**jQuery – AJAX**

# Loading simple data:

This is very easy to load any static or dynamic data using JQuery AJAX. JQuery provides **load()** method to do the job:

## Syntax:

Here is the simple syntax for **load()** method:

|  |
| --- |
| **[selector].**load( URL, [data], [callback] ); |

Here is the description of all the parameters:

* **URL:** The URL of the server-side resource to which the request is sent. It could be a CGI, ASP, JSP, or PHP script which generates data dynamically or out of a database.
* **data:** This optional parameter represents an object whose properties are serialized into properly encoded parameters to be passed to the request. If specified, the request is made using the **POST** method. If omitted, the **GET** method is used.
* **callback:** A callback function invoked after the response data has been loaded into the elements of the matched set. The first parameter passed to this function is the response text recieved from the server and second parameter is the status code.

## Example 1:

|  |
| --- |
| <html>  <head>  <title>the title</title>  <script type="text/javascript" src="jslib/jquery-1.11.1.js"></script>  <script type="text/javascript" language="javascript">  $(document).ready(function() {  $("#driver").click(function(event){  $('#stage').load('jslib/result.html', function(){  alert("Loaded");  });  });  });  </script>  </head>  <body>  <p>Click on the button to load result.html file:</p>  <div id="stage" style="background-color:pink;">  STAGE  </div>  <input type="button" id="driver" value="Load Data" />  </body>  </html> |

Here **load()** initiates an Ajax request to the specified URL **/jslib/result.html** file. After loading this file, all the content would be populated inside <div> tagged with ID *stage*. Assuming, our /jslib/result.html file has three HTML lines:

|  |
| --- |
| <li>THIS IS RESULT...</li>  <li>Another Items.....</li>  Outside the List |

When you click the given button, then result.html file gets loaded. However, you may get only a part of the result.html file by changing the load statement as:

|  |
| --- |
| $('#stage').load('jslib/result.html **li**', function(){ |

# Try it yourself!

# Getting JSON data:

There would be a situation when server would return JSON string against your request. JQuery utility function **getJSON()** parses the returned JSON string and makes the resulting string available to the callback function as first parameter to take further action.

## Syntax:

Here is the simple syntax for **getJSON()** method:

|  |
| --- |
| **[selector].**getJSON( URL, [data], [callback] ); |

Here is the description of all the parameters:

* **URL:** The URL of the server-side resource contacted via the GET method.
* **data:** An object whose properties serve as the name/value pairs used to construct a query string to be appended to the URL, or a preformatted and encoded query string.
* **callback:** A function invoked when the request completes. The data value resulting from digesting the response body as a JSON string is passed as the first parameter to this callback, and the status as the second.

## Example 2:

|  |
| --- |
| <html>  <head>  <title>the title</title>  <script type="text/javascript"  src="jslib/jquery-1.11.1.js"></script>  <script type="text/javascript" language="javascript">  $(document).ready(function() {  $("#driver").click(function(event){  $.getJSON('jslib/result.json', function(jd) {  $('#stage').html('<p> Name: ' + jd.name + '</p>');  $('#stage').append('<p>Age : ' + jd.age+ '</p>');  $('#stage').append('<p> Sex: ' + jd.sex+ '</p>');  });  });  });  </script>  </head>  <body>  <p>Click on the button to load result.html file:</p>  <div id="stage" style="background-color:pink;">  STAGE  </div>  <input type="button" id="driver" value="Load Data" />  </body>  </html> |

Here JQuery utility method **getJSON()** initiates an Ajax request to the specified URL **/jslib/result.json** file. After loading this file, all the content would be passed to the callback function which finally would be populated inside <div> tagged with ID *stage*. Assuming, our /jslib/result.json file has following json formatted content:

|  |
| --- |
| {  "name": "Zara Ali",  "age" : "67",  "sex": "female"  } |

When you click the given button, then result.json file gets loaded.

If the data are in array format, the data file becomes:

|  |
| --- |
| {  "name": ["Zara Ali","Ali"],  "age": ["67",77],  "sex": ["female","male"]  } |

**or**

[{

"name": "Zara Ali”,

"age": "67",

"sex": "female"

} ,

{

"name": "Ali",

"age": 77,

"sex": "male"

} ]

JSON (JavaScript Object Notation)



An array is an ordered collection of values. An array begins with [ (left bracket) and ends with ] (right bracket). Values are separated by , (comma).



A value can be a string in double quotes, or a number, or true or false or null, or an object or an array. These structures can be nested.



A string is a sequence of zero or more Unicode characters, wrapped in double quotes, using backslash escapes. A character is represented as a single character string. A string is very much like a C or Java string.

