding jQuery to Your Web Pages

1. **Download the jQuery library from jQuery.com**

   The jQuery library is a single JavaScript file, and you reference it with the HTML <script> tag

```
<head>
 <script src="jquery-1.10.2.min.js"></script>
</head>
```

2. **Include jQuery from a CDN, like Google**

   Can include it from a CDN (Content Delivery Network) like Google and Microsoft host jQuery.

   To use jQuery from Google CDN

```
<head>
<script  src="http://ajax.googleapis.com/ajax/libs/jquery/1.10.2
                                              /jquery.min.js">
</script>
</head>
```

   To use jQuery form Microsoft CDN

```
<head>
<script src="http://ajax.aspnetcdn.com/ajax/jQuery
                                        /jquery-1.10.2.min.js">
</script>
</head>
```

jQuery Syntax

Basic syntax is:

$(selector).action()

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

Example:

```
$("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 in our examples, 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).

jQuery selectors

jQuery selectors allow you to select and manipulate HTML element(s).

The element Selector

- The jQuery element selector selects elements based on the element name.
- For example select all <p> elements on a page like this:

$("p")

The #id Selector

- The jQuery #id selector uses the id attribute of an HTML tag to find the specific element.

$("#test")

The .class Selector

- The jQuery class selector finds elements with a specific class.

$(".test")

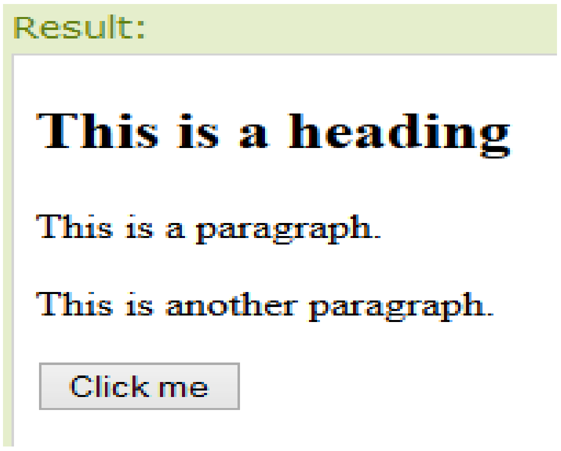
More jQuery Selectors

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

$("a[target='_blank']")	Selects all <a> elements with a target attribute value equal to "_blank"
$("a[target!='_blank']")	Selects all <a> elements with a target attribute value NOT equal to "_blank"
$(":button")	Selects all <button> elements and <input> elements of type="button"
$("tr:even")	Selects all even <tr> elements
$("tr:odd")	Selects all odd <tr> elements

Example -1-jQuery

```
<html>
<head>
<script src=" jquery-1.10.2.min.js "></script>
<script>
   $(document).ready (function()
      {
         $("button").click(function()
           {
              $("p").hide();
           });
        });
</script></head>
<body>
<h2>This is a heading</h2>
<p>This is a paragraph.</p>
<p>This is another paragraph.</p>
<button>Click me</button>
</body>
</html>
```


Result:

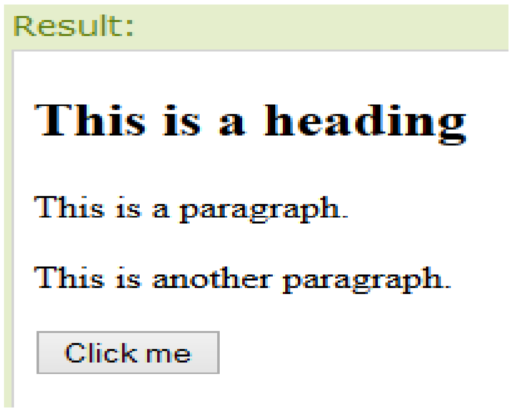
This is a heading

This is a paragraph.

This is another paragraph.

Example -2-jQuery

```
<!DOCTYPE html>
<html>
<head>
<script src=" jquery-1.10.2.min.js ">
</script>
<script>
$(document).ready(function(){
  $("button").click(function(){
    $("p.intro").hide();
  });
});
</script>
</head>
<body>
<h2 class="intro">This is a heading</h2>
<p class="intro">This is a paragraph.</p>
<p>This is another paragraph.</p>
<button>Click me</button>
</body>
</html>
```



This is a heading

This is a paragraph.

This is another paragraph.

Example -3-jQuery

```
<!DOCTYPE html>
<html>
<head>
<script src=" jquery-1.10.2.min.js ">
</script>
<script>
$(document).ready(function(){
  $("tr:even").css("background-color","yellow");
});
</script>
</head>
```

```
<body>
<h1>Welcome to My Web Page</h1>
<table border="1">
<tr><th>Company</th><th>Country</th>
</tr>
<tr><td>Alfreds Futterkiste</td><td>Germany</td>
</tr><tr>
<td>Berglunds snabbköp</td><td>Sweden</td>
</tr>
<tr><td>Centro comercial Moctezuma</td><td>Mexico</td>
</tr>
<tr><td>Ernst Handel</td><td>Austria</td>
</tr>
<tr><td>Island Trading</td><td>UK</td></tr>
</table>
</body></html>
```

jQuery Events

- jQuery is tailor-made to respond to events in an HTML page.
- An event represents the precise moment when something happens.

Examples:

- moving a mouse over an element
- selecting a radio button
- clicking on an element

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**

Example 1:

```
<html>
<head>
<script src=" jquery-1.10.2.min.js"></script>
<script>
$(document).ready(function(){
  $("#p1").mouseleave(function(){
    alert("Bye! You now leave p1!");
  });
});
</script></head>
```

```
<body>
<p id="p1">This is a paragraph.</p>
</body>
</html>
```

Example 2:

```
<html>
<head>
<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.10.2
                                        /jquery.min.js"> </script>
<script>
$(document).ready(function(){
  $("input").focus(function(){
    $(this).css("background-color","red");
  });
  $("input").blur(function(){
    $(this).css("background-color",“white");
  });
});
</script></head>
<body>
         Name: <input type="text" name="fullname"><br>
         Email: <input type="text" name="email">
</body>
</html>
```

jQuery - Get Content and Attributes

- jQuery contains powerful methods for changing and manipulating HTML elements and attributes

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

Getting the content from the HTML

```
<!DOCTYPE html>
<html>
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>
<script>
$(document).ready(function(){
  $("#btn1").click(function(){
    alert("Text: " + $("#test").text());
  });
  $("#btn2").click(function(){
    alert("HTML: " + $("#test").html());
  });
});
</script>
</head>
<body>

<p id="test">This is some <b>bold</b> text in a paragraph.</p>

<button id="btn1">Show Text</button>
<button id="btn2">Show HTML</button>

</body>
</html>
```

Get value

```
<!DOCTYPE html>
<html>
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>
<script>
$(document).ready(function(){
  $("button").click(function(){
    alert("Value: " + $("#test").val());
  });
});
</script>
</head>
<body>

<p>Name: <input type="text" id="test" value="Mickey Mouse"></p>

<button>Show Value</button>

</body>
</html>
```

SET value

```
<!DOCTYPE html>
<html>
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>
<script>
$(document).ready(function(){
  $("#btn1").click(function(){
    $("#test1").text("Hello world!");
  });
  $("#btn2").click(function(){
    $("#test2").html("<b>Hello world!</b>");
  });
  $("#btn3").click(function(){
    $("#test3").val("Dolly Duck");
  });
});
</script>
</head>
<body>

<p id="test1">This is a paragraph.</p>
<p id="test2">This is another paragraph.</p>

<p>Input field: <input type="text" id="test3" value="Mickey Mouse"></p>

<button id="btn1">Set Text</button>
<button id="btn2">Set HTML</button>
<button id="btn3">Set Value</button>

</body>
</html>
```

Adding content to a HTML element

```
<!DOCTYPE html>
<html>
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>
<script>
$(document).ready(function(){
  $("#btn1").click(function(){
    $("p").append(" <b>Appended text</b>.");
  });

  $("#btn2").click(function(){
    $("ol").append("<li>Appended item</li>");
  });
});
</script>
</head>
<body>
```

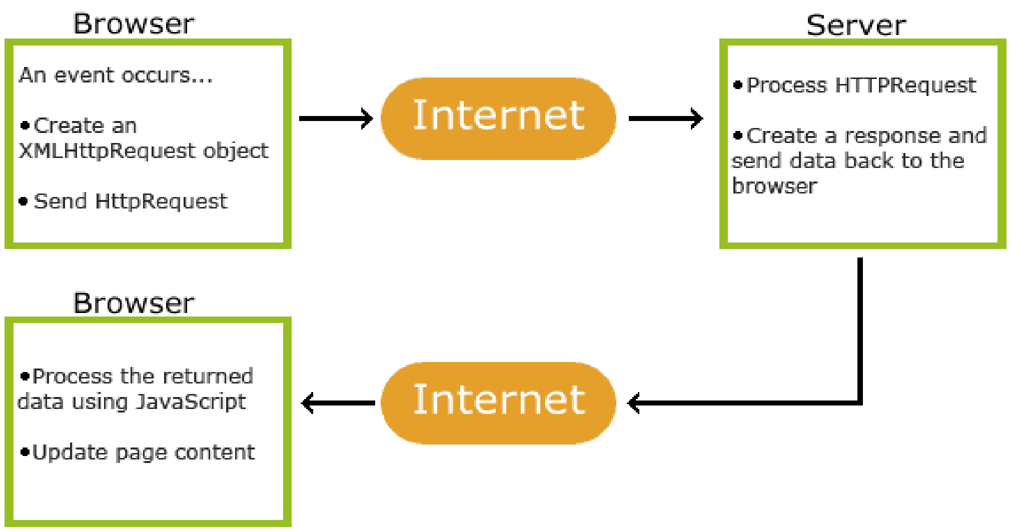
```
<p>This is a paragraph.</p>
<p>This is another paragraph.</p>

<ol>
  <li>List item 1</li>
  <li>List item 2</li>
  <li>List item 3</li>
</ol>

<button id="btn1">Append text</button>
<button id="btn2">Append list items</button>

</body>
</html>
```

AJAX

Definition:

- AJAX stands for Asynchronous JavaScript and XML.
- AJAX is the art of exchanging data with a server asynchronously
- Use to update part of a web page without reloading the whole page.
- It is technique for creating fast and dynamic web pages.
- **Examples** of applications using AJAX: **Google Maps, Gmail, Youtube, and Facebook tabs.**

How AJAX Works


Browser
An event occurs...
•Create an XMLHttpRequest object
•Send HttpRequest
Internet
Server
•Process HTTPRequest
•Create a response and send data back to the browser
Browser
•Process the returned data using JavaScript
•Update page content
Internet

AJAX is Based on Internet Standards

AJAX is based on internet standards, and uses a combination of:

- XMLHttpRequest object (to exchange data asynchronously with a server)
- JavaScript/DOM (to display/interact with the information)
- CSS (to style the data)
- XML (often used as the format for transferring data)

AJAX – Request and Response

- **The XMLHttpRequest object** is used to exchange data with a server.

Syntax for creating an XMLHttpRequest object:

```
variable=new XMLHttpRequest();
```

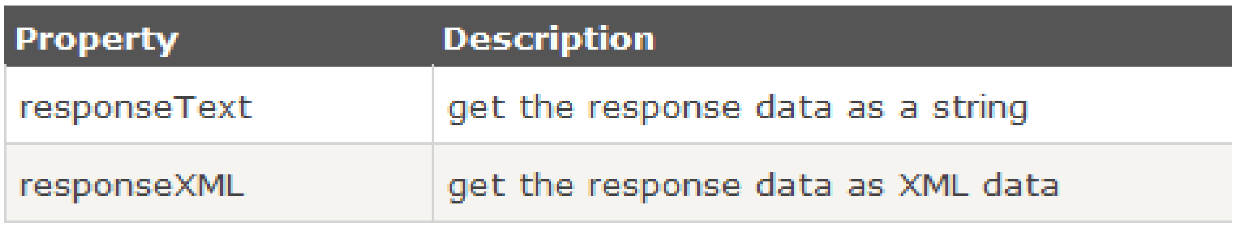
Old versions of Internet Explorer (IE5 and IE6) uses an ActiveX Object:

```
variable=new ActiveXObject("Microsoft.XMLHTTP");
```

- **To send a request to a server,** use the **open() and send()** methods of the XMLHttpRequest object:
  Syntax
  open(method,url,async)
  - method the type of the request (GET OR POST)
  - location of the file on the server
  - true (asynchronous) or false(synchronous)

  Syntax
  send() -sends the request to the server

<u>To get the response from a server, use the</u>
responseText or **responseXML** property of the XMLHttpRequest.



Property	Description
responseText	get the response data as a string
responseXML	get the response data as XML data

To send a request to a server

```
var  xmlhttp=new XMLHttpRequest();
xmlhttp.open("GET",”sample.txt", true);
xmlhttp. send();
```

To get the response from a server

```
var str=xmlhttp.responseText;
```

<u>AJAX-Events:</u>

The onreadystatechange event:

When a request to a server is sent, we want to perform some actions based on the response.

Property	**Description**
onreadystatechange	Stores a function to be called automatically each time the readyState property changes
readyState	Holds the status of the XMLHttpRequest. Changes from 0 to 4: 0: request not initialized 1: server connection established 2: request received 3: processing request 4: request finished and response is ready

status	200: "OK" 404: Page not found

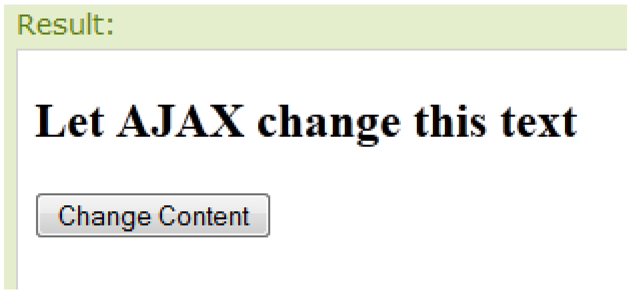
Example

```
xmlhttp.onreadystatechange=function()
  {
  if (xmlhttp.readyState==4 && xmlhttp.status==200)
    {
    document.getElementById("myDiv").innerHTML=xmlhttp.responseText;
    }
  }
```

AJAX Example program

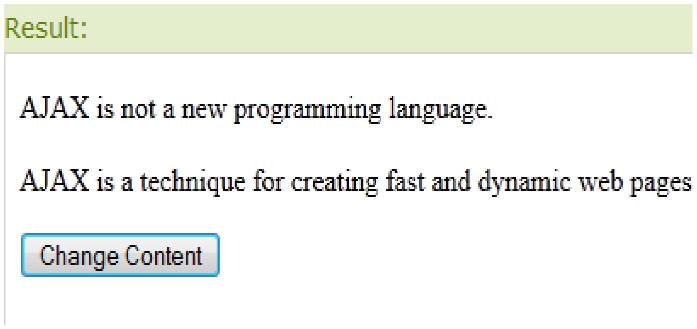
```
<!DOCTYPE html>
<html>
<head>
<script>
function loadXMLDoc()
{
var xmlhttp;
if (window.XMLHttpRequest)
  {// code for IE7+, Firefox, Chrome, Opera, Safari
  xmlhttp=new XMLHttpRequest();
  }
else
  {// code for IE6, IE5
  xmlhttp=new ActiveXObject("Microsoft.XMLHTTP");
  }
xmlhttp.onreadystatechange=function()
  {
  if (xmlhttp.readyState==4 && xmlhttp.status==200)
    {
    document.getElementById("myDiv").innerHTML=xmlhttp.responseText;
    }
  }
xmlhttp.open("GET","ajax_info.txt",true);
xmlhttp.send();
}
</script>
</head>
<body>
```

```
<div id="myDiv"><h2>Let AJAX change this text</h2></div>
<button type="button" onclick="loadXMLDoc()">Change Content</button>

</body>
</html>
```



Result:

Let AJAX change this text



Result:

AJAX is not a new programming language.

AJAX is a technique for creating fast and dynamic web pages

jQuery and AJAX

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

jQuery AJAX Methods

- **$(selector).load():** Load a piece of text or html into a container DOM.
- **$(selector).getJSON():** Load a JSON with GET method.
- **$(selector).getScript():** Load a JavaScript.
- **$(selector).get():** - To make a GET call
- **$(selector).post()**:- To make a POST call
- **$(selector).ajax()**: The ajax() method is used to perform an AJAX (asynchronous HTTP) request.

getScript() Method

- The getScript() method is used to get and execute a JavaScript using an AJAX HTTP GET request.

Syntax

$(selector).getScript(url, success(response,status))

- url- Specifies the url to send the request
- *success(response,status)* -Optional.
  Specifies the function to run if the request succeeds

Example:

```
<html>
<head>
<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.10.2/jquery.min.js">
</script>
<script>
$(document).ready(function(){
  $("button").click(function(){
    $.getScript("demo.js");
  });
});
</script>
</head>
<body>
<button>Use Ajax to get and then run a JavaScript</button>
</body>
</html>
```

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() Method:

- The $.get() method requests data from the server with an HTTP GET request.
  Syntax:
  $.get(*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.

$.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.

jQuery AJAX Methods

ajax() Method

- The ajax() method is used to perform an AJAX (asynchronous HTTP) request.
- This method is mostly used for requests where the other methods cannot be used.

Syntax

$.ajax({name:value, name:value, ... })

The parameters specifies one or more name/value pairs for the AJAX request.

url	Specifies the URL to send the request to. Default is the current page
dataType	The data type expected of the server response.
data	Specifies data to be sent to the server
success(result,status,xhr)	A function to be run when the request succeeds

Example:

```
<html>
<head>
<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.10.2/jquery.min.js">
</script>
<script>
$(document).ready(function(){
  $("button").click(function(){
    $.ajax( {url:"demo_ajax_script.js",dataType:"script"});  });
});
</script>
</head>
<body>
<button>Use Ajax to get and then run a JavaScript</button>
</body>
</html>
```