JQuary:-1

**What is JQuary?**

JQuary is a lightweight JavaScript library that simplifies programming with JavaScript

**According to JQuary.com**

JQuary is a fast, small, and feature-rich JavaScript library. It makes thing like HTML document traversal and manipulation, event handling, animation, and Ajax much simpler with an easy-to-use API that works across a multitude of browsers.

With a combination of versatility and extensibility, JQuary has changed the way that millions of people write JavaScript.

Why should we use JQuary or Advantages of using JQuary over raw JavaScript?

The use of JQuary has several benefits over using the raw JavaScript.

\* JQuary is cross-browser

\* JQuary is a lot more easy to use then raw JavaScript.

\* JQuary is extensible

\* JQuary simplifies and has rich AJAX support

\* JQuary has large development community and many plugins. Example autocomplete

Textbox plugin.

\* Excellent documentation

**How to use JQuary in a Web Application?**

Download the JQuary file from JQuary.com and reference it in your application just

Like other JavaScript file.

**What is the difference between JQuary1.x and 2.x?**

If you want to support IE6/7/8 ,then user JQuary 1.x where as if you don’t not have the need to support IE^/7/7 then use JQuary2.x JQuary 2.x is smaller in size then JQuary 1.x.

Adding a click event handler for a button control using JQuary.

<script type ="text/JavaScript">

$('document').ready(function()

{

$('#button1').click(function()

{

alert('jQuary tutorial');

});

);

</script>

<input type="button" value="Click me" id="button1"/>

With JQuary we have less code to achieve the same thing. We don't have to worry about cross-browser issues, as all this is takes care by JQuary.

**Point to be remember:-**

1.ready() function ensure that the DOM is fully loaded.

2.$ is shortcut for JQuary.

3. All three of the following syntaxes are equivalent:.

$(document).ready(handler)

$().ready (handler)(this is not recommended)

$(handler)

html xmlns="http://www.w3.org/1999/xhtml">

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

<head runat="server">

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

<title></title>

</head>

<body>

<form id="form1" runat="server">

<div>

<script type="text/javascript">

//window.onload = function () {

// document.getElementById('button1').addEventListener('click', clickHandler,false);

//}

//function clickHandler()

//{

// alert('jQuary tutorial');

//}

jQuery(document).ready(function () {

jQuery('#button1').click(function () {

alert('jQuary tutorial');

});

});

</script>

<input type="button" id="button1" value="Click ME" />

</div>

</form>

</body>

</html>

2.$(document).ready(function()in JQuary:-

$(document).ready is a JQuary event. It fires as soon as the DOM is loaded and ready to be manipulated by script. This is the earliest point in the page load process where the script can safely access element in the page's html dom. This events is fired before all the images, css etc... are fully loaded.

<html xmlns="http://www.w3.org/1999/xhtml">

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

<head runat="server">

<title></title>

</head>

<body>

<form id="form1" runat="server">

<script type="text/javascript">

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

$("#button1").click(function () {

alert("J Query Tutorial");

})

})

</script>

<input type="button" id="button1" value="Click ME" />

<div id="div1"></div>

</form>

</body>

</html>

**In the following example, we have removed the ready () method.** When you click the button now, you don’t get the alert. This is because the jQuery code is present before the button element, so by the time the jQuery code is not loaded into DOM.

<html xmlns="http://www.w3.org/1999/xhtml">

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

<head runat="server">

<script type="text/javascript">

</script>

<title></title>

</head>

<body>

<form id="form1" runat="server">

<script type="text/javascript">

$("#button1").click(function () {

alert("J Query Tutorial");

})

</script>

<input type="button" id="button1" value="Click ME" />

<div id="div1"></div>

</form>

</body>

</html>

**To make this example work, you have 2 options.**

1. Place your JQuary code in $(document).ready function or

2. Place your script at the bottom of the page just before the closing</body> element.

<body>

<form id="form1" runat="server">

<input type="button" id="button1" value="Click ME" />

<script type="text/javascript">

//$('document').ready(function () {

// $("#button1").click(function () {

// alert("J Query Tutorial");

// })

//})

$("#button1").click(function () {

alert("J Query Tutorial");

})

</script>

<div id="div1"></div>

</form>

</body>

$(window).load Event:-

It fires when the Dom and all the content on the page (images, css. etc. ) is fully loaded. Since the window load events waits for images, css etc. to be fully loaded, this events fires after ready event.

<html xmlns="http://www.w3.org/1999/xhtml">

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

<head runat="server">

<script type="text/javascript">

$(window).load(function () {

alert('Window loaded');

});

$(document).ready(function () {

alert("DOM loaded");

});

</script>

<title></title>

</head>

<body>

</body>

</html>

When you run the page with above script, notice that the alert is ready function is displayed before the alert load function.

**In most cases, the script can be run as soon as the dom hierarchy has been fully constructed**.so ready() is usually the best place to write your JavaScript code. However, in your application there could be scenarios where you should be using $(window).load over $(document).ready.

**Example:-**to get the actual image dimensions, we will have to wait until the image is fully loaded, so the JQuary code to get the height and width should be in $(window).load event.

<html xmlns="http://www.w3.org/1999/xhtml">

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

<head runat="server">

<script type="text/javascript">

//$(window).load(function () {

// alert('Window loaded');

//});

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

// alert("DOM loaded");

//})

$(window).load(function () {

$('#div1').html("Height= " + $('#img1').height() + "<br/>" + "Width= " + $('#img1').width());

});

</script>

<title></title>

</head>

<body>

<div id="div1"></div>

<img src="Chrysanthemum.jpg" id="img1"/>

</body>

</html>

3. What is CDN?

CDN stands for Content Delivery Network. A CDN is a system of distributed servers that hosts resources such as images, css, javascript files etc.

Companies like Microsoft, Google, Yahoo etc. have a free public CDN from which we can load JQuary instead of hosting it on our web server

Microsoft JQuery CDN.

http://www.asp.net/ajax/cdn#jquery\_release\_on\_the\_CDN\_)

Google JQuery CDN

https://developers.google.com/speed/libraries/devguide#JQuery

**Benefits of Using CDN**

Distributed CDN servers: The query file can be download from the CDN server is closest to the user.

Browser Caching:- jQuery is used on many popular websites. If a user has already visited a webpages that uses jQuery from CDN, and then if he arrives at your page, the JQuery file has been already been cached by the browser so there is no need download it again.

Parallel Downloads:-There is a browser limit on how many file can be concurrently downloaded from a give domain. This number varies from browser to browser. For example, if the browser allows only 2 concurrent downloads from a given domain, the 3rd download is blocked until one of the previous file has been fully download. since the jQuery file on a CDN.it is being download form the different domain. So this means the browser allows 2 parallel download form the CDN server.

Reduced server Load:-The HTTP request for JQuery file is handled by the CDN server, so the load on your web server is reduce. This also means there is a saving on your website bandwidth consumption which is tum will reduce your hosting cost.

**Disadvantages:-**your client firewall may block the CDN. So your may have to request your clients to whitelist the CDN.

What if the required JQuery file cannot be download form CDN.?

Let’s assume that, the CDN is down or because of some network issue we are not able to download jQuery from CDN. In this case we will have to fall-back to use JQuery file that we hosted on our own server.

If JQuery is successfully download, jQuery property is added to the window object. If the property is not found then jQuery is not downloading

We can use this way to validate in case if download error from CDN.

window.jQuery || document.write("<script src='jquery-1.10.2.js'><\/script>");

<html xmlns="http://www.w3.org/1999/xhtml">

<head runat="server">

<title></title>

<%--<script src="jquery-1.10.2.js"></script>--%>

<script src="http://ajax.aspnetcdn.com/ajax/jQuery/jquery-2.1.4.js"></script>

<script>

window.jQuery || document.write("<script src='jquery-1.10.2.js'><\/script>");

</script>

</head>

<body>

<script type="text/javascript">

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

$('#button1').click(function () {

alert('jQuery Tutorial');

});

});

</script>

<input type="button" id="button1" value="Click ME!" />

</body>

</html>

4.JQuery #id Selector:-

**What are JQuery selectors?**

One of the most important concepts in jQuery is selector. JQuery selectors allow you to select and manipulate HTML elements.

**Different selectors in JQuery**

JQuery selectors allow you to select html elements in the DOM by

1. Element ID

2. Element Tag Name

3. Element ClassName

4. Element attributes

5. Element Attribute values and many more.

**1. Element Id:-**

**JQuery #id selector uses the javascript document.getElementByID() function**. JQuery #Id selector is the most efficient amount all jQuery selector. If you know the id of an element that you want to find, then always use the #id selector.

**HTML element IDS must be unique on the page. JQuery #Id selector will not throw an error. To check if an element is returned by the #id selector use length property.**

Javascript’s document.getElementById() function throws an error if the statement with the given id is not found, whereas jQuery #id selector will not throw an error. To check if an element is return by the #id selector use length property.

**Javascript document.getElementById() and JQuery(#Id) selector are not the same.**

**document.getElmentbyId() returns a raw DOM objects whereas jQuery ('#id') selector**

**Returns a JQuery object that wraps the DOM objects and provides jQuery method. This is the reason you are able to call JQuery methods like css(),click() on the objects returned by jQuery. To get the underlying DOM Objects from a JQuery objects use $('#id')[0]**

document.getElementById() is faster than jQuery selector. Use document.getElmentbyId()

Over JQuery('#id') selector unless you need the extra functionality provided by the JQuery object.

<html xmlns="http://www.w3.org/1999/xhtml">

<head runat="server">

<title></title>

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

<script type="text/javascript">

$(document).ready(function () {

$('#button1').css('background-color', "yellow");

// document.getElementById('button2').style.backgroundColor = "yellow";

if ($('button1').length>0) {

alert("element not found.")

}

$('#button1').click(function () {

alert("Hello JQuery tutorials");

});

});

</script>

</head>

<body>

<form id="form1" runat="server">

<div>

<input id="button1" type="button" value="Click Me" />

<input id="button1" type="button" value="Click Me" />

</div>

</form>

</body>

</html>

**5. JQuery Element Selector**

In this video we will discuss **jQuery Element Selector**, i.e selecting elements by tag name.

To select the elements by tag name use **jQuery Element Selector**  
  
**Syntax :** $(element)

$('td') // Selects all td elements

$('div a') // Select all anchor elements that are descendants of div element

$('div, span, a') // Selects all div, span and anchor elements

Alerts the total count of td elements on the page

<html>

<head>

    <title></title>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            alert($('td').length);

        });

    </script>

</head>

<body>

    <table style="border:1px solid black; border-collapse:collapse">

        <tr>

            <td>C#</td>

            <td>ASP.NET</td>

            <td>SQL Server</td>

        </tr>

        <tr>

            <td>ADO.NET</td>

            <td>jQuery</td>

            <td>JavaScript</td>

        </tr>

        <tr>

            <td>AJAX</td>

            <td>CSS</td>

            <td>HTML</td>

        </tr>

        <tr>

            <td>VB</td>

            <td>Dot NET</td>

            <td>Visual C++</td>

        </tr>

        <tr>

            <td>Oracle</td>

            <td>Java</td>

            <td>J2EE</td>

        </tr>

    </table>

    <br /><br />

    <div>

        DIV 1

        <br />

        <a href="http://pragimtech.com">PragimTech</a>

    </div>

    <br /><br />

    <a href="http://google.com">Google</a>

    <br /><br />

    <div>DIV 2</div>

    <br /><br />

    <span>SPAN 1</span>

    <br /><br />

    <div>DIV 3</div>

</body>

</html>

Selects all the tr elements on the page and changes their background colour to red

<script type="text/javascript">

    $(document).ready(function () {

        $('tr').css('background-Color', 'red');

    });

</script>

Alerts the HTML content of the table

<script type="text/javascript">

    $(document).ready(function () {

        alert($('table').html());

    });

</script>

Alerts the HTML content of each table row

<script type="text/javascript">

    $(document).ready(function () {

        $('table tr').each(function () {

            alert($(this).html());

        });

    });

</script>

Select and changes the background colour of all the div, span and anchor elements

<script type="text/javascript">

    $(document).ready(function () {

        $('div, span, a').css('background-Color', 'yellow');

    });

</script>

Select all anchor elements that are descendants of div element

<html>

<head>

    <title></title>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            $('div a').css('background-Color', 'yellow');

        });

    </script>

</head>

<body>

    <div>

        <a href="http://pragimtech.com">PragimTech</a>

    </div>

    <br />

    <a href="http://microsoft.com">Microsoft</a>

</body>

</html>

Changes the background color of even rows to gray and odd rows to yellow on both the tables.

<html>

<head>

    <title></title>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            $('tr:even').css('background-Color', 'gray');

            $('tr:odd').css('background-Color', 'yellow');

        });

    </script>

</head>

<body>

    <table id="table1" border="1">

        <tr>

            <td>C#</td>

            <td>ASP.NET</td>

            <td>SQL Server</td>

        </tr>

        <tr>

            <td>ADO.NET</td>

            <td>jQuery</td>

            <td>JavaScript</td>

        </tr>

        <tr>

            <td>AJAX</td>

            <td>CSS</td>

            <td>HTML</td>

        </tr>

        <tr>

            <td>VB</td>

            <td>Dot NET</td>

            <td>Visual C++</td>

        </tr>

        <tr>

            <td>Oracle</td>

            <td>Java</td>

            <td>J2EE</td>

        </tr>

    </table>

    <br />

    <table id="table2" border="1">

        <tr>

            <td>Mark</td>

            <td>Mary</td>

            <td>Mike</td>

        </tr>

        <tr>

            <td>John</td>

            <td>Jade</td>

            <td>Joy</td>

        </tr>

        <tr>

            <td>Rob</td>

            <td>Roy</td>

            <td>Rachel</td>

        </tr>

        <tr>

            <td>Sam</td>

            <td>Sara</td>

            <td>Smith</td>

        </tr>

        <tr>

            <td>Todd</td>

            <td>Tom</td>

            <td>Theo</td>

        </tr>

    </table>

</body>

</html>

To change the background color of even rows to gray and odd rows to yellow just for one of the table, use #id selector along with element selector.

<script type="text/javascript">

    $(document).ready(function () {

        $('#table1 tr:even').css('background-Color', 'gray');

        $('#table1 tr:odd').css('background-Color', 'yellow');

    });

</script>

6. JQuery Class Selector

Syntax : $('.class')   
  
jQuery class selectors uses JavaScript's native **getElementsByClassName**() function if the browser supports it.

$('.small') // Selects all elements with class small

$('.small,.big') // Selects all elements with class small or big

$('div.small,.big') // Selects div elements with class small and any element with class big

Selects all elements **with class "small"** and sets 5px solid red border

<html>

<head>

    <title></title>

    <script src="Scripts/jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            $('.small').css('border', '5px solid red');

        });

    </script>

</head>

<body>

    <span class="small">

        Span 1

    </span>

    <br /><br />

    <div class="small">

        Div 1

    </div>

    <br />

    <span class="big">

        Span 2

    </span>

    <p class="big">This is a paragraph</p>

</body>

</html>

Selects all elements **with class "small" or "big"** and sets 5px solid red border

<script type="text/javascript">

    $(document).ready(function () {

        $('.small, .big').css('border', '5px solid red');

    });

</script>

Selects all elements **with class "small" and all span elements with class "big"** and sets 5px solid red border

<script type="text/javascript">

    $(document).ready(function () {

        $('.small, span.big').css('border', '5px solid red');

    });

</script>

Selects all elements with **class small that are nested in a an element with id=div2** and sets 5px solid red border

<html>

<head>

    <title></title>

    <script src="Scripts/jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            $('#div2 .small').css('border', '5px solid red');

        });

    </script>

</head>

<body>

    <div id="div1" class="small">

        DIV 1

    </div>

    <br />

    <div id="div2">

        Div 2

        <br />

        <div class="small">

            DIV 3

        </div>

        <br />

        <span class="small">

            SPAN

        </span>

    </div>

</body>

</html>

Selects all elements **with class small** and sets 5px solid red border. Notice div1 has 2 classes - small and big.

<html>

<head>

    <title></title>

    <script src="Scripts/jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            $('.small').css('border', '5px solid red');

        });

    </script>

</head>

<body>

    <div class="small big">

        DIV 1

    </div>

    <br />

    <div class="small">

        DIV 2

    </div>

</body>

</html>

Selects all elements **that has both the classes - small and big**. There should be no space between the class names.

<script type="text/javascript">

    $(document).ready(function () {

        $('.small.big').css('border', '5px solid red');

    });

</script>

If you have a **space between the two class names then we are trying to find descendants**, i.e. find elements with class big that are descendants of an element with class small.

<html>

<head>

    <title></title>

    <script src="Scripts/jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            $('.small .big').css('border', '5px solid red');

        });

    </script>

</head>

<body>

    <div class="small big">

        DIV 1

    </div>

    <br />

    <div class="small">

        DIV 2

        <div class="big">

            DIV 3

        </div>

    </div>

</body>

</html>

**Another way to selects all elements that has both the classes - small and big is by using filter method**. But this approach is slower because it will first create a list of objects with class "small" and then removes elements that does not have class "big"

<script type="text/javascript">

    $(document).ready(function () {

        $('.small').filter('.big').css('border', '5px solid red');

    });

</script>

### 7. jQuery attribute selector

**In this video we will discuss selecting elements**  
1. That have specified attribute  
2. That have specified attribute values

**Syntax :**  
$('[attribute]')  
$('[attribute="value"]')

$('[title]') // Selects all elements that have title attribute

$('div[title]') // Selects all div elements that have title attribute

$('[title="divTitle"]') // Selects all elements that have title attribute value - divTitle

$('div[title="divTitle"]') // Selects all div elements that have title attribute value - divTitle

**Selects all elements with title attribute** and sets 5px solid red border

<html>

<head>

    <title></title>

    <script src="Scripts/jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            $('[title]').css('border', '5px solid red');

        });

    </script>

</head>

<body>

    <div title="div1Title">

        DIV 1

    </div>

    <br />

    <div title="div2Title">

        DIV 2

    </div>

    <p title="pTitle">

        This is a paragraph

    </p>

    <span title="div1Title">

        SAPN 1

    </span>

    <br /><br />

    <span>

        SPAN 2

    </span>

</body>  
</html>

**Selects all div elements with title attribute** and sets 5px solid red border

<script type="text/javascript">

    $(document).ready(function () {

        $('div[title]').css('border', '5px solid red');

    });

</script>

**Selects all elements with title attribute value - div1Title**, and sets 5px solid red border

<script type="text/javascript">

    $(document).ready(function () {

        $('[title="div1Title"]').css('border', '5px solid red');

    });

</script>

**Selects all div elements with title attribute value - div1Title**, and sets 5px solid red border

<script type="text/javascript">

    $(document).ready(function () {

        $('div[title="div1Title"]').css('border', '5px solid red');

    });

</script>

**Selects all div elements with both title and style attributes**, and sets 5px solid black border

<html>

<head>

    <title></title>

    <script src="Scripts/jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            $('div[title][style]').css('border', '5px solid black');

        });

    </script>

</head>

<body>

    <div title="divTitle" style="background-color:red">

        Red DIV

    </div>

    <br />

    <div title="divTitle" style="background-color:green">

        Green DIV

    </div>

    <br />

    <div title="divTitle">

        Normal Div

    </div>

    <br />

    <div>

        Normal Div without any attributes

    </div>

</body>

</html>

Selects all div elements with **title attribute value - divTitle**, and **style attribute value - background-color:red**, and sets 5px solid black border

<script type="text/javascript">

    $(document).ready(function () {

        $('div[title="divTitle"][style="background-color:red"]')

            .css('border', '5px solid black');

    });

</script>

**Selects all div elements with either title or style attributes**, and sets 5px solid black border

<script type="text/javascript">

    $(document).ready(function () {

        $('div[title],[style]').css('border', '5px solid black');

    });

</script>

### 8. jQuery attribute value selector

### In this video we will discuss  Attribute Equals Selector [name="value"] Attribute Not Equal Selector [name!="value"] Attribute Contains Selector [name\*="value"] Attribute Contains Word Selector [name~="value"] Attribute Contains Prefix Selector [name|="value"] Attribute Starts With Selector [name^="value"] Attribute Ends With Selector [name$="value"]

This is continuation to [Part 7](http://csharp-video-tutorials.blogspot.com/2015/03/jquery-attribute-selector.html), please watch [Part 7](http://csharp-video-tutorials.blogspot.com/2015/03/jquery-attribute-selector.html) before proceeding.  
  
Selects all elements that have title attribute value equal to div1Title  
$('[title="div1Title"]')   
  
Selects all elements that have title attribute value not equal to div1Title  
$('[title!="div1Title"]')   
  
Selects all elements that have title attribute value containing the given substring - Title  
$('[title\*="Title"]')   
  
Selects all elements that have title attribute value containing the given word - mySpan, delimited by spaces  
$('[title~="mySpan"]')  
  
Selects all elements that have title attribute value equal to myTitle or starting with myTitle followed by a hyphen (-)  
$('[title|="myTitle"]')   
  
Selects all elements that have title attribute value starting with div  
$('[title^="div"]')  
Selects all elements that have title attribute value ending with Heading  
$('[title$="Heading"]')  
  
Selects all elements that have title attribute value equal to div1Title and sets 5px solid red border

<html>

<head>

    <title></title>

    <script src="Scripts/jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            $('[title="div1Title"]').css('border', '5px solid red');

        });

    </script>

</head>

<body>

    <div title="div1Title">

        DIV 1

    </div>

    <br />

    <div title="div2Title">

        DIV 2

    </div>

    <p title="myTitle-paragraph">

        This is a paragraph

    </p>

    <p title="myTitleHeading">

        This is a paragraph

    </p>

    <span title="div1Title">

        SAPN 1

    </span>

    <br /><br />

    <span title="mySpan Heading">

        SPAN 2

    </span>

</body>

</html>

Selects all div elements that have title attribute value not equal to div1Title and sets 5px solid red border

<script type="text/javascript">

    $(document).ready(function () {

        $('div[title!="div1Title"]').css('border', '5px solid red');

    });

</script>

**THIS IS**

$('div[title!="div1Title"]').css('border', '5px solid red');

**EQUIVALENT TO**

$('div:not([title="div1Title"])').css('border', '5px solid red');

Selects all elements that have title attribute value containing the given substring - Title, and sets 5px solid red border

<script type="text/javascript">

    $(document).ready(function () {

        $('[title\*="Title"]').css('border', '5px solid red');

    });

</script>

Selects all elements that have title attribute value containing the given word - mySpan, delimited by spaces, and sets 5px solid red border

<script type="text/javascript">

    $(document).ready(function () {

        $('[title~="mySpan"]').css('border', '5px solid red');

    });

</script>

Selects all elements that have title attribute value equal to myTitle or starting with myTitle followed by a hyphen (-) and sets 5px solid red border

<script type="text/javascript">

    $(document).ready(function () {

        $('[title|="myTitle"]').css('border', '5px solid red');

    });

</script>

Selects all elements that have title attribute value starting with div and sets 5px solid red border

<script type="text/javascript">

    $(document).ready(function () {

        $('[title^="div"]').css('border', '5px solid red');

    });

</script>

Selects all elements that have title attribute value ending with Heading and sets 5px solid red border

<script type="text/javascript">

    $(document).ready(function () {

        $('[title$="Heading"]').css('border', '5px solid red');

    });

</script>

### 9. JQuery case insensitive attribute selector

### In this video we will discuss, how to write a jQuery case-insensitive attribute value selector. Let us understand this with an example.

The following example, only selects **DIV 1**. This is because jQuery attribute value selector is case-sensitive.

<html>

<head>

    <title></title>

    <script src="Scripts/jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            $('div[title="DivTitle"]').css('border', '3px solid red');

        });

    </script>

</head>

<body>

    <div title="DivTitle">

        DIV 1

    </div>

    <br />

    <div title="DIVTITLE">

        DIV 2

    </div>

    <br />

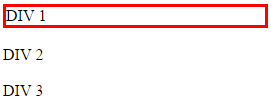
    <div title="divtitle">

        DIV 3

    </div>

</body>

</html>

  
  
Use the following code to make the jQuery attribute value selector case-insensitive

<script type="text/javascript">

    $(document).ready(function () {

        $('div[title]').filter(function () {

            return $(this).attr('title').toLowerCase() == 'divtitle';

        }).css('border', '3px solid red');

    });

</script>

The above script should select all the 3 divs.  
  
  
Now let us look at an example of making **attribute contains selector [name\*="value"], case-insensitive.**  
  
The following example, selects only DIV 1 element. This is because the attribute contains selector is case-sensitive.

<html>

<head>

    <title></title>

    <script src="Scripts/jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            $('div[title\*="Div"]').css('border', '3px solid red');

        });

    </script>

</head>

<body>

    <div title="DivTitle1">

        DIV 1

    </div>

    <br />

    <div title="DIVTITLE2">

        DIV 2

    </div>

    <br />

    <div title="divtitle3">

        DIV 3

    </div>

</body>

</html>

  
  
To make attribute contains selector case-insensitive, use filter() method and regular expression as shown below.

<script type="text/javascript">

    $(document).ready(function () {

        $('div[title]').filter(function () {

            return (/Div/i).test($(this).attr('title'));

        }).css('border', '3px solid red');

    });

</script>

OR

<script type="text/javascript">

    $(document).ready(function () {

        $('div[title]').filter(function () {

            return RegExp('Div', 'i').test($(this).attr('title'));

        }).css('border', '3px solid red');

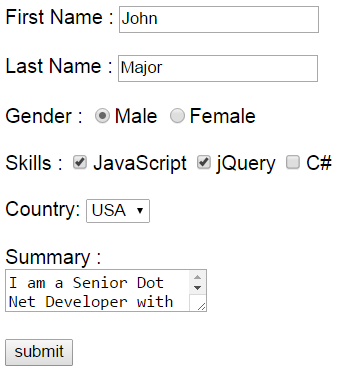
    });

</script>

### The above script should select all the 3 divs. jquery attribute contains case insensitive Visual Studio Keyboard Shortcuts Convert Selected Text to Upper Case - CTRL + SHIFT + U Convert Selected Text to Lower Case - CTRL + U

### 10. jQuery input vs :input

### In this video we will discuss the difference between $(input) and $(:input) selectors

$(':input') selects all input, textarea, select and button elements where as $('input') just selects elements with an input tag.   
  
**Consider the web page below**   
  
  
The following is the HTML of the above page

<html>

<head>

    <title></title>

    <script src="jquery-1.11.2.js"></script>

</head>

<body style="font-family:Arial">

    First Name : <input type="text" value="John" />

    <br /><br />

    Last Name : <input type="text" value="Major" />

    <br /><br />

    Gender :

    <input type="radio" name="gender" checked="checked" value="Male">Male

    <input type="radio" name="gender" value="Female">Female

    <br /><br />

    Skills :

    <input type="checkbox" name="skills" checked="checked"

           value="JavaScript" />JavaScript

    <input type="checkbox" name="skills" checked="checked"

           value="jQuery" />jQuery

    <input type="checkbox" name="skills" value="C#" />C#

    <br /><br />

    Country:

    <select>

        <option selected="selected" value="USA">USA</option>

        <option value="India">India</option>

        <option value="UK">UK</option>

    </select>

    <br /><br />

    Summary :

    <br />

    <textarea>

        I am a Senior Dot Net Developer with 10 years experience

    </textarea>

    <br /><br />

    <input type="submit" value="submit" />

</body>

</html>

Now we want to get the text value from all the textboxes. On this page we have 2 textboxes  
1. First Name  
2. Last Name   
  
**jQuery code to get textbox value using $(input)**

<script type="text/javascript">

    $(document).ready(function () {

        $('input[type="text"]').each(function () {

            alert($(this).val());

        });

    });

</script>

**jQuery code to get textbox value using $(:input)**

<script type="text/javascript">

    $(document).ready(function () {

        $(':input[type="text"]').each(function () {

            alert($(this).val());

        });

    });

</script>

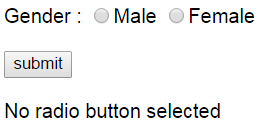
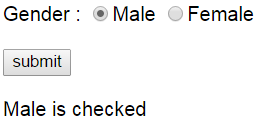
### Which one is better for performance $('input[type="text"]') or $(':input[type="text"]') $('input[type="text"]') is better for performance over $(':input[type="text"]').  This is because $(':input[type="text"]') needs to scan all input elements, textarea, select etc, where as $('input[type="text"]') scans only input elements. So if you want to find elements with an input tag, it is always better to use $('input[type="text"]') over $(':input[type="text"]')

### 11. jQuery checked selector

### In this video we will discuss the jQuery :checked selector.

The **:checked selector selects all checked check-boxes or radio buttons**. Let us understand this with an example.   
  
**Selects all checked radio button elements**

$('input[type="radio"]:checked')

**Example :** When you click the submit button without selecting any radio button, "No radio button checked" message should be displayed.   
  
  
When you click the submit button after a radio button is checked, then a message stating "Male is checked" or "Female is checked" should be displayed.   


<html>

<head>

    <title></title>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            $('#btnSubmit').click(function () {

                var result = $('input[type="radio"]:checked');

                if (result.length > 0) {

                    $('#divResult').html(result.val() + " is checked");

                }

                else {

                    $('#divResult').html("No radio button checked");

                }

            });

        });

    </script>

</head>

<body style="font-family:Arial">

    Gender :

    <input type="radio" name="gender" value="Male">Male

    <input type="radio" name="gender" value="Female">Female

    <br /><br />

    <input id="btnSubmit" type="submit" value="submit" />

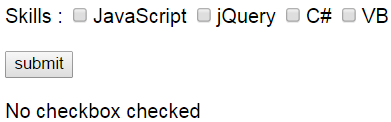
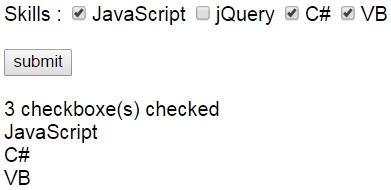
    <br /><br />

    <div id="divResult">

    </div>

</body>

</html>

**Selects all checked checkbox elements**  
$('input[type="checkbox"]:checked')  
  
**Example :** When you click the submit button without checking any checkbox, "No checkbox checked" message should be displayed.   
  
  
When you click the submit button after checking a checkbox, then a message stating the number of checkboxes checked and their values should be displayed.   


<html>

<head>

    <title></title>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            $('#btnSubmit').click(function () {

                var result = $('input[type="checkbox"]:checked');

                if (result.length > 0) {

                    var resultString = result.length + " checkboxe(s) checked<br/>";

                    result.each(function () {

                        resultString += $(this).val() + "<br/>";

                    });

                    $('#divResult').html(resultString);

                }

                else {

                    $('#divResult').html("No checkbox checked");

                }

            });

        });

    </script>

</head>

<body style="font-family:Arial">

    Skills :

    <input type="checkbox" name="skills" value="JavaScript" />JavaScript

    <input type="checkbox" name="skills" value="jQuery" />jQuery

    <input type="checkbox" name="skills" value="C#" />C#

    <input type="checkbox" name="skills" value="VB" />VB

    <br /><br />

    <input id="btnSubmit" type="submit" value="submit" />

    <br /><br />

    <div id="divResult">

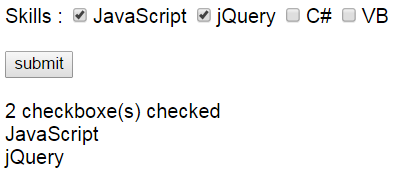
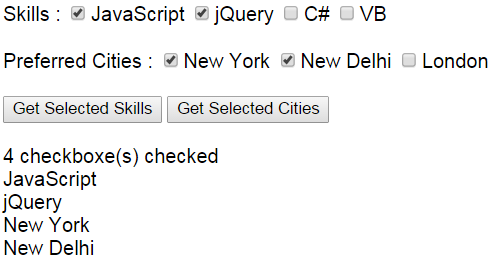
    </div>

</body>

</html>

### 12. Select values of checkbox group with jquery

In this video we will discuss, how to **select values of checked checkboxes that are in different groups using jQuery**. Along the way, we will also discuss **how to pass a variable to jquery selector**.

This is continuation to [Part 11](http://csharp-video-tutorials.blogspot.com/2015/03/jquery-checked-selector.html), please watch [Part 11](http://csharp-video-tutorials.blogspot.com/2015/03/jquery-checked-selector.html) from [jQuery tutorial](https://www.youtube.com/playlist?list=PL6n9fhu94yhVDV697uvHpavA3K_eWGQap) before proceeding.   
  
If you have just one group of checkboxes on your page, to get all the checked checkboxes you can use $('input[type="checkbox"]:checked').    
  
  
However, if you have more than one checkbox group, then $('input[type="checkbox"] :checked') is going to select all checked checkboxes from both the checkbox groups.   
  
  
If you prefer to get checked checkboxes from a specific checkbox group, depending on which button you have clicked, you can use $('input[name="skills"]:checked') or  $('input[name="cities"]:checked'). This will ensure that the checked checkboxes from only the skills or cities checkbox group are selected.

<html>

<head>

    <title></title>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            $('#btnGetSelectedSkills').click(function () {

                getSelectedCheckBoxes('skills');

            });

            $('#btnGetSelectedCities').click(function () {

                getSelectedCheckBoxes('cities');

            });

            var getSelectedCheckBoxes = function (groupName) {

                var result = $('input[name="' + groupName + '"]:checked');

                if (result.length > 0) {

                    var resultString = result.length + " checkboxe(s) checked<br/>";

                    result.each(function () {

                        resultString += $(this).val() + "<br/>";

                    });

                    $('#divResult').html(resultString);

                }

                else {

                    $('#divResult').html("No checkbox checked");

                }

            };

        });

    </script>

</head>

<body style="font-family:Arial">

    Skills :

    <input type="checkbox" name="skills" value="JavaScript" />JavaScript

    <input type="checkbox" name="skills" value="jQuery" />jQuery

    <input type="checkbox" name="skills" value="C#" />C#

    <input type="checkbox" name="skills" value="VB" />VB

    <br /><br />

    Preferred Cities :

    <input type="checkbox" name="cities" value="New York" />New York

    <input type="checkbox" name="cities" value="New Delhi" />New Delhi

    <input type="checkbox" name="cities" value="London" />London

    <br /><br />

    <input id="btnGetSelectedSkills" type="submit" value="Get Selected Skills" />

    <input id="btnGetSelectedCities" type="submit" value="Get Selected Cities" />

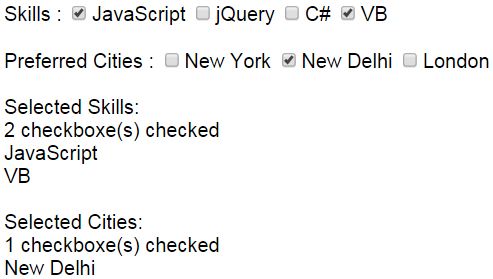
    <br /><br />

    <div id="divResult">

    </div>

</body>

</html>

At the moment to get the checked checkboxes values, we are using a button click event. You can also use the click event of the checkbox to do this.   


<html>

<head>

    <title></title>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            $('input[name="skills"]').click(function () {

                getSelectedCheckBoxes('skills');

            });

            $('input[name="cities"]').click(function () {

                getSelectedCheckBoxes('cities');

            });

            var getSelectedCheckBoxes = function (groupName) {

                var result = $('input[name="' + groupName + '"]:checked');

                if (result.length > 0) {

                    var resultString = result.length + " checkboxe(s) checked<br/>";

                    result.each(function () {

                        resultString += $(this).val() + "<br/>";

                    });

                    $('#div' + groupName).html(resultString);

                }

                else {

                    $('#div' + groupName).html("No checkbox checked");

                }

            };

        });

    </script>

</head>

<body style="font-family:Arial">

    Skills :

    <input type="checkbox" name="skills" value="JavaScript" />JavaScript

    <input type="checkbox" name="skills" value="jQuery" />jQuery

    <input type="checkbox" name="skills" value="C#" />C#

    <input type="checkbox" name="skills" value="VB" />VB

    <br /><br />

    Preferred Cities :

    <input type="checkbox" name="cities" value="New York" />New York

    <input type="checkbox" name="cities" value="New Delhi" />New Delhi

    <input type="checkbox" name="cities" value="London" />London

    <br /><br />

    Selected Skills:<br />

    <div id="divskills"></div>

    <br />

    Selected Cities:<br />

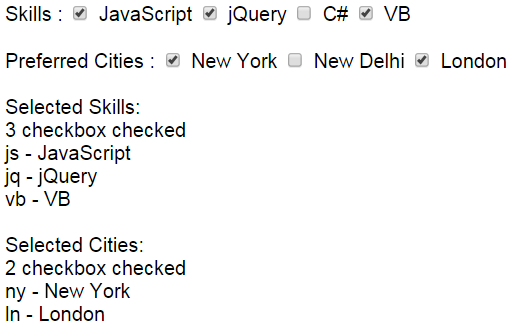
    <div id="divcities"></div>

</body>

</html>

### 13. jQuery get selected checkbox text

### In this video we will discuss how to get the checked checkbox text along with the value. This is continuation to [Part 12](http://csharp-video-tutorials.blogspot.com/2015/03/select-values-of-checkbox-group-with.html), please watch [Part 12](http://csharp-video-tutorials.blogspot.com/2015/03/select-values-of-checkbox-group-with.html) from [jQuery tutorial](https://www.youtube.com/playlist?list=PL6n9fhu94yhVDV697uvHpavA3K_eWGQap) before proceeding.

Here is what we want to do. As we check the checkboxes, we want to display the count of checkboxes checked and their text and values.   


<html>

<head>

    <title></title>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            $('input[name="skills"]').click(function () {

                getSelectedCheckBoxes('skills');

            });

            $('input[name="cities"]').click(function () {

                getSelectedCheckBoxes('cities');

            });

            var getSelectedCheckBoxes = function (groupName) {

                var result = $('input[name="' + groupName + '"]:checked');

                if (result.length > 0) {

                    var resultString = result.length + " checkbox checked<br/>";

                    result.each(function () {

                        var selectedValue = $(this).val();

                        resultString += selectedValue + " - "

                            + $('label[for="option-' + selectedValue + '"]').text() + "<br/>";

                    });

                    $('#div' + groupName).html(resultString);

                }

                else {

                    $('#div' + groupName).html("No checkbox checked");

                }

            };

        });

    </script>

</head>

<body style="font-family:Arial">

    Skills :

    <input type="checkbox" name="skills" value="js" />

    <label for="option-js">JavaScript</label>

    <input type="checkbox" name="skills" value="jq" />

    <label for="option-jq">jQuery</label>

    <input type="checkbox" name="skills" value="cs" />

    <label for="option-cs">C#</label>

    <input type="checkbox" name="skills" value="vb" />

    <label for="option-vb">VB</label>

    <br /><br />

    Preferred Cities :

    <input type="checkbox" name="cities" value="ny" />

    <label for="option-ny">New York</label>

    <input type="checkbox" name="cities" value="nd" />

    <label for="option-nd">New Delhi</label>

    <input type="checkbox" name="cities" value="ln" />

    <label for="option-ln">London</label>

    <br /><br />

    Selected Skills:<br />

    <div id="divskills"></div>

    <br />

    Selected Cities:<br />

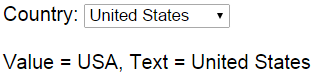
    <div id="divcities"></div>

</body>

</html>

### 13. jQuery selected selector

In this video we will discuss **jQuery :selected selector**.

To select all checked checkboxes or radio buttons, we use **:checked selector**. To select all selected options of a select element use **:selected selector**.   
  
**How to get selected option from single select dropdown in jquery** : We want to get the selected option text and value   


<html>

<head>

    <title></title>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            $('#selectCountries').change(function () {

                var selectedOption = $('#selectCountries option:selected');

                $('#divResult').html('Value = ' + selectedOption.val()

                    + ', Text = ' + selectedOption.text());

            });

        });

    </script>

</head>

<body style="font-family:Arial">

    Country:

    <select id="selectCountries">

        <option selected="selected" value="USA">United States</option>

        <option value="IND">India</option>

        <option value="UK">United Kingdom</option>

        <option value="CA">Canada</option>

        <option value="AU">Australia</option>

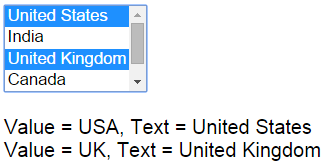
    </select>

    <br /><br />

    <div id="divResult"></div>

</body>

</html>

**How to get all selected options from multi-select dropdown in jquery** : We want to get all the selected options text and value.   


<html>

<head>

    <title></title>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            $('#selectCountries').change(function () {

                var selectedOptions = $('#selectCountries option:selected');

                if (selectedOptions.length > 0) {

                    var resultString = '';

                    selectedOptions.each(function () {

                        resultString += 'Value = ' + $(this).val() +

                            ', Text = ' + $(this).text() + '<br/>';

                    });

                    $('#divResult').html(resultString);

                }

            });

        });

    </script>

</head>

<body style="font-family:Arial">

    <select id="selectCountries" multiple="multiple">

        <option selected="selected" value="USA">United States</option>

        <option value="IND">India</option>

        <option value="UK">United Kingdom</option>

        <option value="CA">Canada</option>

        <option value="AU">Australia</option>

    </select>

    <br /><br />

    <div id="divResult"></div>

</body>

</html>

**Please note :** Hold down the CTRL key, to select more than one item.

### 15. jQuery disabled selector

In this video we will discuss  
1. **jQuery disabled selector**  
2. **jQuery enabled selector**  
3. **Where you can find jquery selectors documentation**

Selects all disabled elements

$(':disabled')

**Selects all disabled elements and sets a 3px solid red border**

<html>

<head>

    <title></title>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            $(':disabled').css('border', '3px solid red');

        });

    </script>

</head>

<body style="font-family:Arial">

    <table>

        <tr>

            <td>First Name </td>

            <td><input type="text" disabled="disabled" /></td>

        </tr>

        <tr>

            <td>Last Name </td>

            <td><input type="text" disabled="disabled" /></td>

        </tr>

        <tr>

            <td>Email </td>

            <td><input type="text" /></td>

        </tr>

        <tr>

            <td>Gender </td>

            <td>

                <select id="selectGender" disabled="disabled">

                    <option value="Male">Male</option>

                    <option value="Female">Female</option>

                </select>

            </td>

        </tr>

        <tr>

            <td>Comments</td>

            <td><textarea></textarea></td>

        </tr>

        <tr>

            <td colspan="2">

                <input type="submit" value="Submit" disabled="disabled" />

            </td>

        </tr>

    </table>

</body>

</html>

**Selects all input elements that are disabled and sets a 3px solid red border**

<script type="text/javascript">

    $(document).ready(function () {

        $('input:disabled').css('border', '3px solid red');

    });

</script>

**Selects all input elements with type=text that are disabled and sets a 3px solid red border**

<script type="text/javascript">

    $(document).ready(function () {

        $('input[type="text"]:disabled').css('border', '3px solid red');

    });

</script>

**Selects all enabled elements**

$(':enabled')

**Selects all enabled textarea elements and sets a 3px solid red border**

<script type="text/javascript">

    $(document).ready(function () {

        $('textarea:enabled').css('border', '3px solid red');

    });

</script>

### Where can you find jquery selectors documentation <https://api.jquery.com/category/selectors/>

### 16. jQuery each function

### In this video we will discuss  1. Use of jQuery each function 2. How to exit from each function in jQuery 3. Implicit iteration in jQuery 4. Performance considerations when using jquery each function

**jQuery each function is used to iterate over the items in a collection**. For each item in the collection the anonymous function is called. The index of the element and the element itself are passed to the anonymous function.

$('li').each(function (index, element) {

    alert(index + ' : ' + $(element).text());

});

To refer to the current element that we are iterating over you can also use this keyword.

$('li').each(function (index) {

    alert(index + ' : ' + $(this).text());

});

If you don't care about the index and just need the text of the list item, then you can get rid of the index parameter

$('li').each(function () {

    alert($(this).text());

});

**How to exit from each function in jQuery** : To exit from each function, return false.

<script type="text/javascript">

    $(document).ready(function () {

        $('li').each(function () {

            if ($(this).text() == 'UK') {

                return false;

            }

            alert($(this).text());

        });

    });

</script>

**Implicit iteration in jQuery :** The $('li') selector returns all list item elements. Notice that we are calling the css() jquery function on the jquery collection itself. In this case, css() method implicitly iterates over each element in the entire collection.

<html>

<head>

    <title></title>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            $('li').css('color', 'red');

        });

    </script>

</head>

<body style="font-family:Arial">

    <ul>

        <li>US</li>

        <li>India</li>

        <li>UK</li>

        <li>Canada</li>

        <li>Australia</li>

    </ul>

</body>

</html>

There is no need to explicitly iterate over each element in the collection. Most jQuery methods implicitly iterate over the entire collection.

<script type="text/javascript">

    $(document).ready(function () {

        $('li').each(function () {

            $(this).css('color', 'red');

        });

    });

</script>

**Performance considerations when using jquery each function**

<html>

<head>

    <title></title>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            $('li').each(function () {

                $('#divResult').html($('#divResult').html() + '<br/>' + $(this).text())

            });

        });

    </script>

</head>

<body style="font-family:Arial">

    <ul>

        <li>US</li>

        <li>India</li>

        <li>UK</li>

        <li>Canada</li>

        <li>Australia</li>

    </ul>

    <div id="divResult"></div>

</body>

</html>

**From a performance standpoint, there are 2 problems with the above code**  
1. jQuery needs to search the DOM for div element with id = divResult, for each list item in the collection. Since there are 5 list items, jquery searches the DOM 5 times for the same div element with id = divResult. The performance of the above code can be improved by caching the divResult element.  
  
2. The DOM element (div element with id = divResult) is being updated on each iteration. Again this is bad for performance. To improve the performance build a string variable with the required data on each iteration of the loop. Once the loop has completed, update the DOM element with the value that is present in the string varibale. This will ensure that the DOM element is updated only once and this is much better for performance.  
  
**The following is much better from a performance standpoint**

<script type="text/javascript">

    $(document).ready(function () {

        var result = '';

        $('li').each(function () {

            result += '<br/>' + $(this).text();

        });

        $('#divResult').html(result);

    });

</script>

### 17.JQuery Method Chaining

### In this video we will discuss  1. What is method chaining 2. How method chaining works 3. When will method chaining not work

Let us understand this with an example. Consider the following HTML

<ul>

    <li>US</li>

    <li>India</li>

    <li>UK</li>

    <li>Canada</li>

    <li>Australia</li>

</ul>

**The jQuery code should do the following**  
1. Change the color of all the list items to blue  
2. All the list items should slide down (animation)  
3. All the list items should slide up (animation)  
4. Change the title attribute of all the list items to MY TITLE  
  
**One way to do this is by using the following jQuery code**

<html>

<head>

    <title></title>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            $('li').css('color', 'blue');

            $('li').slideUp(1000);

            $('li').slideDown(1000);

            $('li').attr('title', 'MY TITLE');

        });

    </script>

</head>

<body style="font-family:Arial">

    <ul>

        <li>US</li>

        <li>India</li>

        <li>UK</li>

        <li>Canada</li>

        <li>Australia</li>

    </ul>

</body>

</html>

**Instead the jQuery methods can be chained as shown below**

<script type="text/javascript">

    $(document).ready(function () {

        $('li').css('color', 'blue').slideUp(1000).slideDown(1000).attr('title', 'MY TITLE');

    });

</script>

**How does method chaining work**  
1. $('li') returns a jquery collection object that contains all the elements that match the selector  
2. On the collection object css() method is called. This method applies the color to all the elements in the collection and returns the collection again.  
3. On the collection that is returned by the css() method, the slideUp() method is called.  
4. This process continues until we reach the last method in the chain.  
  
When chaining methods like this, the line of code will become quite long and the readability will be lost. To improve the readability of the code you can format chained methods as shown below.

<script type="text/javascript">

    $(document).ready(function () {

        $('li')

            .css('color', 'blue')

            .slideUp(1000)

            .slideDown(1000)

            .attr('title', 'MY TITLE');

    });

</script>

**When will method chaining not work**  
Method chaining will not work if a method in the chain does not return an object. In the example below, text() method returns a string that contains the text of all the list items and not an object. Hence the chaining does not work in this case.

<script type="text/javascript">

    $(document).ready(function () {

        $('li')

            .text()

            .css('color', 'blue')

            .slideUp(1000)

            .slideDown(1000)

            .attr('title', 'MY TITLE');

    });

</script>

**With the same example**, method chaining works, if you use text() method to set the value for the list item. In this case, text() method sets a value for each list item in the jQuery collection object and returns that collection object. Hence the chaining works.

<script type="text/javascript">

    $(document).ready(function () {

        $('li')

            .text('MY VALUE')

            .css('color', 'blue')

            .slideUp(1000)

            .slideDown(1000)

            .attr('title', 'MY TITLE');

    });

</script>

### 18.What is JSON

In this video we will discuss  
1. **What is JSON**  
2. **JSON Arrays**  
3. **Nested JSON object**

**What is JSON**  
JSON stands for JavaScript Object Notation. JSON is a lightweight data-interchange format. JSON is an easier-to-use alternative to XML.   
  
**Creating a JSON object** : Employee data can be stored in a JSON object as shown below.

var employeeJSON = {

    "firstName": "Todd",

    "lastName": "Grover",

    "gender": "Male",

    "salary": 50000

};

1. employeeJSON is a JSON object  
2. In the curly braces we include the "name": "value" pairs, separated by commas  
3. The name and value of a property are separated using a colon (:)  
4. You can declare any number of properties  
  
If you want to represent the same data using XML, you may have XML that would look as shown below.

<Employee>

  <firstName>Todd</firstName>

  <lastName>Grover</lastName>

  <gender>Male</gender>

  <salary>50000</salary>

</Employee>

Reading data from the JSON object : To read data from the JSON object, use the property names.  
var firstName = employeeJSON.firstName;  
  
**Creating and accessing data from a JSON object**

<html>

<head>

    <title></title>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            // Creating a JSON object

            var employeeJSON = {

                "firstName": "Todd",

                "lastName": "Grover",

                "gender": "Male",

                "salary": 50000

            };

            // Accessing data from a JSON object

            var result = '';

            result += 'First Name = ' + employeeJSON.firstName + '<br/>';

            result += 'Last Name = ' + employeeJSON.lastName + '<br/>';

            result += 'Gender = ' + employeeJSON.gender + '<br/>';

            result += 'Salary = ' + employeeJSON.salary + '<br/>';

            $('#resultDiv').html(result);

        });

    </script>

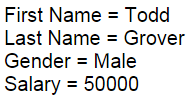
</head>

<body style="font-family:Arial">

    <div id="resultDiv"></div>

</body>

</html>

**Output :**   
  
  
**JSON Arrays :** What if you want to store more than one employee data in the JSON object. This is when JSON arrays can be used. A JSON array can contain multiple objects.  
  
To **create a JSON array**  
1. Wrap the objects in square brackets  
2. Each object must be separated with a comma  
  
**Creating a JSON array**

var employeesJSON = [

{

    "firstName": "Todd",

    "lastName": "Grover",

    "gender": "Male",

    "salary": 50000

},

{

    "firstName": "Sara",

    "lastName": "Baker",

    "gender": "Female",

    "salary": 40000

}];

**Reading from a JSON array :** To access the employee objects in the JSON array, use the object's index position.  
  
Retrieves the lastName of first employee object in the JSON array  
var result = employeesJSON[0].lastName;  
  
Retrieves the fistName of second employee object in the JSON array  
var result = employeesJSON[1].firstName;  
  
**Nested JSON object :** You can also store multiple employees in the JSON object by nesting them.  
  
**Nested JSON object example :**

var employeesJSON = {

    "Todd": {

        "firstName": "Todd",

        "lastName": "Grover",

        "gender": "Male",

        "salary": 50000

    },

    "Sara": {

        "firstName": "Sara",

        "lastName": "Baker",

        "gender": "Female",

        "salary": 40000

    }

};

Retrieves the gender of employee Todd  
var result = employeesJSON.Todd.gender;  
  
Retrieves the salary of employee Sara  
var result = employeesJSON.Sara.salary;  
  
In our upcoming videos we will discuss where we could use JSON formatted data.

### 19. Convert JSON object to string

In this video we will discuss  
1. **How to convert JSON object to string**  
2. **How to convert string to JSON object**

This is continuation [Part 18](http://csharp-video-tutorials.blogspot.com/2015/04/what-is-json.html). Please watch [Part 18](http://csharp-video-tutorials.blogspot.com/2015/04/what-is-json.html) from [jQuery tutorial](https://www.youtube.com/playlist?list=PL6n9fhu94yhVDV697uvHpavA3K_eWGQap) before proceeding.   
  
The following example **converts JSON array to a string**. JSON.stringify() method converts a JSON object (or array) into a JSON string.

<html>

<head>

    <title></title>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            var employeesJSON = [

                {

                    "firstName": "Todd",

                    "lastName": "Grover",

                    "gender": "Male",

                    "salary": 50000

                },

                {

                    "firstName": "Sara",

                    "lastName": "Baker",

                    "gender": "Female",

                    "salary": 40000

                }

            ];

            var JSONString = JSON.stringify(employeesJSON);

            $('#resultDiv').html(JSONString);

        });

    </script>

</head>

<body style="font-family:Arial">

    <div id="resultDiv"></div>

</body>

</html>

**Output :**  
[{"firstName":"Todd","lastName":"Grover","gender":"Male","salary":50000},  
{"firstName":"Sara","lastName":"Baker","gender":"Female","salary":40000}]  
  
The following example **converts a JSON string to a JSON array**. JSON.parse() method converts a JSON string to JSON array. We then use the jQuery each() method to loop thru each employee JSON object and retrieve the respective property values.

<html>

<head>

    <title></title>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            var JSONString = '[{ "firstName": "Todd", "lastName": "Grover", "gender": "Male", "salary": 50000 }, { "firstName": "Sara", "lastName": "Baker", "gender": "Female", "salary": 40000 }]';

            var employeesJSON = JSON.parse(JSONString);

            var result = '';

            $.each(employeesJSON, function (i, item) {

                result += 'First Name = ' + item.firstName + '<br/>';

                result += 'Last Name = ' + item.lastName + '<br/>';

                result += 'Gender = ' + item.gender + '<br/>';

                result += 'Salary = ' + item.salary + '<br/><br/>';

            });

            $('#resultDiv').html(result);

        });

    </script>

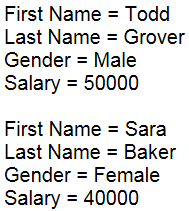
</head>

<body style="font-family:Arial">

    <div id="resultDiv"></div>

</body>

</html>

**Output :**   


### 20. Convert JSON string to .net object

In this video we will discuss  
1. **How to convert .NET object to JSON string**  
2. **How to convert a JSON string to .NET object**

We will be using the following **Employee** class

public class Employee

{

    public string firstName { get; set; }

    public string lastName { get; set; }

    public string gender { get; set; }

    public int salary { get; set; }

}

The following example converts List<Employee> objects to a JSON string. **Serialize**() method of **JavaScriptSerializer** class **converts a .NET object to a JSON string**. **JavaScriptSerializer** class is present in **System.Web.Script.Serialization** namespace.

using System;

using System.Collections.Generic;

using System.Web.Script.Serialization;

namespace Demo

{

    public partial class WebForm1 : System.Web.UI.Page

    {

        protected void Page\_Load(object sender, EventArgs e)

        {

            Employee employee1 = new Employee

            {

                firstName = "Todd",

                lastName = "Grover",

                gender = "Male",

                salary = 50000

            };

            Employee employee2 = new Employee

            {

                firstName = "Sara",

                lastName = "Baker",

                gender = "Female",

                salary = 40000

            };

            List<Employee> listEmployee = new List<Employee>();

            listEmployee.Add(employee1);

            listEmployee.Add(employee2);

            JavaScriptSerializer javaScriptSerializer = new JavaScriptSerializer();

            string JSONString = javaScriptSerializer.Serialize(listEmployee);

            Response.Write(JSONString);

        }

    }

}

**Output :**  
[{"firstName":"Todd","lastName":"Grover","gender":"Male","salary":50000},{"firstName":"Sara","lastName":"Baker","gender":"Female","salary":40000}]   
  
The following example converts a JSON string to List<Employee> objects. **Deserialize()** method of **JavaScriptSerializer** class **converts a JSON string to a .NET object**.

string jsonString = "[{\"firstName\":\"Todd\",\"lastName\":\"Grover\",\"gender\":\"Male\",\"salary\":50000},{\"firstName\":\"Sara\",\"lastName\":\"Baker\",\"gender\":\"Female\",\"salary\":40000}]";

JavaScriptSerializer javaScriptSerializer = new JavaScriptSerializer();

List<Employee> employees = (List<Employee>)javaScriptSerializer.Deserialize(jsonString, typeof(List<Employee>));

foreach (Employee employee in employees)

{

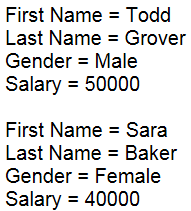
    Response.Write("First Name = " + employee.firstName + "<br/>");

    Response.Write("Last Name = " + employee.lastName + "<br/>");

    Response.Write("Gender = " + employee.gender + "<br/>");

    Response.Write("Salary = " + employee.salary + "<br/><br/>");

}

Output :    


### 21. jQuery DOM manipulation methods

In this video we will discuss  
1. What are jQuery DOM manipulation methods  
2. How to set attribute values using jQuery attr() method  
3. How to retrieve attribute values using jQuery attr() method  
4. How to set multiple attribute values using jQuery attr() method  
5. How to remove an attribute using jQuery removeAttr() method

**What are jQuery DOM manipulation methods**  
jQuery DOM manipulation methods manipulate the DOM in some manner. The complete list of jQuery DOM manipulation methods can be found at the following link.  
<http://api.jquery.com/category/manipulation/>  
  
jQuery attr method is used to set or retrieve attribute values of html elements.  
  
Retrieves the title attribute value of the first matching element  
$('div').attr('title')  
  
**Example :** In this example we have 2 DIV elements. Since attr() function retrieves only the attribute value of first matching element, we only get the title attribute value of the first DIV element.

<html>

<head>

    <title></title>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            alert($('div').attr('title'));

        });

    </script>

</head>

<body style="font-family:Arial">

    <div id="div1" title="My DIV1">

        DIV 1

    </div>

    <div id="div2" title="My DIV2">

        DIV 2

    </div>

</body>

</html>

To retrieve the title attribute value of all the matching elements, jQuery each() method can be used.

<script type="text/javascript">

    $(document).ready(function () {

        $('div').each(function () {

            alert($(this).attr('title'));

        });

    });

</script>

Sets the title attribute value of all the matching elements to "new div title"  
$('div').attr('title', 'new div title');  
  
**Example :** If you want to set the same value for the title attribute of all the elements, then there is no need to loop thru each element. All the div elements in this case will have 'new div title' as the title attribute value.

<script type="text/javascript">

    $(document).ready(function () {

        // set the same title attribute value for all the DIV elements

        $('div').attr('title', 'new div title');

        // Retrieve and alert the title attribute value of all DIV elements

        $('div').each(function () {

            alert($(this).attr('title'));

        });

    });

</script>

If you want to set a different value for the title attribute, then you may need to loop thru each element.

<script type="text/javascript">

    $(document).ready(function () {

        // sets a different title attribute value for each DIV element

        $('div').each(function (i) {

            $(this).attr('title', 'div ' + (i + 1) + ' title');

        });

        // Retrieve and alert the title attribute value of all DIV elements

        $('div').each(function () {

            alert($(this).attr('title'));

        });

    });

</script>

**How to set multiple attribute values :** Using a JSON object to set multiple attribute values

<html>

<head>

    <title></title>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            var config = {

                'title': 'New Div Title',

                'style': 'border:3px solid red;',

                'name': 'My Div'

            };

            $('div').attr(config);

            var result = '';

            $('div').each(function () {

                result += 'title = ' + $(this).attr('title') + '<br/>';

                result += 'style = ' + $(this).attr('style') + '<br/>';

                result += 'name = ' + $(this).attr('name') + '<br/><br/>';

            });

            $('#resultSpan').html(result);

        });

    </script>

</head>

<body style="font-family:Arial">

    <div id="div1" title="My DIV1">

        DIV 1

    </div>

    <br />

    <div id="div2" title="My DIV2">

        DIV 2

    </div>

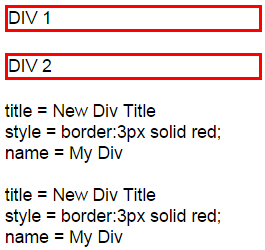
    <br />

    <span id="resultSpan">

    </span>

</body>

</html>

Output :   


### 22. jQuery wrap elements

In this video we will discuss **how to wrap and unwrap elements**

The following jquery methods are used to wrap and unwrap elements. Since these methods modify DOM, they belong to DOM manipulation category.  
**wrap**  
**unwrap**  
**wrapAll**  
**wrapInner**  
  
**wrap -** Wrap an HTML structure around each element in the set of matched elements.  
  
**Consider the following HTML**

<div id="div1">

    DIV 1

</div>

<div id="div2">

    DIV 2

</div>

<div id="div3">

    DIV 3

</div>

The following line of code wraps each of the above div element with another div element.

$('div').wrap('<div class="containerDiv"></div>');

So the HTML in the DOM would now look as shown below. To view the DOM HTML use the browser developer tools.

<div class="containerDiv">

    <div id="div1">

        DIV 1

    </div>

</div>

<div class="containerDiv">

    <div id="div2">

        DIV 2

    </div>

</div>

<div class="containerDiv">

    <div id="div3">

        DIV 3

    </div>

</div>

**Example :**

<html>

<head>

    <title></title>

    <style>

        .containerDiv {

            background-color: red;

            color: white;

            font-weight: bold;

            margin: 5px;

        }

    </style>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            alert($('div.containerDiv').length);

            $('div').wrap('<div class="containerDiv"></div>');

            alert($('div.containerDiv').length);

        });

    </script>

</head>

<body style="font-family:Arial">

    <div id="div1">

        DIV 1

    </div>

    <div id="div2">

        DIV 2

    </div>

    <div id="div3">

        DIV 3

    </div>

</body>

</html>

Output :    
  
  
**unwrap** - Remove the parents of the set of matched elements from the DOM.  
  
**Example :**

<html>

<head>

    <title></title>

    <style>

        .containerDiv {

            background-color: red;

            color: white;

            font-weight: bold;

            margin: 5px;

        }

    </style>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            alert($('div.containerDiv').length);

            $('div').wrap('<div class="containerDiv"></div>');

            alert($('div.containerDiv').length);

            $('div').unwrap();

            alert($('div.containerDiv').length);

        });

    </script>

</head>

<body style="font-family:Arial">

    <div id="div1">

        DIV 1

    </div>

    <div id="div2">

        DIV 2

    </div>

    <div id="div3">

        DIV 3

    </div>

</body>

</html>

**wrapAll** - Wrap an HTML structure around all elements in the set of matched elements.  
  
Consider the following HTML

<div id="div1">

    DIV 1

</div>

<div id="div2">

    DIV 2

</div>

<div id="div3">

    DIV 3

</div>

The following line of code wraps all of the matching div element with another div element.

$('div').wrapAll('<div class="containerDiv"></div>');

So the HTML in the DOM would now look as shown below.

<div class="containerDiv">

    <div id="div1">

        DIV 1

    </div><div id="div2">

        DIV 2

    </div><div id="div3">

        DIV 3

    </div>

</div>

**Example :**

<html>

<head>

    <title></title>

    <style>

        .containerDiv {

            background-color: red;

            color: white;

            font-weight: bold;

            margin: 5px;

        }

    </style>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            alert($('div.containerDiv').length);

            $('div').wrapAll('<div class="containerDiv"></div>');

            alert($('div.containerDiv').length);

            $('div').unwrap();

            alert($('div.containerDiv').length);

        });

    </script>

</head>

<body style="font-family:Arial">

    <div id="div1">

        DIV 1

    </div>

    <div id="div2">

        DIV 2

    </div>

    <div id="div3">

        DIV 3

    </div>

</body>

</html>

**wrapInner** - Wrap an HTML structure around the content of each element in the set of matched elements.  
  
Consider the following HTML

<div id="div1">

    DIV 1

</div>

<div id="div2">

    DIV 2

</div>

<div id="div3">

    DIV 3

</div>

The following line of code wraps each of the above div element content with another div element.

$('div').wrapInner('<div class="containerDiv"></div>');

So the HTML in the DOM would now look as shown below.

<div id="div1">

    <div class="containerDiv">

        DIV 1

    </div>

</div>

<div id="div2">

    <div class="containerDiv">

        DIV 2

    </div>

</div>

<div id="div3">

    <div class="containerDiv">

        DIV 3

    </div>

</div>

Example :

<html>

<head>

    <title></title>

    <style>

        .containerDiv {

            background-color: red;

            color: white;

            font-weight: bold;

            margin: 5px;

        }

    </style>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            $('div').wrapInner('<div class="containerDiv"></div>');

        });

    </script>

</head>

<body style="font-family:Arial">

    <div id="div1">

        DIV 1

    </div>

    <div id="div2">

        DIV 2

    </div>

    <div id="div3">

        DIV 3

    </div>

</body>

</html>

### 23. jQuery append elements

To **append elements** we have  
append()  
appendTo()  
  
To **prepend elements** we have  
prepend()  
prependTo()  
  
Since these methods modify DOM, they belong to DOM manipulation category.  
  
**jquery append example** : The following example appends the specified HTML to all div elements

<html>

<head>

    <title></title>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            $('div').append('<b> Tutorial</b>');

        });

    </script>

</head>

<body style="font-family:Arial">

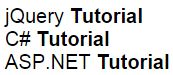
    <div id="div1">jQuery</div>

    <div id="div2">C#</div>

    <div id="div3">ASP.NET</div>

</body>

</html>

Output :   
   
  
**jquery appendto example** : The above example can be rewritten using appendTo as shown below.

<html>

<head>

    <title></title>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            $('<b> Tutorial</b>').appendTo('div');

        });

    </script>

</head>

<body style="font-family:Arial">

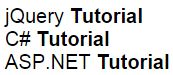
    <div id="div1">jQuery</div>

    <div id="div2">C#</div>

    <div id="div3">ASP.NET</div>

</body>

</html>

Output :   
   
  
**What is the difference between append and appendTo**  
Both these methods perform the same task. The only difference is in the syntax. With append method we first specify the target elements and then the content that we want to append, where as we do the opposite with appendTo method.  
  
**jquery prepend example** : The following example prepends the specified HTML to all div elements

<html>

<head>

    <title></title>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            $('div').prepend('<b>Tutorial </b>');

        });

    </script>

</head>

<body style="font-family:Arial">

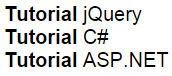
    <div id="div1">jQuery</div>

    <div id="div2">C#</div>

    <div id="div3">ASP.NET</div>

</body>

</html>

Output :   
   
  
**jquery prependTo example** : The above example can be rewritten using prependTo as shown below.

<html>

<head>

    <title></title>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            $('<b>Tutorial </b>').prependTo('div');

        });

    </script>

</head>

<body style="font-family:Arial">

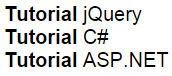
    <div id="div1">jQuery</div>

    <div id="div2">C#</div>

    <div id="div3">ASP.NET</div>

</body>

</html>

Output :   
   
  
**What is the difference between prepend and prependTo**  
Both these methods perform the same task. The only difference is in the syntax. With prepend method we first specify the target elements and then the content that we want to prepend, where as we do the opposite with prependTo method.  
  
**jQuery append existing element example** : These methods (append, appendTo, prepend, prependTo) can also select an element on the page and insert it into another

<html>

<head>

    <title></title>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            $('div').append($('#mySpan'));

        });

    </script>

</head>

<body style="font-family:Arial">

    <span id="mySpan"> Programming</span>

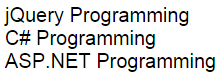
    <div id="div1">jQuery</div>

    <div id="div2">C#</div>

    <div id="div3">ASP.NET</div>

</body>

</html>

Output :    
   
  
**Difference between prepend and append**  
prepend method, inserts the specified content to the beginning of each element in the set of matched elements, where as append method inserts the specified content to the end of each element in the set of matched elements.

**24.JQuery Insert Element Before and After**

To insert an element before another element  
**before**  
**insertBefore**  
  
To insert an element after another element  
**after**  
**inserAfter**  
  
Since these methods modify DOM, they belong to DOM manipulation category.  
  
**jquery before example**  
  
Consider the following HTML

<span>Training Courses</span>

<div>jQuery</div>

<div>C#</div>

<div>ASP.NET</div>

The following line of code inserts h3 element before each of the div elements  
$('div').before('<h3>Programming</h3>');  
  
So the HTML in the DOM would now look as shown below. Notice that h3 element is added before every div element

<span>Training Courses</span>

<h3>Programming</h3><div>jQuery</div>

<h3>Programming</h3><div>C#</div>

<h3>Programming</h3><div>ASP.NET</div>

**jquery insertbefore example :** insertbefore method methods perform the same task as before. The only difference is in the syntax. With before method we first specify the target elements and then the content that we want to insert, where as we do the opposite with insertbefore method.  
  
$('<h3>Programming</h3>').insertBefore('div');  
  
**jquery after example**  
  
Consider the following HTML

<span>Training Courses</span>

<div>jQuery</div>

<div>C#</div>

<div>ASP.NET</div>

The following line of code inserts h3 element after each of the div elements  
$('div').after('<h3>Programming</h3>');  
  
So the HTML in the DOM would now look as shown below. Notice that h3 element is added after every div element

<span>Training Courses</span>

<div>jQuery</div><h3>Programming</h3>

<div>C#</div><h3>Programming</h3>

<div>ASP.NET</div><h3>Programming</h3>

**jquery insertafter example :** insertafter method methods perform the same task as after. The only difference is in the syntax. With after method we first specify the target elements and then the content that we want to insert, where as we do the opposite with insertafter method.  
  
$('<h3>Programming</h3>').insertAfter('div');  
  
**jquery insert existing element before or after another element :** These methods (before, insertBefore, after, inserAfter) can also select an existing element on the page and insert it before or after another element.  
  
Consider the following HTML

<span>Training Courses</span>

<div>jQuery</div><h3>Programming</h3>

<div>C#</div><h3>Programming</h3>

<div>ASP.NET</div><h3>Programming</h3>

The following line of code inserts span element after each of the div elements  
$('div').after($('span'));  
  
So the HTML in the DOM would now look as shown below.

<div>jQuery</div><span>Training Course</span>

<div>C#</div><span>Training Course</span>

<div>ASP.NET</div><span>Training Course</span>

### 25. jQuery add or remove class

|  |  |
| --- | --- |
| **hasClass** | Returns true if an element has the specified class otherwise false |
| **addClass** | Adds one or more specified classes. To add multiple classes separate them with a space. |
| **removeClass** | Removes one or multiple or all classes. To remove multiple classes separate them with a space. To remove all classes, don't specify any class name. |
| **toggleClass** | Toggles one or more specified classes. If the element has the specified class then it is removed, if the class is not present then it is added. |

<html>

<head>

    <title></title>

    <script src="jquery-1.11.2.js"></script>

    <style>

        .boldClass {

            font-weight: bold;

        }

        .italicsClass {

            font-style: italic;

        }

        .colorClass {

            color: red;

        }

    </style>

    <script type="text/javascript">

        $(document).ready(function () {

            $('#btn1').click(function () {

                $('p').addClass('colorClass');

            });

            $('#btn2').click(function () {

                $('p').removeClass('colorClass');

            });

            $('#btn3').click(function () {

                $('p').addClass('colorClass italicsClass');

            });

            $('#btn4').click(function () {

                $('p').removeClass('colorClass italicsClass');

            });

            $('#btn5').click(function () {

                $('p').addClass('colorClass italicsClass boldClass');

            });

            $('#btn6').click(function () {

                $('p').removeClass();

            });

        });

    </script>

</head>

<body style="font-family:Arial">

    <p>Pragim Technologies</p>

    <table>

        <tr>

            <td>

                <input id="btn1" style="width:250px" type="button"

                       value="Add Color Class" />

            </td>

            <td>

                <input id="btn2" style="width:250px" type="button"

                       value="Remove Color Class" />

            </td>

        </tr>

        <tr>

            <td>

                <input id="btn3" style="width:250px" type="button"

                       value="Add Color and Italics Classes" />

            </td>

            <td>

                <input id="btn4" style="width:250px" type="button"

                       value="Remove Color and Italics Classes" />

            </td>

        </tr>

        <tr>

            <td>

                <input id="btn5" style="width:250px" type="button"

                       value="Add Color, Italics & Bold Classes" />

            </td>

            <td>

                <input id="btn6" style="width:250px" type="button"

                       value="Remove All Classes" />

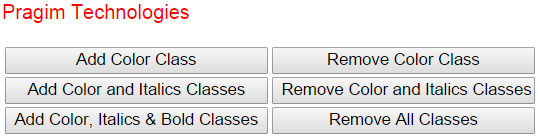
            </td>

        </tr>

    </table>

</body>

</html>

**Output :**   
 

<html>

<head>

    <title></title>

    <script src="jquery-1.11.2.js"></script>

    <style>

        .textBoxStyle {

            background-color: green;

            color: white;

        }

    </style>

    <script type="text/javascript">

        $(document).ready(function () {

            $('input[type="text"]').each(function () {

                $(this).focus(function () {

                    $(this).addClass('textBoxStyle');

                });

                $(this).blur(function () {

                    $(this).removeClass('textBoxStyle');

                });

            });

        });

    </script>

</head>

<body style="font-family:Arial">

    <table>

        <tr>

            <td>First Name</td>

            <td><input type="text" /></td>

        </tr>

        <tr>

            <td>Last Name</td>

            <td><input type="text" /></td>

        </tr>

        <tr>

            <td>Gender</td>

            <td>

                <select>

                    <option value="select">Please Select</option>

                    <option value="male">Male</option>

                    <option value="female">Female</option>

                </select>

            </td>

        </tr>

        <tr>

            <td>Email</td>

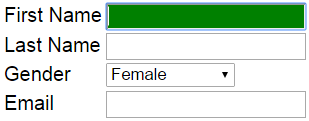
            <td><input type="text" /></td>

        </tr>

    </table>

</body>

</html>

**Output :**   


### 26.Difference between $.each and .each

**Please note :** $ is shortcut for jQuery.  
  
**What is the difference**  
.each is used to iterate over the items in a jQuery collection   
  
**Example :**In this example, the jQuery selector $('li') returns a jQuery object collection. So to loop thru the objects in the jQuery collection we are using .each() method. Notice this method is called on the jQuery object collection, that the selector returns, so the .each() method knows the items it has to iterate over.   
  
**The callback method has 2 parameters**  
1) index - The index of the element   
2) element - The DOM element that we are currently iterating over   
  
Since the element is a raw DOM object, to use use jQuery methods you have to wrap it using the jQuery wrapper $ as shown below.  
$(element)  
  
Instead of using element parameter, you can also use this keyword, which also refers to the raw DOM element that we are currently iterating over. To use jQuery methods you have to wrap it using the jQuery wrapper $ as shown below.  
$(this) 

<html>

<head>

    <title></title>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            var result = '';

            $('li').each(function (index, element) {

                result += 'Index = ' + index + ', Value = ' + $(element).text() + '<br/>';

                //or

                //result += 'Index = ' + index + ', Value = ' + $(this).text() + '<br/>';

            });

            $('#resultDiv').html(result);

        });

    </script>

</head>

<body style="font-family:Arial">

    <ul>

        <li>US</li>

        <li>India</li>

        <li>UK</li>

        <li>Canada</li>

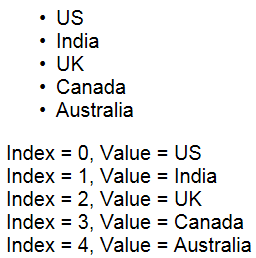
        <li>Australia</li>

    </ul>

    <div id="resultDiv"></div>

</body>

</html>

**Output :**   
   
  
**$.each() or jQuery.each() is used for iterating over javascript objects and arrays.**   
  
**Example :**In the example below intArray is a JavaScript array. So to loop thru the elements in the array we are using $.each() function. Notice this function has 2 parameters  
  
1) The JavaScript object or array that we want to iterate over  
2) The callback function that will execute on each iteration   
  
If you want to use .each() instead of $.each() function on the intArray object, wrap it using the jQuery wrapper $ as shown below. 

$(intArray).each(function (index, element) {

    result += 'Index = ' + index + ', Value = ' + element + '<br />';

})

<html>

<head>

    <title></title>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            var intArray = [100, 200, 300, 400, 500];

            var result = '';

            $.each(intArray, function (index, element) {

                result += 'Index = ' + index + ', Value = ' + element + '<br/>';

            });

            $('#resultDiv').html(result);

        });

    </script>

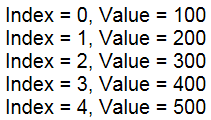
</head>

<body style="font-family:Arial">

    <div id="resultDiv"></div>

</body>

</html>

**Output :**   
   
  
**Using $.each() to iterate over a JavaScript object :**In this example, jsObject is a JavaScript object. In this example, $.each()  is used to iterate over the JavaScript object properties. 

<html>

<head>

    <title></title>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            var jsObject = {

                'USA': 'Washington D.C.',

                'India': 'New Delhi',

                'UK': 'London',

                'Australia': 'Canberra'

            };

            var result = '';

            $.each(jsObject, function (key, value) {

                result += key + ' - ' + value + '<br/>';

            });

            $('#resultDiv').html(result);

        });

    </script>

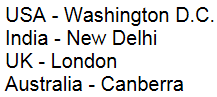
</head>

<body style="font-family:Arial">

    <div id="resultDiv"></div>

</body>

</html>

**Output :**   
   
  
In summary, **.each() method is used to iterate over the items in a jQuery collection where as $.each() method is used to iterate over javascript objects or arrays.**

### 27. jQuery map method

Just like jquery **each**() method, **map**() method is also used to iterate over matched elements.    
  
However, there are some differences between **map**() and **each**() methods which we will discuss in our next video.  
  
In general, if you want to create an array or concatenated string based on all matched elements in a jQuery selector, it is better to use map() over each() method.  
  
**Consider the following HTML**

<ul>

    <li>US</li>

    <li>India</li>

    <li>UK</li>

    <li>Canada</li>

    <li>Australia</li>

</ul>

To create an array of list item text values, we could use either map() or each() methods.   
  
**Using each() method**

$(document).ready(function () {

    var result = [];

    $('li').each(function (index, element) {

        result.push($(element).text());

    });

    alert(result);

});

**Using map() method**

$(document).ready(function () {

    alert($('li').map(function (index, element) {

        return $(element).text();

    }).get());

});

To create a pipe delimited string of all list item text values, we could use either map() or each() methods. The output should be as shown below.  
**US|India|UK|Canada|Australia**  
  
**Using each() method**

$(document).ready(function () {

    var result = '';

    $('li').each(function (index, element) {

        result += $(element).text() + "|";

    });

    result = result.substr(0, result.length - 1);

    alert(result);

});

**Using map() method**

$(document).ready(function () {

    alert($('li').map(function (index, element) {

        return $(element).text();

    }).get().join('|'));

});

### 28. Difference between each and map in jquery

|  |  |
| --- | --- |
| **$.map** | **$.each** |
| map method can be used as an iterator. | each method is an immutable iterator |
| Returns a new array | Returns the original array |
| The order of callback arguments - element, index. $.map(elems, function () { element, index }, arg) | The order of callback arguments - index, element. $.each(elems, function () { index, element }, arg) |
| Does not have a way to terminate the iteration | Return false to terminate the iteration |

**Example :** Notice that the callback arguments in the each method are the reverse of the callback arguments in the map function. Also notice that map returns a new array where as each method returns the original array. This proves the point that each method is an immutable iterator where as map is not.

$(document).ready(function () {

    var intArray = [1, 2, 3, 4, 5];

    function functionA(index, element) {

        return element \* 5;

    }

    function functionB(element, index) {

        return element \* 5;

    }

    var result1 = $.each(intArray, functionA);

    var result2 = $.map(intArray, functionB);

    document.write('each = ' + result1);

    document.write('<br/>')

    document.write('map = ' + result2);

});

**Output :**   
Difference between each and map in jquery   
  
**Example :** Notice that each method terminates the iteration when the element value is 3. The values 3, 4 and 5 are not written to the document. With map method we are not able to break the iteration. When the element value is 3, map method returns false and then continues writing 4 and 5 to the document.

$(document).ready(function () {

    var intArray = [1, 2, 3, 4, 5];

    $.each(intArray, function (index, element) {

        if (element == 3)

            return false;

        document.write(element + ',');

    });

    document.write('<br/>');

    $.map(intArray, function (element, index) {

        if (element == 3)

            return false;

        document.write(element + ',');

    });

});

**Output :**   
jquery map vs each

### 29.jQuery change event

**change event is fired when an element value changes**. All the following elements fire this event  
1. input   
2. textarea   
3. select   
  
**select, radio buttons** and **checkboxes**fire the change event as soon as a selection is made, where as the other element types (**textboxes & textarea**) wait until they loose focus.   
  
Consider the following HTML

<table>

    <tr>

        <td>First Name</td>

        <td>

            <input id="txtFirstName" type="text" class="inputRequired" />

        </td>

    </tr>

    <tr>

        <td>Last Name</td>

        <td>

            <input id="txtLastName" type="text" class="inputRequired" />

        </td>

    </tr>

    <tr>

        <td>City</td>

        <td>

            <select id="ddlCity" class="inputRequired">

                <option value="Select">Select</option>

                <option value="New York">New York</option>

                <option value="London">London</option>

                <option value="Chennai">Chennai</option>

                <option value="Sydney">Sydney</option>

            </select>

        </td>

    </tr>

    <tr>

        <td>Favourite Color</td>

        <td>

            <input id="radioRed" name="color" type="radio" value="Red"

                    class="inputRequired" />Red

            <input id="radioGreen" name="color" type="radio" value="Green"

                    class="inputRequired" />Green

            <input id="radioBlue" name="color" type="radio" value="Blue"

                    class="inputRequired" />Blue

        </td>

    </tr>

    <tr>

        <td>Contact Method</td>

        <td>

            <input id="chkBoxEmail" type="checkbox" value="Email"

                    class="inputRequired" />Email

            <input id="chkBoxPhone" type="checkbox" value="Phone"

                    class="inputRequired" />Phone

            <input id="chkBoxSocialMedia" type="checkbox" value="Social Media"

                    class="inputRequired" />Social Media

        </td>

    </tr>

    <tr>

        <td>

            Comments

        </td>

        <td>

            <textarea id="txtComments" class="inputRequired"></textarea>

        </td>

    </tr>

    <tr>

        <td>

            <input id="btnSubmit" type="button" value="Submit" />

        </td>

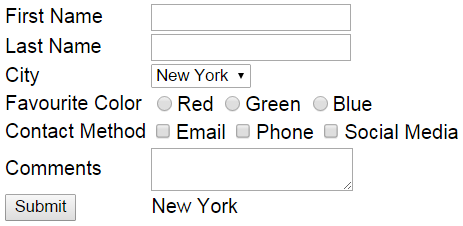
        <td>

            <div id="divResult"></div>

        </td>

    </tr>

</table>

As soon as the selection in the dropdownlist changes, we want to **handle the change event** and display the selected value in the div element with id="divResult". In the example below, we are using the id selector, so only the select element change event is handled.   
  
 

$(document).ready(function () {

    $('#ddlCity').change(function () {

        var selectedValue = $(this).val();

        if (selectedValue == "Select")

            selectedValue = "Please select city";

        $('#divResult').html(selectedValue);

    });

});

The following code handles the change event of all the input elements (**textbox, radio button, checkbox**). Notice that in this example we are using the jquery element selector. Change event of **select**and **textarea**is not handled.

$(document).ready(function () {

    var result = '';

    $('input').change(function () {

        if (result == '') {

            result += $(this).val();

        }

        else {

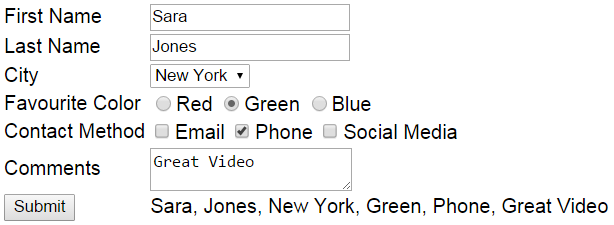
            result += ', ' + $(this).val();

        }

        $('#divResult').html(result);

    });

});

The following code handles the change event of all the elements on the page. Notice that in this example we are using the **jquery class selector**.   
  


$(document).ready(function () {

    var result = '';

    $('.inputRequired').change(function () {

        if (result == '') {

            result += $(this).val();

        }

        else {

            result += ', ' + $(this).val();

        }

        $('#divResult').html(result);

    });

});

### 30.jQuery mouse events

**Consider the HTML below**

<table>

    <tr>

        <td>First Name</td>

        <td style="vertical-align:middle">

            <input id="txtFirstName" type="text" class="inputRequired" />

        </td>

        <td><img id="imgFirstNameHelp" src="help.png" /></td>

        <td><div id="divFirstNameHelp" style="display:none">

            First Name as shown in passport</div></td>

    </tr>

    <tr>

        <td>Last Name</td>

        <td>

            <input id="txtLastName" type="text" class="inputRequired" />

        </td>

        <td><img id="imgLastNameHelp" src="help.png" /></td>

        <td><div id="divLastNameHelp" style="display:none">

            Last Name as shown in passport</div></td>

    </tr>

    <tr>

        <td>City</td>

        <td>

            <select id="ddlCity" class="inputRequired">

                <option value="Select">Select</option>

                <option value="New York">New York</option>

                <option value="London">London</option>

                <option value="Chennai">Chennai</option>

                <option value="Sydney">Sydney</option>

            </select>

        </td>

        <td><img id="imgCityHelp" src="help.png" /></td>

        <td><div id="divCityHelp" style="display:none">

            Your residence city</div></td>

    </tr>

    <tr>

        <td>Favourite Color</td>

        <td>

            <input id="radioRed" name="color" type="radio" value="Red"

                   class="inputRequired" />Red

            <input id="radioGreen" name="color" type="radio" value="Green"

                   class="inputRequired" />Green<br />

            <input id="radioBlue" name="color" type="radio" value="Blue"

                   class="inputRequired" />Blue

        </td>

        <td><img id="imgColorHelp" src="help.png" /></td>

        <td><div id="divColorHelp" style="display:none">

            Your Favourite Color</div></td>

    </tr>

    <tr>

        <td>Contact Method</td>

        <td>

            <input id="chkBoxEmail" type="checkbox" value="Email"

                   class="inputRequired" />Email

            <input id="chkBoxPhone" type="checkbox" value="Phone"

                   class="inputRequired" />Phone<br />

            <input id="chkBoxSocialMedia" type="checkbox" value="Social Media"

                   class="inputRequired" />Social Media

        </td>

        <td><img id="imgContactMethodHelp" src="help.png" /></td>

        <td><div id="divContactMethodHelp" style="display:none">

            How should we contact you</div></td>

    </tr>

    <tr>

        <td>

            Comments

        </td>

        <td>

            <textarea id="txtComments" class="inputRequired"></textarea>

        </td>

        <td><img id="imgCommentsHelp" src="help.png" /></td>

        <td><div id="divCommentsHelp" style="display:none">

            Your comments please</div></td>

    </tr>

    <tr>

        <td>

            <input id="btnSubmit" type="button" value="Submit" />

        </td>

        <td>

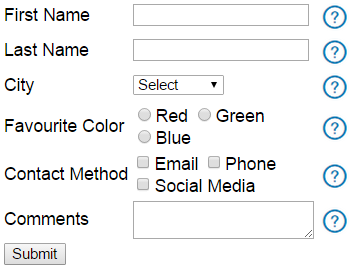
            <div id="divResult"></div>

        </td>

        <td></td>

    </tr>

</table>

When the mouse is over the help icon, we want to display the help text, when the mouse is out, hide the help text.   
   
  
To achieve this **mouseover**and **mouseout**events can be used as shown below.  
  
$(document).ready(function () {

    $('img[src="help.png"]').mouseover(function () {

        $('#' + getDivId(this)).fadeIn(400);

        $(this).css('cursor', 'pointer');

    }).mouseout(function () {

        $('#' + getDivId(this)).fadeOut(400);

    });

    function getDivId(helpIcon) {

        var helpIconId = $(helpIcon).attr('id');

        return helpIconId.replace('img', 'div');

    }

});

**mouseenter** and **mouseleave**events can also be used.

$(document).ready(function () {

    $('img[src="help.png"]').mouseenter(function () {

        $('#' + getDivId(this)).fadeIn(400);

        $(this).css('cursor', 'pointer');

    }).mouseleave(function () {

        $('#' + getDivId(this)).fadeOut(400);

    });

    function getDivId(helpIcon) {

        var helpIconId = $(helpIcon).attr('id');

        return helpIconId.replace('img', 'div');

    }

});

We can also achieve the same using **hover**. hover() function accepts two function arguments, one for **mouseenter**event and one for **mouseleave**event.  
  
$( selector ).hover( handlerIn, handlerOut )   
is shorthand for  
$( selector ).mouseenter( handlerIn ).mouseleave( handlerOut )

$(document).ready(function () {

    $('img[src="help.png"]').hover(function () {

        $('#' + getDivId(this)).fadeIn(400);

        $(this).css('cursor', 'pointer');

    }, function () {

        $('#' + getDivId(this)).fadeOut(400);

    });

    function getDivId(helpIcon) {

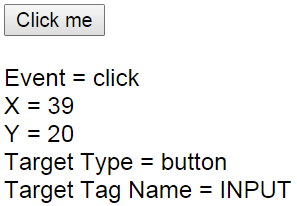
        var helpIconId = $(helpIcon).attr('id');

        return helpIconId.replace('img', 'div');

    }

});

### 31.jQuery event object

Whenever an event (like click, mouseover, mouseout etc) occurs, the relevant data about that event is placed into the event object. For example, the event object contains event data like, the X and Y coordinates of the mouse pointer when the event occurred, the HTML element that fired the event, which mouse button is clicked etc.   
  
**Obtaining the event object is straightforward**. The event object is always passed to the event handler method. Let us understand this with an example. When we click the button, we want to capture the following event data  
1. Event name  
2. Mouse X coordinate when the event occured  
3. Mouse Y coordinate when the event occured  
4. The control that raised the event  
5. The HTML tag name that raised the event   
  
   
  
Notice that in the example below, we are passing event object to getEventDetails() method. This object is the raw JavaScript event object. The type property gives us the event name that occured. pageX and pageY properties return the X and Y coordinates of the mouse pointer. Target property returns the HTML element that raised the event. Target, pageX and pageY properties are supported by all modern browsers and Internet Explorer 9 and above. The following code will not work in Internet Explorer 8 and earlier versions. In addition to click event, the following example returns mouseover and mouseout event data.

<html>

<head>

    <title></title>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            $('#btn').click(function () {

                getEventDetails(event);

            }).mouseover(function () {

                getEventDetails(event);

            }).mouseout(function () {

                getEventDetails(event);

            });

            function getEventDetails(event) {

                var eventDetails = "Event = " + event.type +

                                   "<br/> X = " + event.pageX +

                                   "<br/>Y = " + event.pageY +

                                   "<br/>Target Type = " + event.target.type +

                                   "<br/>Target Tag Name = " + event.target.tagName;

                $('#divResult').html(eventDetails);

            }

        });

    </script>

</head>

<body style="font-family:Arial">

    <input id="btn" type="button" value="Click me" /><br /><br />

    <div id="divResult"></div>

</body>

</html>

**Cross-browser solution :** For the above code to work in all browsers including Internet Explorer 8 and earlier versions, modify getEventDetails() function as shown below. Notice that we are converting JavaScript event object to jQuery event object using $.event.fix()

function getEventDetails(event) {

    var e = $.event.fix(event);

    var eventDetails = "Event = " + e.type +

                        "<br/> X = " + e.pageX +

                        "<br/>Y = " + e.pageY +

                        "<br/>Target Type = " + e.target.type +

                        "<br/>Target Tag Name = " + e.target.tagName;

    $('#divResult').html(eventDetails);

}

### 32. jQuery to detect which mouse button clicked

**With raw JavaScript event object :** Depending on the browser, event.button or event.which properties of the event object are used to determine which mouse button is clicked.    
  
**IE 8 & earlier versions use event.button property**  
Left Button 1  
Middle Button 4  
Right Button 2  
  
**IE 9 & later versions and most other W3C compliant browsers use event.which property**  
Left Button 1  
Middle Button 2  
Right Button 3  
  
If you are using raw JavaScript event object, the following is the amount of code that you have to write to detect which mouse button is clicked. This code works in all browsers.

<html>

<head>

    <title></title>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        function whichMouseButtonClicked(event) {

            var whichButton;

            if (event.which) {

                switch (event.which) {

                    case 1: whichButton = "Left Button Clicked"; break;

                    case 2: whichButton = "Middle Button Clicked"; break;

                    case 3: whichButton = "Right Button Clicked"; break;

                    default: whichButton = "Invalid Button Clicked"; break;

                }

            }

            else {

                switch (event.button) {

                    case 1: whichButton = "Left Button Clicked"; break;

                    case 4: whichButton = "Middle Button Clicked"; break;

                    case 2: whichButton = "Right Button Clicked"; break;

                    default: whichButton = "Invalid Button Clicked"; break;

                }

            }

            document.getElementById('divResult').innerHTML = whichButton;

        }

    </script>

</head>

<body style="font-family:Arial">

    <input id="btn" type="button" value="Click Me"

           onmouseup="whichMouseButtonClicked(event)" />

    <br /><br />

    <div id="divResult"></div>

</body>

</html>

**jQuery normalizes which property of the event object so it will work across all browsers**. The amount of code you have to write is lot less.

<html>

<head>

    <title></title>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            $('#btn').mouseup(function (event) {

                switch (event.which) {

                    case 1: whichButton = "Left Button Clicked"; break;

                    case 2: whichButton = "Middle Button Clicked"; break;

                    case 3: whichButton = "Right Button Clicked"; break;

                    default: whichButton = "Invalid Button Clicked"; break;

                }

                $('#divResult').html(whichButton);

            });

        });

    </script>

</head>

<body style="font-family:Arial">

    <input id="btn" type="button" value="Click Me" />

    <br /><br />

    <div id="divResult"></div>

</body>

</html>

### 33.Binding event handlers in jquery

One way to bind event handlers to events is by using the **jQuery shorthand functions**like .click, .mouseover etc. Bind method is another way of doing the same.  
  
The following example binds click event handler to the button - btnClickMe

$('#btnClickMe').bind('click', function () {

    $('#divResult').html('Button Clicked');

});

Binds **multiple event handlers** to the button - btnClickMe. If required, the event object can also be passed to the event handler method, although it is optional.

$('#btnClickMe').bind('click mouseover mouseout', function (event) {

    if (event.type == 'click') {

        $('#divResult').html('Button Clicked at ' + 'X = '

            + event.pageX + ' Y = ' + event.pageY);

    }

    else if (event.type == 'mouseover') {

        $(this).addClass('ButtonStyle');

    }

    else {

        $(this).removeClass('ButtonStyle');

    }

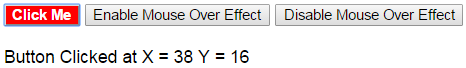
});

**Use the unbind() method to unbind the event handler**

$('#btnClickMe').unbind('mouseover');

**To unbind all the event handlers of an element, use unbind() method without any parameters**

$('#btnClickMe').unbind();

**Please note:** If you are using jQuery 1.7 or higher, you should be using on() and off() methods instead of bind() and unbind() methods. We will discuss on() and off() methods in our next video.  
  
**Complete Example :**   
 

<html>

<head>

    <title></title>

    <style>

        .ButtonStyle {

            background-color: red;

            cursor: pointer;

            font-weight: bold;

            color: white;

        }

    </style>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            $('#btnClickMe').bind('click mouseover mouseout', function (event) {

                if (event.type == 'click') {

                    $('#divResult').html('Button Clicked at '

                        + 'X = ' + event.pageX + ' Y = ' + event.pageY);

                }

                else if (event.type == 'mouseover') {

                    $(this).addClass('ButtonStyle');

                }

                else {

                    $(this).removeClass('ButtonStyle');

                }

            });

            $('#btnEnableMouseOverEffect').click(function () {

                $('#btnClickMe').bind('mouseover', function () {

                    $(this).addClass('ButtonStyle');

                });

            });

            $('#btnDisableMouseOverEffect').click(function () {

                $('#btnClickMe').unbind('mouseover');

            });

        });

    </script>

</head>

<body style="font-family:Arial">

    <input id="btnClickMe" type="button" value="Click Me" />

    <input id="btnEnableMouseOverEffect" type="button"

           value="Enable Mouse Over Effect" />

    <input id="btnDisableMouseOverEffect" type="button"

           value="Disable Mouse Over Effect" />

    <br /><br />

    <div id="divResult"></div>

</body>

</html>

### 34.Binding event handlers using jquery on method

In [Part 33](http://csharp-video-tutorials.blogspot.com/2015/04/binding-event-handlers-in-jquery.html) of [jQuery Tutorial](https://www.youtube.com/playlist?list=PL6n9fhu94yhVDV697uvHpavA3K_eWGQap), we discussed how to bind event handlers using **bind()**method. If you are using jQuery 1.7 or higher, you should be using **on()**and **off()**methods instead of **bind()**and **unbind()**methods.   
  
The syntax for using on() and off() methods is very similar to using bind() and unbind() methods. The following example is the same example we worked with in [Part 33](http://csharp-video-tutorials.blogspot.com/2015/04/binding-event-handlers-in-jquery.html). Instead of using bind() and unbind() we are using on() and off() methods.

<html>

<head>

    <title></title>

    <style>

        .ButtonStyle {

            background-color: red;

            cursor: pointer;

            font-weight: bold;

            color: white;

        }

    </style>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            $('#btnClickMe').on('click mouseover mouseout', function (event) {

                if (event.type == 'click') {

                    $('#divResult').html('Button Clicked at ' + 'X = '

                        + event.pageX + ' Y = ' + event.pageY);

                }

                else if (event.type == 'mouseover') {

                    $(this).addClass('ButtonStyle');

                }

                else {

                    $(this).removeClass('ButtonStyle');

                }

            });

            $('#btnEnableMouseOverEffect').click(function () {

                $('#btnClickMe').on('mouseover', function () {

                    $(this).addClass('ButtonStyle');

                });

            });

            $('#btnDisableMouseOverEffect').click(function () {

                $('#btnClickMe').off('mouseover');

            });

        });

    </script>

</head>

<body style="font-family:Arial">

    <input id="btnClickMe" type="button" value="Click Me" />

    <input id="btnEnableMouseOverEffect" type="button"

           value="Enable Mouse Over Effect" />

    <input id="btnDisableMouseOverEffect" type="button"

           value="Disable Mouse Over Effect" />

    <br /><br />

    <div id="divResult"></div>

</body>

</html>

In the example above we are binding a single anonymous function as the event handler for all the 3 events  
1. click  
2. mouseover  
3. mouseout  
  
If you want to bind a different anonymous function, you could do so using the following syntax.

$('#btnClickMe').bind({

    click: function (event) {

        $('#divResult').html('Button clicked at X = '

            + event.pageX + ' Y = ' + event.pageY);

    },

    mouseover: function () {

        $(this).addClass('ButtonStyle');

    },

    mouseout: function () {

        $(this).removeClass('ButtonStyle');

    }

});

jQuery shorthand functions (.click, .mouseover, .mouseout etc.) call **on()** method behind the scenes.  
  
So far in this video series we have seen 3 different ways of binding event handlers in jQuery  
1. Using jQuery shorthand functions (.click, .mouseover, .mouseout etc.)  
element.click(function () { ... });  
  
2. With jQuery version < 1.7, bind() method can be used  
element.bind('click', function () { ... });  
  
3. With jQuery version 1.7 or higher, on() method can be used.   
element.on('click', function () { ... });  
  
According to jQuery.com, as of jQuery 1.7, the .on() method is the preferred method for attaching event handlers.

### 35.Passing data to event handler in jQuery

**The following example,**  
1. Binds the click event handler to the button using on function  
2. We are passing 3 arguments to the on() function  
    a) The name of the event  
    b) JSON object that contains data that we want to pass to the event handler  
    c) Event handler method name  
3. In the event handler method (sayHello), we access the data using event object's data property.

<html>

<head>

    <title></title>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            $('#btnClickMe').on('click', {

                firstName: 'John',

                lastName: 'Doe'

            }, sayHello);

            $('#btnClickMe').on('click', {

                firstName: 'Mary'

            }, sayHello);

            $('#btnClickMe').on('click', sayHello);

            function sayHello(event) {

                if (event.data == null) {

                    alert('No name provided');

                }

                else {

                    alert('Hello ' + event.data.firstName +

                        (event.data.lastName != null ? ' ' + event.data.lastName : ''));

                }

            }

        });

    </script>

</head>

<body style="font-family:Arial">

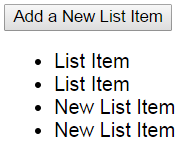
    <input id="btnClickMe" type="button" value="Click Me" />

</body>

</html>

### Output :  Hello John Doe Hello Mary No name provided

### 36. jQuery add event handler to dynamically created element

The following example, allows us to dynamically create new list item (li), attach a click event handler and add it to the unordered list (ul). This happens when you click **"Add a New List Item"** button. The problem with this approach is that we are binding a click event handler to every list item. This means if you have 500 list items, then there will be 500 event handlers in the memory and this may negatively affect the performance of your application.   
  
 

<html>

<head>

    <title></title>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            $('li').on('click', function () {

                $(this).fadeOut(500);

            });

            $('#btnAdd').on('click', function () {

                var newListItem = $('<li>New List Item</li>').on('click', function () {

                    $(this).fadeOut(500);

                });

                $('ul').append(newListItem);

            });

        });

    </script>

</head>

<body style="font-family:Arial">

    <input id="btnAdd" type="button" value="Add a New List Item" />

    <ul>

        <li>List Item</li>

        <li>List Item</li>

    </ul>

</body>

</html>

A better way of doing the same from a performance standpoint is shown below. In this example, the click event handler is attached to the listitem (li) parent element (ul). Even if you have 500 list items, there is only one click event handler in memory.   
  
**So how does this work**  
1. When you click on a list item (li), the event gets bubbled up to its parent (ul) as the list item (li) does not have an event handler   
2. The bubbled event is handled by the the parent (ul) element, as it has a click event handler.  
3. When a new list item is added dynamicaly, you don't have to add the click event handler to it. Since the newly created list item (li) is added to the same parent element (ul), the click event of this list item also gets bubbled upto the same parent and will be handled by it. 

<html>

<head>

    <title></title>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            $('ul').on('click', 'li', function () {

                $(this).fadeOut(500);

            });

            $('#btnAdd').on('click', function () {

                $('ul').append('<li>New List Item</li>');

            });

        });

    </script>

</head>

<body style="font-family:Arial">

    <input id="btnAdd" type="button" value="Add a New List Item" />

    <ul>

        <li>List Item</li>

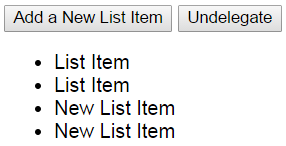
        <li>List Item</li>

    </ul>

</body>

</html>

### 37. jQuery event delegation

3. When a new list item is added dynamicaly, you don't have to add the click event handler explicitly to it. Since the newly created list item (li) is added to the same parent element (ul), the click event of this list item also gets bubbled upto the same parent and will be handled by it.  
4. undelegate() stops event delegation   
  
 

<html>

<head>

    <title></title>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            $('ul').delegate('li', 'click', function () {

                $(this).fadeOut(500);

            });

            $('#btnAdd').on('click', function () {

                $('ul').append('<li>New List Item</li>');

            });

            $('#btnUndelegate').on('click', function () {

                $('ul').undelegate('li', 'click');

            });

        });

    </script>

</head>

<body style="font-family:Arial">

    <input id="btnAdd" type="button" value="Add a New List Item" />

    <input id="btnUndelegate" type="button" value="Undelegate" />

    <ul>

        <li>List Item</li>

        <li>List Item</li>

    </ul>

</body>

</html>

If you are using **jQuery 1.7 or higher** version, jQuery recommends to use **on()** to perform event delegation instead of **delegate()**. The above example can be very easily rewritten using **on()**and **off()**functions, instead of **delegate()**and **undelegate()**functions as shown below. We discussed performing event delegation using on() function in detail in [Part 36](http://csharp-video-tutorials.blogspot.com/2015/04/jquery-add-event-handler-to-dynamically.html) of [jQuery tutorial](https://www.youtube.com/playlist?list=PL6n9fhu94yhVDV697uvHpavA3K_eWGQap).

<html>

<head>

    <title></title>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            $('ul').on('click', 'li', function () {

                $(this).fadeOut(500);

            });

            $('#btnAdd').on('click', function () {

                $('ul').append('<li>New List Item</li>');

            });

            $('#btnUndelegate').on('click', function () {

                $('ul').off('click', 'li');

            });

        });

    </script>

</head>

<body style="font-family:Arial">

    <input id="btnAdd" type="button" value="Add a New List Item" />

    <input id="btnUndelegate" type="button" value="Undelegate" />

    <ul>

        <li>List Item</li>

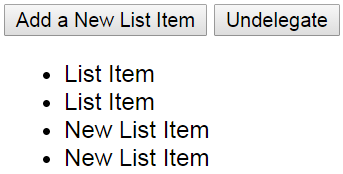
        <li>List Item</li>

    </ul>

</body>

</html>

### 38.jQuery live function

We discussed how to perform event delegation, using **on()** method in [Part 36](http://csharp-video-tutorials.blogspot.com/2015/05/jquery-event-delegation.html) and using **delegate()**method in [Part 37](http://csharp-video-tutorials.blogspot.com/2015/04/jquery-add-event-handler-to-dynamically.html). Another way to perform event delegation is by using **live()**function.   
  
With **on()** and **delegate()**functions the event gets bubbled up to the specified parent element, where as with **live()**function the event gets bubbled up all the way to the document object.   
  
The example that we worked with in [Part 36](http://csharp-video-tutorials.blogspot.com/2015/04/jquery-add-event-handler-to-dynamically.html) and [Part 37](http://csharp-video-tutorials.blogspot.com/2015/05/jquery-event-delegation.html) is rewritten using **live()** and **die()**functions as shown below. Notice that, to perform event delegation we are using **live()**function and to stop event delegation we are using **die()**function.   
  
 

<html>

<head>

    <title></title>

    <!--<script src="jquery-1.11.2.js"></script>-->

    <script src="http://ajax.aspnetcdn.com/ajax/jQuery/jquery-1.7.js">

    </script>

    <script type="text/javascript">

        $(document).ready(function () {

            $('li').live('click', function () {

                $(this).fadeOut(500);

            });

            $('#btnAdd').on('click', function () {

                $('ul').append('<li>New List Item</li>');

            });

            $('#btnUndelegate').on('click', function () {

                $('li').die('click');

            });

        });

    </script>

</head>

<body style="font-family:Arial">

    <input id="btnAdd" type="button" value="Add a New List Item" />

    <input id="btnUndelegate" type="button" value="Undelegate" />

    <ul>

        <li>List Item</li>

        <li>List Item</li>

    </ul>

</body>

</html>

**live()**function is deprecated in jQuery 1.7 and completely removed in jQuery 1.9. Everything that can be achieved with the following methods can be achieved by using **.on()**function.  
**live()**  
**bind()**  
**delegate()**  
  
So, if you are using jQuery 1.7 or higher version, jQuery recommends to use on() function.   
  
**Please note :** Older versions of jQuery can be found on the Microsoft CDN  
<http://www.asp.net/ajax/cdn>

### 39.jquery execute event only once

Every time you click the button, you will get the JavaScript alert.

$(document).ready(function () {

    $('#btn').on('click', function () {

        alert('Button Clicked');

    });

});

If you want to execute the click event handler only once, then you will have to explicitly remove the click event handler. The following example removes the click event handler using **off()** function, after the alert is displayed.

$(document).ready(function () {

    $('#btn').on('click', function () {

        alert('Button Clicked');

        $(this).off('click');

    });

});

jQuery **one()** function does exactly the same thing. The click event is raised only once. On the first click, JavaScript alert is displayed, but on subsequent clicks nothing happens.

$(document).ready(function () {

    $('#btn').one('click', function () {

        alert('Button Clicked');

    });

});

The following example binds 3 events(mouseover, mouseout, click) using on() function. If we want all these 3 events to execute only once, then we have to explicitly remove each event after first execution using off() method. 

$(document).ready(function () {

    $('#btn').on({

        mouseover: function () {

            $(this).css('background-color', 'yellow');

            $(this).off('mouseover');

        },

        mouseout: function () {

            $(this).css('background-color', 'white');

            $(this).off('mouseout');

        },

        click: function () {

            alert('Button clicked');

            $(this).off('click');

        }

    });

});

The above example can be rewritten using one() function as shown below.

$(document).ready(function () {

    $('#btn').one({

        mouseover: function () {

            $(this).css('background-color', 'yellow');

        },

        mouseout: function () {

            $(this).css('background-color', 'white');

        },

        click: function () {

            alert('Button clicked');

        }

    });

});

one() function executes the handler at most once per element per event type. In the following example, click, mouseover and mouseount events are executed atmost once for each button element.

<html>

<head>

    <title></title>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            $('input[type="button"]').one({

                mouseover: function () {

                    $(this).css('background-color', 'yellow');

                },

                mouseout: function () {

                    $(this).css('background-color', 'white');

                },

                click: function () {

                    alert('Button clicked');

                }

            });

        });

    </script>

</head>

<body style="font-family:Arial">

    <input type="button" value="Click Me" />

    <input type="button" value="Click Me" />

</body>

</html>

### 40.jQuery how to check if event is already bound

Why is it important to check if an event is already bound  
To prevent attaching event handler multiple times  
  
The following example checks if a click event handler is already bound. If it's not already bound, then a click event handler is attached.

<html>

<head>

    <title></title>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            var jQueryObject = $('#btn');

            var rawDOMElement = jQueryObject.get(0);

            var eventObject = $.\_data(rawDOMElement, 'events');

            if (eventObject != undefined && eventObject.click != undefined) {

                alert('Click event exists');

            }

            else {

                $('#btn').on('click', function () {

                    alert('Button Clicked');

                });

            }

        });

    </script>

</head>

<body style="font-family:Arial">

    <input id="btn" type="button" value="Click Me" />

</body>

</html>

Please note that this only works if you have attached event handlers using jQuery. This will not work if you have attached event handlers using raw JavaScript or element attributes.  
  
**Another way to prevent attaching event handlers multiple times is by using jQuery off() and on() methods**. The off() method ensures that all existing click event handlers of the button are removed before again adding a new click event handler using on() method.

$('#btn').off('click').on('click', function () {

    alert('Button Clicked');

});

### 41. jQuery preventdefault

### In this video we will discuss how to prevent browser default action using jQuery. First let's look at some of the browser default actions. For example, 1. When you right click on a web page, the browser displays the context menu  2. When you click on a link, the browser navigates to the page specified in the link

In some situations you may want to prevent these default actions of the browser. For example some of the websites prevent you from right clicking on the page. Disabling right click is annoying users. Many people say they disabled right click for security, because they do not want their content to be copied. But if you disable JavaScript in the browser, you will still be able to right click and copy the content. So you are achieving nothing by disabling right click.     
  
Having said that, now let us see how to prevent the context menu from appearing when you right click on the web page. We discussed how to achieve this using raw JavaScript in [Part 43](http://csharp-video-tutorials.blogspot.com/2015/01/preventing-browser-default-action.html) of [JavaScript Tutorial](https://www.youtube.com/playlist?list=PL6n9fhu94yhUA99nOsJkKXBqokT3MBK0b).   
  
Let us now discuss, how to achieve this using jQuery 

<html>

<head>

    <title></title>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            $(this).on('contextmenu', function (e) {

                e.preventDefault();

                $('#divResult').append('Right click disabled<br/>')

            });

        });

    </script>

</head>

<body style="font-family:Arial">

    <h3>

        Right click disabled on this page. Try

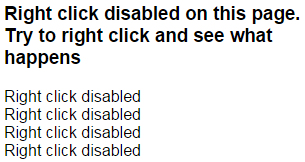
        to right click and see what happens

    </h3>

    <div id="divResult"></div>

</body>

</html>

   
  
**When you click on a link, how to prevent the browser from navigating to the page specified in the link.** 

<html>

<head>

    <title></title>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            $('#myHyperLink').on('click', function (e) {

                e.preventDefault();

                $('#divResult').append('Hyperlink default action prevented<br/>')

            });

        });

    </script>

</head>

<body style="font-family:Arial">

    <a id="myHyperLink" href="http://pragimtech.com">

        Clicking on the link will not take you to PragimTech

    </a>

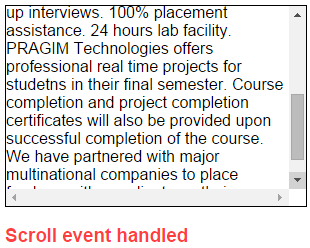
    <br /><br />

    <div id="divResult"></div>

</body>

</html>

### jquery prevent link click  42. jQuery scroll event

scroll event is raised when the user scrolls to a different place in the element. It applies to window objects, but also to scrollable frames and elements with the overflow CSS property set to scroll.   
  
In the following example, notice that the div element overflow style is set to scroll. The div element has a scroll bar and as you scroll up and down the div element, the scroll event is raised and handled.   
  
 

<html>

<head>

    <title></title>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            $('div').scroll(function () {

                $('h3').css('display', 'inline').fadeOut(1000);

            });

        });

    </script>

</head>

<body style="font-family:Arial">

    <div style="width:300px; height:200px; border:1px solid black; overflow:scroll">

        PRAGIM Specialty in training arena unlike other training institutions. Training delivered by real time software experts having more than 10 years of experience. Interview questions and real time scenarios discussion on topics covered for the day. Real time projects discussion relating to the possible interview questions. Trainees can attend training and use lab until you get a job. Resume preparation and mock up interviews. 100% placement assistance. 24 hours lab facility. PRAGIM Technologies offers professional real time projects for students in their final semester. Course completion and project completion certificates will also be provided upon successful completion of the course. We have partnered with major multinational companies to place freshers with our clients on their course completion.

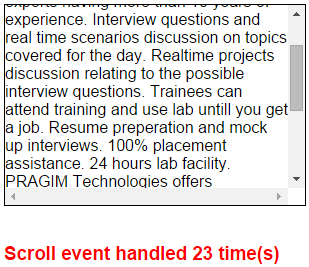
    </div>

    <br />

    <h3 style="display:none; color:red">Scroll event handled</h3>

</body>

</html>

**The following example, keeps track of how many times the scroll event is raised**.   
 

<html>

<head>

    <title></title>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            var count = 0;

            $('div').scroll(function () {

                $('span').text(count += 1);

            });

        });

    </script>

</head>

<body style="font-family:Arial">

    <div style="width:300px; height:200px; border:1px solid black; overflow:scroll">

        PRAGIM Speciality in training arena unlike other training institutions. Training delivered by real time software experts having more than 10 years of experience. Interview questions and real time scenarios discussion on topics covered for the day. Realtime projects discussion relating to the possible interview questions. Trainees can attend training and use lab untill you get a job. Resume preperation and mock up interviews. 100% placement assistance. 24 hours lab facility. PRAGIM Technologies offers professional real time projects for studetns in their final semester. Course completion and project completion certificates will also be provided upon successful completion of the course. We have partnered with major multinational companies to place freshers with our clients on their course completion.

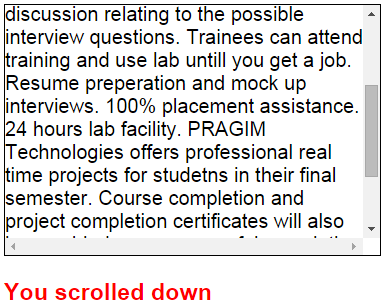
    </div>

    <br />

    <h3 style="color:red">Scroll event handled <span>0</span> time(s)</h3>

</body>

</html>

**The following example, keeps track of whether the user scrolled up or down.**   
 

<html>

<head>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            var lastScrollTop = 0;

            $('div').scroll(function () {

                var currentScrollPosition = $(this).scrollTop();

                if (currentScrollPosition > lastScrollTop) {

                    $('h3').text('You scrolled down');

                } else {

                    $('h3').text('You scrolled up');

                }

                lastScrollTop = currentScrollPosition;

            });

        });

    </script>

</head>

<body style="font-family:Arial">

    <div style="width:300px; height:200px; border:1px solid black; overflow:scroll">

        PRAGIM Speciality in training arena unlike other training institutions. Training delivered by real time software experts having more than 10 years of experience. Interview questions and real time scenarios discussion on topics covered for the day. Realtime projects discussion relating to the possible interview questions. Trainees can attend training and use lab untill you get a job. Resume preperation and mock up interviews. 100% placement assistance. 24 hours lab facility. PRAGIM Technologies offers professional real time projects for studetns in their final semester. Course completion and project completion certificates will also be provided upon successful completion of the course. We have partnered with major multinational companies to place freshers with our clients on their course completion.

    </div>

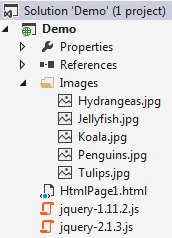
    <br />

    <h3 style="display:inline; color:red"></h3>

</body>

</html>

### 43.jQuery image gallery

The image gallery should be as shown in the image below. When you click on the image thumnail, the respective image should be displayed in the main section of the page.   
   
  
For the purpose of this demo we will be using the images that can be found on any windows machine at the following path.  
C:\Users\Public\Pictures\Sample Pictures   
  
**Step 1 :** Open Visual Studio and create a new empty asp.net web application project. Name it **Demo**.  
  
**Step 2 :** Right click on the Project Name in Solution Explorer in Visual Studio and create a new folder with name = **Images**.  
  
**Step 3 :** Copy images from C:\Users\Public\Pictures\Sample Pictures to **Images**folder in your project.  
  
**Step 4 :** Right click on the Project Name in Solution Explorer in Visual Studio and add a new HTML Page. It should automatically add HTMLPage1.htm.    
  
**Step 5 :** Add a reference to the download jQuery file. I am using jquery-1.11.2.js version for this demo. At this point your solution explorer should look as shown below.   
  
   
  
**Step 6 :** Copy and paste the following HTML and jQuery code in HTMLPage1.htm page. 

<html>

<head>

    <style type="text/css">

        .imgStyle {

            width: 100px;

            height: 100px;

            border: 3px solid grey;

        }

    </style>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            $('#divId img').on({

                mouseover: function () {

                    $(this).css({

                        'cursor': 'hand',

                        'border-Color': 'red'

                    });

                },

                mouseout: function () {

                    $(this).css({

                        'cursor': 'default',

                        'border-Color': 'grey'

                    });

                },

                click: function () {

                    var imageURL = $(this).attr('src');

                    $('#mainImage').fadeOut(1000, function () {

                        $(this).attr('src', imageURL);

                    }).fadeIn(1000);

                }

            });

        });

    </script>

</head>

<body>

    <img id="mainImage" style="border:3px solid grey"

         src="/Images/Hydrangeas.jpg" height="500" width="540" />

    <br />

    <div id="divId">

        <img class="imgStyle" src="/Images/Hydrangeas.jpg" />

        <img class="imgStyle" src="/Images/Jellyfish.jpg" />

        <img class="imgStyle" src="/Images/Koala.jpg" />

        <img class="imgStyle" src="/Images/Penguins.jpg" />

        <img class="imgStyle" src="/Images/Tulips.jpg" />

    </div>

</body>

</html>

### In our next video, we will discuss how to make this image gallery a bit more efficient using the concept of event bubbling.

### 44.Optimise jQuery image gallery

In this video we will discuss how to optimise the image gallery we created in [Part 43](http://csharp-video-tutorials.blogspot.com/2015/05/jquery-image-gallery.html), using the concept of event bubbling. We will also enhance the example, using a dropdownlist for effects and duration. We will also be able to enlarge and shrink the height and width of the main image. Please watch [Part 43](http://csharp-video-tutorials.blogspot.com/2015/05/jquery-image-gallery.html) from [jQuery tutorial](https://www.youtube.com/playlist?list=PL6n9fhu94yhVDV697uvHpavA3K_eWGQap) before proceeding.

   
  
The problem with the image gallery that we created in [Part 43](http://csharp-video-tutorials.blogspot.com/2015/05/jquery-image-gallery.html) is that we are binding event handlers (mouseover, mouseout & click) to every image element. This means if you have 500 image elements, then there will be 1500 event handlers (mouseover, mouseout & click) in the memory and this may negatively affect the performance of your application.    
  
A better way of doing the same from a performance standpoint is shown below. In this example, the event handlers are attached to the div element and not to the individual img elements. So, even if you have 500 img elements, there are only 3 event handlers in memory.    
  
**So how does this work**  
1. When you click on an img element, the event gets bubbled up to its parent (div) as the img element does not have an event handler   
2. The bubbled event is handled by the the parent (div) element, as it has a click event handler. 

<html>

<head>

    <style type="text/css">

        .imgStyle {

            width: 100px;

            height: 100px;

            border: 3px solid grey;

        }

    </style>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            $('#divId').on({

                mouseover: function () {

                    $(this).css({

                        'cursor': 'hand',

                        'border-Color': 'red'

                    });

                },

                mouseout: function () {

                    $(this).css({

                        'cursor': 'default',

                        'border-Color': 'grey'

                    });

                },

                click: function () {

                    var imageURL = $(this).attr('src');

                    var effect = $('#selectImgEffect').val();

                    var duration = $('#selectImgDuration').val() \* 1000;

                    if (effect == 'Slide') {

                        $('#mainImage').slideUp(duration, function () {

                            $(this).attr('src', imageURL);

                        }).slideDown(duration);

                    }

                    else {

                        $('#mainImage').fadeOut(duration, function () {

                            $(this).attr('src', imageURL);

                        }).fadeIn(duration);

                    }

                }

            }, 'img');

            var mainImageElement = $('#mainImage');

            var height = parseInt(mainImageElement.attr('height'));

            var width = parseInt(mainImageElement.attr('width'))

            $('#btnEnlarge').click(function () {

                height += 100;

                width += 100;

                mainImageElement.animate({ 'height': height, 'width': width });

            });

            $('#btnShrink').click(function () {

                height -= 100;

                width -= 100;

                mainImageElement.animate({ 'height': height, 'width': width });

            });

        });

    </script>

</head>

<body style="font-family:Arial">

    Select Effect :

    <select id="selectImgEffect">

        <option value="Fade">Fade</option>

        <option value="Slide">Slide</option>

    </select>

    Time in seconds:

    <select id="selectImgDuration">

        <option value="0.5">0.5</option>

        <option value="1">1</option>

        <option value="2">2</option>

        <option value="3">3</option>

        <option value="4">4</option>

        <option value="5">5</option>

    </select>

    <input id="btnEnlarge" type="button" value="Enlarge" />

    <input id="btnShrink" type="button" value="Shrink" />

    <br /><br />

    <img id="mainImage" style="border:3px solid grey"

         src="/Images/Hydrangeas.jpg" height="500" width="540" />

    <br />

    <div id="divId">

        <img class="imgStyle" src="/Images/Hydrangeas.jpg" />

        <img class="imgStyle" src="/Images/Jellyfish.jpg" />

        <img class="imgStyle" src="/Images/Koala.jpg" />

        <img class="imgStyle" src="/Images/Penguins.jpg" />

        <img class="imgStyle" src="/Images/Tulips.jpg" />

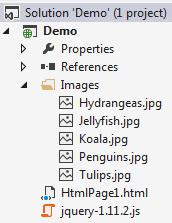
    </div>

</body>

</html>

### 45. jquery image slideshow with thumbnails

In this video, we will discuss creating a simple **image slideshow using jQuery**. We will be using **setInterval()** and **clearInterval()** JavaScript methods to achieve this. We discussed these functions in detail in [Part 34](http://csharp-video-tutorials.blogspot.com/2015/01/javascript-timing-events.html) of [JavaScript Tutorial](https://www.youtube.com/playlist?list=PL6n9fhu94yhUA99nOsJkKXBqokT3MBK0b).

The slideshow should be as shown in the image below. When you click **"Start Slide Show"** button the image slideshow should start and when you click the **"Stop Slide Show"** button the image slideshow should stop.  
  
   
  
For the purpose of this demo we will be using the images that can be found on any windows machine at the following path.  
C:\Users\Public\Pictures\Sample Pictures   
  
**Here are the steps to create the image slideshow using jQuery**   
  
**Step 1 :** Open Visual Studio and create a new empty asp.net web application project. Name it Demo.  
  
**Step 2 :**Right click on the Project Name in Solution Explorer in Visual Studio and create a new folder with name = Images.  
  
**Step 3 :**Copy the images from C:\Users\Public\Pictures\Sample Pictures to Images folder in your project.   
  
**Step 4 :**Right click on the Project Name in Solution Explorer in Visual Studio and add a new HTML Page. It should automatically add HTMLPage1.htm. Also add a jQuery file. At this point your solution explorer should look as shown below.    
  
   
  
**Step 5 :** Copy and paste the following HTML and jQuery code in HTMLPage1.htm page.

<html>

<head>

    <style type="text/css">

        .imgStyle {

            width: 100px;

            height: 100px;

            border: 3px solid grey;

        }

    </style>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            var imageURLs = new Array();

            var intervalId;

            var btnStart = $('#btnStartSlideShow');

            var btnStop = $('#btnStopSlideShow');

            $('#divId img').each(function () {

                imageURLs.push($(this).attr('src'));

            });

            function setImage() {

                var mainImageElement = $('#mainImage');

                var currentImageURL = mainImageElement.attr('src');

                var currentImageIndex = $.inArray(currentImageURL, imageURLs);

                if (currentImageIndex == (imageURLs.length - 1)) {

                    currentImageIndex = -1;

                }

                mainImageElement.attr('src', imageURLs[currentImageIndex + 1])

            }

            btnStart.click(function () {

                intervalId = setInterval(setImage, 500);

                $(this).attr('disabled', 'disabled');

                btnStop.removeAttr('disabled');

            });

            btnStop.click(function () {

                clearInterval(intervalId);

                $(this).attr('disabled', 'disabled');

                btnStart.removeAttr('disabled');

            }).attr('disabled', 'disabled');

        });

    </script>

</head>

<body style="font-family:Arial">

    <input id="btnStartSlideShow" type="button" value="Start Slideshow" />

    <input id="btnStopSlideShow" type="button" value="Stop Slideshow" />

    <br /><br />

    <img id="mainImage" style="border:3px solid grey"

         src="/Images/Hydrangeas.jpg" height="500" width="540" />

    <br />

    <div id="divId">

        <img class="imgStyle" src="/Images/Hydrangeas.jpg" />

        <img class="imgStyle" src="/Images/Jellyfish.jpg" />

        <img class="imgStyle" src="/Images/Koala.jpg" />

        <img class="imgStyle" src="/Images/Penguins.jpg" />

        <img class="imgStyle" src="/Images/Tulips.jpg" />

    </div>

</body>

</html>

### 46.jquery animate function

The following example animates the div element, while changing the font-size property of the div element from its initial size to 50 pixels over a period of 2000 milli-seconds (2 seconds).

<html>

<head>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            $('#myButton').click(function () {

                $('#myDiv').animate({ 'font-size': '50' }, 2000);

            });

        });

    </script>

</head>

<body style="font-family:Arial">

    <input type="button" id="myButton" value="Animate" />

    <br /><br />

    <div id="myDiv">

        jQuery animate function

    </div>

</body>

</html>

**Syntax of jquery animate function**  
.animate( properties [, duration ] [, easing ] [, complete ] )   
  
Animate function has 4 parameters. Only the first parameter (properties) is the required parameter. Rest 3 are optional. 

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| properties | An object of CSS properties and values |
| duration | The duration for animation in milliseconds. Default is 400. |
| easing | Easing function to use for the transition. Default is swing. You could also use linear. |
| complete | A function to call once the animation is complete |

**What is jQuery easing**  
Easing is a technique where the speed and/or direction of animation are changed while the animation is in progress. Easing can make the animation start off slow and gradually speed up, start up fast and gradually slow down, and a whole host of other effects. The difference between linear and swing easing is very subtle.  
  
The following page shows all the easings provided by jQuery UI  
<https://jqueryui.com/easing>  
  
The following example increases the height and width of the image to 400 pixels on mouseover. On mouseout the height and width are reduced to 100 pixels.

<html>

<head>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            $('#myImage').on({

                mouseover: function () {

                    $(this).animate({

                        'height': 400,

                        'width': 400,

                    }, 3000);

                },

                mouseout: function () {

                    $(this).animate({

                        'height': 100,

                        'width': 100,

                    }, 3000);

                }

            });

        });

    </script>

</head>

<body style="font-family:Arial">

    <img id="myImage" height="100" width="100" src="/Images/Tulips.jpg" />

</body>

</html>

In the following example, several calls to animate() method are chained together. By default these calls are placed into a queue to be executed one after the other in series rather than executing all of them simultaneously in parallel.

<html>

<head>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            $('#myImage').click(function () {

                $(this)

                    .animate({ 'left': '300' })

                    .animate({ 'top': '200' })

                    .animate({ 'left': '10' })

                    .animate({ 'top': '10' });

            });

        });

    </script>

</head>

<body style="font-family:Arial">

    <img id="myImage" height="100" width="100" style="position:absolute"

         src="/Images/Tulips.jpg" />

</body>

</html>

**Please note:** By default, all HTML elements have a static position, and cannot be moved. To modify the position , set the CSS position property of the element to fixed, absolute or relative.

### 47. jquery animation queue

When several calls to animate() method are chained together. By default these calls are placed into a queue to be executed one after the other in series rather than executing all of them simultaneously in parallel. The name of this queue is **fx**.   
  
Each HTML element has its own queue. With the following code there will be 5 calls to animate method placed in the queue of each div element. This means both div elements (myDiv1 & myDiv2) may start to execute the first call to animate method more or less at the same time. However, from the given queue the queued methods are executed one after the other in series.

<html>

<head>

    <style>

        .myDivClass {

            width: 150px;

            padding: 5px;

            font-size: 18px;

            border: 1px solid black;

            opacity: .4;

            background-color: red;

            color: white;

        }

    </style>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            $('#myButton').click(function () {

                $('#myDiv1')

                    .animate({ 'width': 500 }, 1500)

                    .animate({ 'padding': 20 }, 1500)

                    .animate({ 'font-size': 50 }, 1500)

                    .animate({ 'border-width': 10 }, 1500)

                    .animate({ 'opacity': 1 }, 1500);

                $('#myDiv2')

                    .animate({ 'width': 500 }, 1500)

                    .animate({ 'padding': 20 }, 1500)

                    .animate({ 'font-size': 50 }, 1500)

                    .animate({ 'border-width': 10 }, 1500)

                    .animate({ 'opacity': 1 }, 1500);

            });

        });

    </script>

</head>

<body style="font-family:Arial">

    <input type="button" id="myButton" value="Start Animation" />

    <br /><br />

    <div id="myDiv1" class="myDivClass">

        jQuery animations

    </div>

    <br />

    <div id="myDiv2" class="myDivClass">

        Pragim Tech

    </div>

</body>

</html>

The following code finds the total number of method calls waiting in the queue. While the first call to animate method is being executed, the other calls are added to the queue and waiting to be executed one after the other in sequence.

$('#myDiv1')

    .animate({ 'width': 500 }, 1500)

    .queue(function () {

        console.log('Queued calls = ' +

            $(this).queue('fx').length); $(this).dequeue();

    })

    .animate({ 'padding': 20 }, 1500)

    .animate({ 'font-size': 50 }, 1500)

    .animate({ 'border-width': 10 }, 1500)

    .animate({ 'opacity': 1 }, 1500)

    .queue(function () {

        console.log('Queued calls = ' +

            $(this).queue('fx').length); $(this).dequeue();

    });

**To globally disable all animations**  
$.fx.off = true or jQuery.fx.off = true   
  
The following example toggles animation on and off

<html>

<head>

    <style>

        .myDivClass {

            width: 150px;

            padding: 5px;

            font-size: 18px;

            border: 1px solid black;

            opacity: .4;

            background-color: red;

            color: white;

        }

    </style>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            $('#myButton').click(function () {

                jQuery.fx.off = $('#chkBox').is(':checked');

                $('#myDiv1')

                    .animate({ 'width': 500 }, 1500)

                    .animate({ 'padding': 20 }, 1500)

                    .animate({ 'font-size': 50 }, 1500)

                    .animate({ 'border-width': 10 }, 1500)

                    .animate({ 'opacity': 1 }, 1500)

                $('#myDiv2')

                    .animate({ 'width': 500 }, 1500)

                    .animate({ 'padding': 20 }, 1500)

                    .animate({ 'font-size': 50 }, 1500)

                    .animate({ 'border-width': 10 }, 1500)

                    .animate({ 'opacity': 1 }, 1500);

            });

        });

    </script>

</head>

<body style="font-family:Arial">

    <input type="button" id="myButton" value="Start Animation" />

    <input id="chkBox" type="checkbox" /> Turn off animations

    <br /><br />

    <div id="myDiv1" class="myDivClass">

        jQuery animations

    </div>

    <br />

    <div id="myDiv2" class="myDivClass">

        Pragim Tech

    </div>

</body>

</html>

If you want the calls to animate() to be executed simultaneously in parallel, then set queue option to false. Modify the jQuery code as shown below.

$('#myDiv1')

    .animate({ 'width': 500 }, { duration: 1500, queue: false })

    .animate({ 'padding': 20 }, { duration: 1500, queue: false })

    .animate({ 'font-size': 50 }, { duration: 1500, queue: false })

    .animate({ 'border-width': 10 }, { duration: 1500, queue: false })

    .animate({ 'opacity': 1 }, { duration: 1500, queue: false });

$('#myDiv2')

    .animate({ 'width': 500 }, { duration: 1500, queue: false })

    .animate({ 'padding': 20 }, { duration: 1500, queue: false })

    .animate({ 'font-size': 50 }, { duration: 1500, queue: false })

    .animate({ 'border-width': 10 }, { duration: 1500, queue: false })

    .animate({ 'opacity': 1 }, { duration: 1500, queue: false });

There are 2 variations of animate method. We discussed Variation 1 in [Part 46](http://csharp-video-tutorials.blogspot.com/2015/05/jquery-animate-function.html) of [jQuery tutorial](https://www.youtube.com/playlist?list=PL6n9fhu94yhVDV697uvHpavA3K_eWGQap). In the code snippet above we are using Variation 2.    
  
**Variation 1**  
.animate( properties [, duration ] [, easing ] [, complete ] )   
  
**Variation 2**  
.animate( properties, options )   
  
For the list of all additional options that you can pass to animate method please check <http://api.jquery.com/animate>   
  
An easier way to animate multiple css properties simultaneously in parallel, is to include all those css properties in a single JSON object. 

$('#myDiv1')

    .animate({

        'width': 500,

        'padding': 20,

        'font-size': 50,

        'border-width': 10,

        'opacity': 1

    }, 1500);

$('#myDiv2')

    .animate({

        'width': 500,

        'padding': 20,

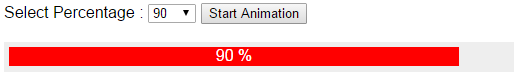
        'font-size': 50,

        'border-width': 10,

        'opacity': 1

    }, 1500);

### 48. Simple JQuery progress bar

<html>

<head>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            $('#myButton').click(function () {

                animateProgressBar($('#ddlPercentage').val());

            });

            function animateProgressBar(percentageCompleted) {

                $('#innerDiv').animate({

                    'width': (500 \* percentageCompleted) / 100

                }, 3000);

                $({ counter: 1 }).animate({ counter: percentageCompleted }, {

                    duration: 3000,

                    step: function () {

                        $('#innerDiv').text(Math.ceil(this.counter) + ' %');

                    }

                })

            }

        });

    </script>

</head>

<body style="font-family:Arial">

    Select Percentage :

    <select id="ddlPercentage">

        <option value="10">10</option>

        <option value="20">20</option>

        <option value="30">30</option>

        <option value="40">40</option>

        <option value="50">50</option>

        <option value="60">60</option>

        <option value="70">70</option>

        <option value="80">80</option>

        <option value="90">90</option>

        <option value="100">100</option>

    </select>

    <input type="button" id="myButton" value="Start Animation" />

    <br /><br />

    <div id="outerDiv" style="background-color:#EEEEEE;

         height:20px; width:500px; padding:5px">

        <div id="innerDiv" style="background-color:red; height:19px;

             width:0px; color:white; text-align:center">

        </div>

    </div>

</body>

</html>

### 49.optimize jquery progress bar

**At the moment the progress bar always counts from 1**. For example  
1. When you select 30%, it starts to count from 1 to 30 which is good.  
2. Now if you select 70%, it starts again from 1 and counts all the way till 70, instead of continuing to count from 30 to 70.  
  
**The following code counts from the previous point**. For example  
1. If you select 30% first, it starts to count from 1 to 30. Now if you select 90, it counts from 30 to 90.  
2. At this point, if you select 20%, it counts down from 90 to 20.

$(document).ready(function () {

    var currentPercentage = 0;

    var previousPercenage = 0;

    $('#myButton').click(function () {

        previousPercenage = currentPercentage;

        currentPercentage = $('#ddlPercent').val();

        animateProgressBar(previousPercenage, currentPercentage);

    });

    function animateProgressBar(previousPercenage, currentPercentage) {

        $('#innerDiv').animate({

            'width': (500 \* currentPercentage) / 100

        }, 2000);

        if (previousPercenage > currentPercentage)

            currentPercentage = currentPercentage - 1;

        $({ counter: previousPercenage }).animate({ counter: currentPercentage }, {

            duration: 2000,

            step: function () {

                $('#innerDiv').text(Math.ceil(this.counter) + ' %');

            }

        });

    }

});

The above code can be optimized as shown below. This optimization is suggested by **Aptem A**, one of our YouTube subscribers. This is great, thanks to him for his valuable contribution.

$(document).ready(function () {

    $('#myButton').click(function () {

        animateProgressBar($('#ddlPercent').val());

    });

    function animateProgressBar(currentPercentage) {

        $("#innerDiv").animate({ "width": (currentPercentage \* 500) / 100 }, {

            duration: 3000,

            step: function (now, fx) {

                $("#innerDiv").text(Math.ceil((now / 500) \* 100) + ' %');

            }

        });

    }

});

**step** **option**of the **animate function**can be used to define a function that gets called after each step of the animation. This method has 2 parameters - now & tween.

|  |  |
| --- | --- |
| **now** | contains the value being animated |
| **tween** | is a complex object and contains several properties. A few are listed below. For the complete list set a break point and inspect the object |

|  |  |
| --- | --- |
| **elem** | The DOM element being animated |
| **now** | The value the animation is currently at |
| **end** | The value the animation will end at |

jQuery animate method documentation  
<http://api.jquery.com/animate/>

### 50.jquery show hide password

**Here is what we want to achieve**  
1. When Show password checkbox IS NOT CHECKED, the password should be masked   
show password checkbox jquery   
  
2. When Show password checkbox IS CHECKED, the password should be in clear text and visible to the user   
jquery show hide password checkbox   
  
One of the simplest ways of achieving this is by changing the type attribute of the password field depending on the checked status of Show password checkbox  
1. If the Show password checkbox is CHECKED change the type to text  
2. If the Show password checkbox is NOT CHECKED change the type to password   
  
**The following is the HTML and jQuery code**

<html>

<head>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            $('#cbShowPassword').click(function () {

                $('#txtPassword').attr('type', $(this).is(':checked') ? 'text' : 'password');

            });

        });

    </script>

</head>

<body style="font-family:Arial">

    Password :

    <input type="password" id="txtPassword" />

    <input type="checkbox" id="cbShowPassword" />Show password

</body>

</html>

The problem with the above approach is that it does not work in IE 8 and earlier versions. This is because with IE8 and earlier versions the type attribute of input elements cannot be changed once set. The following code works in all browsers including IE8 and earlier versions.    
  
**When Show password checkbox is clicked**  
1. Retrieve the value the from the password textbox and store it in a variable for later use.  
2. Delete the password input filed.   
3. If the "Show password" checkbox is CHECKED, then add a new input filed of type text, else add a new input filed of password. In both the cases set the value attribute of the input element = the variable created in Step 1.

<html>

<head>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            $('#cbShowPassword').click(function () {

                var currentPassowrdFiled = $('#txtPassword');

                var currentPassword = currentPassowrdFiled.val();

                currentPassowrdFiled.remove();

                if ($(this).is(':checked')) {

                    $(this).before('<input type="text" id="txtPassword" value="'

                        + currentPassword + '">');

                }

                else {

                    $(this).before('<input type="password" id="txtPassword" value="'

                        + currentPassword + '">');

                }

            });

        });

    </script>

</head>

<body style="font-family:Arial">

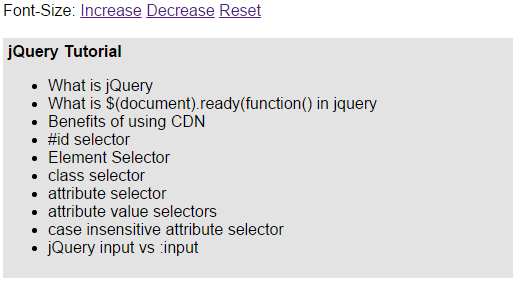
    Password :

    <input type="password" id="txtPassword" />

    <input type="checkbox" id="cbShowPassword" />Show password

</body>

</html>

**51.Increase decrease font size using jquery**  
 

<html>

<head>

    <style>

        .divClass {

            font-size: 16px;

            background-color: #E3E3E3;

            width: 500px;

            padding: 5px;

        }

    </style>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            $('#linkIncrease').click(function () {

                modifyFontSize('increase');

            });

            $('#linkDecrease').click(function () {

                modifyFontSize('decrease');

            });

            $('#linkReset').click(function () {

                modifyFontSize('reset');

            })

            function modifyFontSize(flag) {

                var divElement = $('#divContent');

                var currentFontSize = parseInt(divElement.css('font-size'));

                if (flag == 'increase')

                    currentFontSize += 3;

                else if (flag == 'decrease')

                    currentFontSize -= 3;

                else

                    currentFontSize = 16;

                divElement.css('font-size', currentFontSize);

            }

        });

    </script>

</head>

<body style="font-family:Arial">

    Font-Size:

    <a id="linkIncrease" href="#">Increase</a>

    <a id="linkDecrease" href="#">Decrease</a>

    <a id="linkReset" href="#">Reset</a>

    <br /><br />

    <div id="divContent" class="divClass">

        <b>jQuery Tutorial</b>

        <ul>

            <li>What is jQuery</li>

            <li>What is $(document).ready(function() in jquery</li>

            <li>Benefits of using CDN</li>

            <li>#id selector</li>

            <li>Element Selector</li>

            <li>class selector</li>

            <li>attribute selector</li>

            <li>attribute value selectors</li>

            <li>case insensitive attribute selector</li>

            <li>jQuery input vs :input</li>

        </ul>

    </div>

</body>

</html>

### 52.jquery floating div

We want the div element in the sidebar to be floating and always visible as we scroll down the page.   
   
  
**Example :** In this example we are using position() and scrollTop() functions. The object returned by position() function has top and left properties, which can be used to know the current top and left positions (coordinates). We are using this function to find the top position of the div element that we want to keep floating as we scroll down. To get the current vertical position of the scroll bar, we are using scrollTop() function.    
  
As we scroll and when the current vertical position of the scroll bar becomes GREATER THAN the top position of the div element, then we want the div element to **start floating**. To do this set **position**style to **fixed**. A fixed position element is positioned relative to the browser window. So as you scroll down it will be floating in the browser window.   
  
If the current vertical position of the scroll bar becomes LESS THAN the top position of the div element, then we don't want the div element to float, so we set **position**style to **relative**. A relative position element is positioned relative to itself. So if you set **position**to **relative**and **top**to **0**, it will continue to stay where it is without floating. 

<html>

<head>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            var floatingDiv = $("#divfloating");

            var floatingDivPosition = floatingDiv.position();

            $(window).scroll(function () {

                var scrollPosition = $(window).scrollTop();

                if (scrollPosition >= floatingDivPosition.top) {

                    floatingDiv.css({

                        'position': 'fixed',

                        'top': 3

                    });

                } else {

                    floatingDiv.css({

                        'position': 'relative',

                        'top': 0

                    });

                }

            });

        });

    </script>

</head>

<body style="font-family:Arial;">

    <table align="center" border="1" style="border-collapse:collapse">

        <tr>

            <td style="width:500px">

                Main Page Content

            </td>

            <td style="width:150px; vertical-align:top">

                Side panel content

                <br /><br />

                <div id="divfloating" style="background-color:silver;

                                             width:150px; height:150px">

                    Floating Div - Keeps floating as you scroll down the page

                </div>

            </td>

        </tr>

    </table>

</body>

</html>

### 53.\_ jquery ajax load

**What is AJAX**  
**AJAX** stands for **Asynchronous JavaScript and XML**, and allow parts of the page to be updated without having to reload the entire page.    
  
**Syntax**  
load( url [, data ] [, complete ] ) 

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| url | Required. URL to which the request is sent |
| data | Optional. A JSON object or string that is sent to the server along with the request |
| complete | A callback function that is called when the request completes |

The following example loads HTML data from the server. When a text box receives focus, the help text associated with that field is loaded from the server and displayed. When the focus is lost the help text disappears.   
  
   
  
**HtmlPage1.html**

<html>

<head>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            var textBoxes = $('input[type="text"]');

            textBoxes.focus(function () {

                var helpDiv = $(this).attr('id') + 'HelpDiv';

                $('#' + helpDiv).load('Help.html #' + helpDiv);

            });

            textBoxes.blur(function () {

                var helpDiv = $(this).attr('id') + 'HelpDiv';

                $('#' + helpDiv).html('');

            });

        });

    </script>

</head>

<body style="font-family:Arial">

    <table>

        <tr>

            <td>First Name</td>

            <td><input id="firstName" type="text" /></td>

            <td><div id="firstNameHelpDiv"></div></td>

        </tr>

        <tr>

            <td>Last Name</td>

            <td><input id="lastName" type="text" /></td>

            <td><div id="lastNameHelpDiv"></div></td>

        </tr>

        <tr>

            <td>Email</td>

            <td><input id="email" type="text" /></td>

            <td><div id="emailHelpDiv"></div></td>

        </tr>

        <tr>

            <td>Income</td>

            <td><input id="income" type="text" /></td>

            <td><div id="incomeHelpDiv"></div></td>

        </tr>

    </table>

</body>

</html>

**Help.html**

<div id="firstNameHelpDiv">

    Your fisrt name as it appears in passport

</div>

<div id="lastNameHelpDiv">

    Your last name as it appears in passport

</div>

<div id="emailHelpDiv">

    Your email address for communication

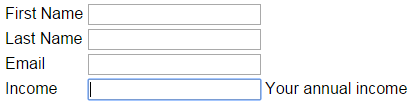
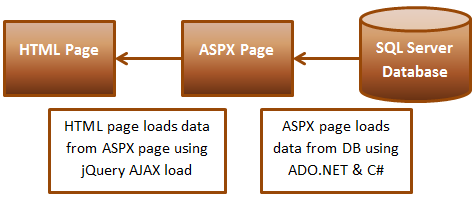
</div>

<div id="incomeHelpDiv">

    Your annual income

</div>

54.jquery ajax load aspx page

When a text box receives focus, the help text associated with that field should be loaded from the server and displayed. When the focus is lost the help text disappears. The help text is stored in the database.   
   
  
We will be using the jquery ajax load function to achieve this. Here is how this is going to work  
1. The ASPX page loads data from the SQL Server database using ADO.NET and C#  
2. The HTML page loads HTML data from the ASPX page using jQuery AJAX load function   
   
  
**Step 1 :** Create SQL Server table and insert helptext data

Create table tblHelpText

(

     HelpTextKey nvarchar(50) primary key,

     HelpText nvarchar(250)

)

GO

Insert into tblHelpText values

('firstName','Your fisrt name as it appears in passport')

Insert into tblHelpText values

('lastName','Your last name as it appears in passport')

Insert into tblHelpText values

('email','Your email address for communication')

Insert into tblHelpText values

('income','Your annual income')

**Step 2 :**Create stored procedure that the ASPX page will call to get helptext data from the database

Create procedure spGetHelpTextByKey

@HelpTextKey nvarchar(50)

as

Begin

     Select HelpText from tblHelpText where HelpTextKey=@HelpTextKey

End

**Step 3 :** Create new asp.net web application project. Name it Demo.   
  
**Step 4 :** Include a connection string in the web.config file to your database.

<add name="DBCS"

      connectionString="server=.;database=SampleDB;integrated security=SSPI" />

**Step 5 :** Add a new WebForm. Name it GetHelpText.aspx. Copy and paste the following HTML in the WebForm.

<%@ Page Language="C#" AutoEventWireup="true"

         CodeBehind="GetHelpText.aspx.cs" Inherits="Demo.GetHelpText" %>

<div id="divResult" runat="server"></div>

**Step 6 :** Copy and paste the followng code in the code-behind file

using System;

using System.Configuration;

using System.Data;

using System.Data.SqlClient;

namespace Demo

{

    public partial class GetHelpText : System.Web.UI.Page

    {

        protected void Page\_Load(object sender, EventArgs e)

        {

            string helpTextKey = Request["HelpTextKey"];

            divResult.InnerText = GetHelpTextByKey(helpTextKey);

        }

        private string GetHelpTextByKey(string key)

        {

            string helpText = string.Empty;

            string cs = ConfigurationManager.ConnectionStrings["DBCS"].ConnectionString;

            using (SqlConnection con = new SqlConnection(cs))

            {

                SqlCommand cmd = new SqlCommand("spGetHelpTextByKey", con);

                cmd.CommandType = CommandType.StoredProcedure;

                SqlParameter parameter = new SqlParameter("@HelpTextKey", key);

                cmd.Parameters.Add(parameter);

                con.Open();

                helpText = cmd.ExecuteScalar().ToString();

            }

            return helpText;

        }

    }

}

**Step 7 :** Add an HTML page to the ASP.NET project. Copy and paste the following HTML and jQuery code

<html>

<head>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            var textBoxes = $('input[type="text"]');

            textBoxes.focus(function () {

                var helpDiv = $(this).attr('id');

                $('#' + helpDiv + 'HelpDiv').load('GetHelpText.aspx', { HelpTextKey: helpDiv });

            });

            textBoxes.blur(function () {

                var helpDiv = $(this).attr('id') + 'HelpDiv';

                $('#' + helpDiv).html('');

            });

        });

    </script>

</head>

<body style="font-family:Arial">

    <table>

        <tr>

            <td>First Name</td>

            <td><input id="firstName" type="text" /></td>

            <td><div id="firstNameHelpDiv"></div></td>

        </tr>

        <tr>

            <td>Last Name</td>

            <td><input id="lastName" type="text" /></td>

            <td><div id="lastNameHelpDiv"></div></td>

        </tr>

        <tr>

            <td>Email</td>

            <td><input id="email" type="text" /></td>

            <td><div id="emailHelpDiv"></div></td>

        </tr>

        <tr>

            <td>Income</td>

            <td><input id="income" type="text" /></td>

            <td><div id="incomeHelpDiv"></div></td>

        </tr>

    </table>

</body>

</html>

### 55. jquery load callback function

**Syntax**  
.load( url [, data ] [, complete ] )   
  
complete parameter is a callback function that is executed when the request completes. The callback function has 3 parameters.  
  
CallbackFunction(response, status, xhr)  
  
In the following example, the anonymous function that is passed as the third parameter to the load function is executed when the AJAX request completes.

<html>

<head>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            var textBoxes = $('input[type="text"]');

            textBoxes.focus(function () {

                var helpDiv = $(this).attr('id');

                $('#' + helpDiv + 'HelpDiv').load('GetHelpText.aspx', { HelpTextKey: helpDiv },

                      function (response, status, xhr) {

                          var errorMessage = 'status : ' + xhr.status + '<br/>';

                          errorMessage += 'status text : ' + xhr.statusText + '<br/>';

                          errorMessage += 'response : ' + response;

                          $('#divError').html(errorMessage);

                      });

            });

            textBoxes.blur(function () {

                var helpDiv = $(this).attr('id') + 'HelpDiv';

                $('#' + helpDiv).html('');

            });

        });

    </script>

</head>

<body style="font-family:Arial">

    <table>

        <tr>

            <td>First Name</td>

            <td><input id="firstName" type="text" /></td>

            <td><div id="firstNameHelpDiv"></div></td>

        </tr>

        <tr>

            <td>Last Name</td>

            <td><input id="lastName" type="text" /></td>

            <td><div id="lastNameHelpDiv"></div></td>

        </tr>

        <tr>

            <td>Email</td>

            <td><input id="email" type="text" /></td>

            <td><div id="emailHelpDiv"></div></td>

        </tr>

        <tr>

            <td>Income</td>

            <td><input id="income" type="text" /></td>

            <td><div id="incomeHelpDiv"></div></td>

        </tr>

    </table>

    <br />

    <div id="divError"></div>

</body>

</html>

If you want the code in the callback function to execute only if these is an error, you can do so using status parameter as shown below.

$('#' + helpDiv + 'HelpDiv').load('GetHelpText.aspx', { HelpTextKey: helpDiv },

    function (response, status, xhr) {

        if (status == 'error') {

            var errorMessage = 'status : ' + xhr.status + '<br/>';

            errorMessage += 'status text : ' + xhr.statusText + '<br/>';

            errorMessage += 'response : ' + response;

            $('#divError').html(errorMessage);

        }

    });

### 56.jquery ajax get function

We discussed how to make AJAX requests using jQuery load() function in [Part 53](http://csharp-video-tutorials.blogspot.com/2015/05/jquery-ajax-load.html) and [Part 54](http://csharp-video-tutorials.blogspot.com/2015/05/jquery-ajax-load-aspx-page.html). Let us rewrite the example we worked with in [Part 54](http://csharp-video-tutorials.blogspot.com/2015/05/jquery-ajax-load-aspx-page.html), using jQuery AJAX get() function.   
  
Please change the following jquery code in HtmlPage1.html

$('#' + helpDiv + 'HelpDiv').load('GetHelpText.aspx', { HelpTextKey: helpDiv });

AS SHOWN BELOW

$.get('GetHelpText.aspx', { HelpTextKey: helpDiv }, function (response) {

    $('#' + helpDiv + 'HelpDiv').html(response);

});

**What is the difference between jQuery get and load functions**  
1. jQuery load function can be used to load only the HTML data from the server, where as jquery get function can be used to load any type of data (xml, json, script, or html).  
  
2. jQuery load function may issue a get or post request depending on whether the data parameter is specified or not. POST method is used if data is provided, otherwise, GET is used. On the other hand jquery get method always issues a GET request.  
  
3. With load function we specify a selector first. The HTML content retrieved from the server is automatically inserted into the DOM elements matched by the selector.

$('selector').load('url', 'data');

With get function we do not specify any selector first, instead we will make use of the callback function to insert the data received from the server into the DOM elements.

$.get('url', 'data', function (response) {

    $('selector').html(response);

});

In our next video, we will discuss **how to load json data from the server using jquery get function**.

### 57. load json data using jquery ajax

**Syntax**  
$.get( url [, data ] [, success ] [, dataType ] )  
  
**dataType** parameter specifies the type of data expected from the server. The dataType can be xml, json, script, or html. By default jQuery makes an intelligent guess.   
  
The following steps modify the example we worked with in Part 56, so that HtmlPage1.html will be able to retrieve and display JSON data from the server.  
  
**Step 1 :** Modify GetHelpText.aspx to return JSON data. Remove the following div element from GetHelpText.aspx.

<div id="divResult" runat="server"></div>

**Step 2 :** Modify the code in GetHelpText.aspx.cs as shown below.

using System;

using System.Configuration;

using System.Data;

using System.Data.SqlClient;

using System.Web.Script.Serialization;

namespace Demo

{

    public partial class GetHelpText : System.Web.UI.Page

    {

        protected void Page\_Load(object sender, EventArgs e)

        {

            JavaScriptSerializer js = new JavaScriptSerializer();

            string JSONString = js.Serialize(GetHelpTextByKey(Request["HelpTextKey"]));

            Response.Write(JSONString);

        }

        private HelpText GetHelpTextByKey(string key)

        {

            HelpText helpText = new HelpText();

            string cs = ConfigurationManager.ConnectionStrings["DBCS"].ConnectionString;

            using (SqlConnection con = new SqlConnection(cs))

            {

                SqlCommand cmd = new SqlCommand("spGetHelpTextByKey", con);

                cmd.CommandType = CommandType.StoredProcedure;

                SqlParameter parameter = new SqlParameter("@HelpTextKey", key);

                cmd.Parameters.Add(parameter);

                con.Open();

                helpText.Text = cmd.ExecuteScalar().ToString();

                helpText.Key = key;

            }

            return helpText;

        }

    }

    public class HelpText

    {

        public string Key { get; set; }

        public string Text { get; set; }

    }

}

**Step 3 :** Modify the jQuery code in HtmlPage1.html as shown below

$(document).ready(function () {

    var textBoxes = $('input[type="text"]');

    textBoxes.focus(function () {

        var helpDiv = $(this).attr('id');

        $.get('GetHelpText.aspx', { HelpTextKey: helpDiv }, function (response) {

            $('#' + helpDiv + 'HelpDiv').html(response.Text);

        }, 'json');

    });

    textBoxes.blur(function () {

        var helpDiv = $(this).attr('id') + 'HelpDiv';

        $('#' + helpDiv).html('');

    });

});

### 58.jquery ajax get xml data

The following steps modify the example we worked with in [Part 57](http://csharp-video-tutorials.blogspot.com/2015/06/load-json-data-using-jquery-ajax.html), so that HtmlPage1.html will be able to retrieve and display XML data from the server.   
  
**Step 1 :** Modify the code in GetHelpText.aspx.cs as shown below.

using System;

using System.Configuration;

using System.Data;

using System.Data.SqlClient;

using System.Xml.Serialization;

namespace Demo

{

    public partial class GetHelpText : System.Web.UI.Page

    {

        protected void Page\_Load(object sender, EventArgs e)

        {

            Response.ContentType = "text/xml";

            XmlSerializer xmlSerializer = new XmlSerializer(typeof(HelpText));

            xmlSerializer.Serialize(Response.OutputStream,

                GetHelpTextByKey(Request["HelpTextKey"]));

        }

        private HelpText GetHelpTextByKey(string key)

        {

            HelpText helpText = new HelpText();

            string cs = ConfigurationManager.ConnectionStrings["DBCS"].ConnectionString;

            using (SqlConnection con = new SqlConnection(cs))

            {

                SqlCommand cmd = new SqlCommand("spGetHelpTextByKey", con);

                cmd.CommandType = CommandType.StoredProcedure;

                SqlParameter parameter = new SqlParameter("@HelpTextKey", key);

                cmd.Parameters.Add(parameter);

                con.Open();

                helpText.Text = cmd.ExecuteScalar().ToString();

                helpText.Key = key;

            }

            return helpText;

        }

    }

    public class HelpText

    {

        public string Key { get; set; }

        public string Text { get; set; }

    }

}

**Step 2 :** Modify the jQuery code in HtmlPage1.html as shown below

$(document).ready(function () {

    var textBoxes = $('input[type="text"]');

    textBoxes.focus(function () {

        var helpDiv = $(this).attr('id');

        $.get('GetHelpText.aspx', { HelpTextKey: helpDiv }, function (response) {

            var jQueryXml = $(response);

            var textElement = jQueryXml.find("Text");

            $('#' + helpDiv + 'HelpDiv').html(textElement.text());

        }, 'xml');

    });

    textBoxes.blur(function () {

        var helpDiv = $(this).attr('id') + 'HelpDiv';

        $('#' + helpDiv).html('');

    });

});

### Please Note : 1. Use fiddler to inspect the data that is sent to and received from GetHelpText.aspx 2. Type the following URL in the browser to see the generated XML http://localhost:PortNumber/GetHelpText.aspx?HelpTextKey=firstName

### 59.jquery make a post request

This is continuation to [Part 58](http://csharp-video-tutorials.blogspot.com/2015/06/jquery-ajax-get-xml-data.html), please watch [Part 58](http://csharp-video-tutorials.blogspot.com/2015/06/jquery-ajax-get-xml-data.html) before proceeding.  
  
To make a get request we use **jquery get function**. jQuery load function can be used to make either a **get or post request**, depending on whether the data parameter is specified or not. POST method is used if data is provided, otherwise, GET is used.  
  
Another method that is available in jQuery to make a post request is **post method**.  
  
**jQuery post function syntax**  
$.post( url [, data ] [, success ] [, dataType ] )  
  
Please note that the syntax of post function is similar to get function  
  
To modify the example we worked with in [Part 58](http://csharp-video-tutorials.blogspot.com/2015/06/jquery-ajax-get-xml-data.html), to issue a post request simply replace get() function with post function.  
  
CHANGE THE FOLLOWING LINE FROM

$.get('GetHelpText.aspx', { HelpTextKey: helpDiv }, function (response) {

TO  
$.post('GetHelpText.aspx', { HelpTextKey: helpDiv }, function (response) {  
  
At this point if you fire fiddler and inspect the request you will find that a POST request is issued.  
  
**What is the difference between get and post**  
1. GET is designed for getting data from the server. POST is designed for modifying data on the server.  
  
2. Both GET and POST can be used to send data to the server.   
  
3. When you are sending data with a GET request, the data is appended to the URL as query strings. With POST request the data is included in the message body.  
  
4. There is a limit on how much data can be sent using a GET request due to the limit on the length of the data that can be passed as part of the URL. The size limitations associated with GET are different depending on the client and server software. So, If you're sending large amounts of data, use POST over GET. If it's just a small amount of data then you may use GET.  
  
5. In general, a GET request should have no side-effects, in the sense that it shouldn't modify data. Usually if we are using a GET request to send data to the server, it should be to retrieve an item with an id. For example, GetCustomer.aspx?id=1, will retrieve a customer with id=1.  
  
A lot of people keep asking **what is the difference between GET and POST in AJAX**  
Whether you make GET and POST request using AJAX or by some other means is irrelevant, the differences are still the same.

### 60.jquery ajax method

The other methods that are available in jquery to make ajax requests are **load**(), **get**() and **post**(). We discussed these methods in detail in the previous sessions of [jQuery tutorial](https://www.youtube.com/playlist?list=PL6n9fhu94yhVDV697uvHpavA3K_eWGQap) video series. All these methods are wrapper methods and use **ajax**() method under the hood. In Visual Studio, if you right click on any of these methods and select **"Go To Definition"** from the context menu, you can see that these methods call ajax() method.  
  
The wrapper methods are easier to use but they do not provide much flexibility. If you want to have complete control on configuring the ajax request use **ajax**() method.  
  
**Syntax of jquery ajax method**  
$.ajax( [ settings ] )  
  
**settings** is a JavaScript object that we use to configure the Ajax request. For the list of all available options please check the jquery ajax method documentation  
<http://api.jquery.com/jquery.ajax/>   
  
Let us now modify the example we worked with in [Part 59](http://csharp-video-tutorials.blogspot.com/2015/06/jquery-make-post-request.html), to use ajax() method instead of post() method.

$(document).ready(function () {

    var textBoxes = $('input[type="text"]');

    textBoxes.focus(function () {

        var helpDiv = $(this).attr('id');

        $.ajax({

            url: 'GetHelpText.aspx',

            data: { HelpTextKey: helpDiv },

            success: function (response, status, xhr) {

                var jQueryXml = $(response);

                var textElement = jQueryXml.find("Text");

                $('#' + helpDiv + 'HelpDiv').html(textElement.text());

            },

            dataType: 'xml',

            method: 'post'

        });

    });

    textBoxes.blur(function () {

        var helpDiv = $(this).attr('id') + 'HelpDiv';

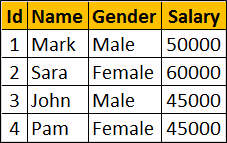
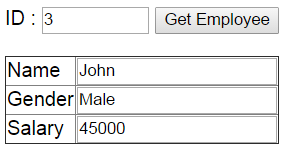
        $('#' + helpDiv).html('');

    });

});

In our next video, we will discuss **how to call an asp.net web service using jQuery AJAX and consume the XML data returned by the web service**.

### 61. Calling asp.net web services using jquery ajax

We want to retrieve data from the following database table **tblEmployee** using asp.net web service and jQuery AJAX   
  
  
  
The retrieved data should be displayed as shown below   
  
  
  
**Step 1 :** Create SQL Server table and insert employee data

Create table tblEmployee

(

     Id int primary key identity,

     Name nvarchar(50),

     Gender nvarchar(10),

     Salary int

)

GO

Insert into tblEmployee values ('Mark', 'Male', 50000)

Insert into tblEmployee values ('Sara', 'Female', 60000)

Insert into tblEmployee values ('John', 'Male', 45000)

Insert into tblEmployee values ('Pam', 'Female', 45000)

GO

**Step 2 :** Create a stored procedure to retrieve employee data by ID

Create procedure spGetEmployeeById

@Id int

as

Begin

     Select ID, Name, Gender, Salary

     from tblEmployee

     where ID = @Id

End

**Step 3 :** Create new asp.net web application project. Name it Demo.   
  
**Step 4 :** Include a connection string in the web.config file to your database.

<add name="DBCS"

      connectionString="server=.;database=SampleDB;integrated security=SSPI" />

**Step 5 :** Add a class file to the project. Name it Employee.cs. Copy and paste the following code.

namespace Demo

{

    public class Employee

    {

        public int ID { get; set; }

        public string Name { get; set; }

        public string Gender { get; set; }

        public int Salary { get; set; }

    }

}

**Step 6 :** Add a new WebService (ASMX). Name it EmployeeService.asmx. Copy and paste the following code.

using System;

using System.Configuration;

using System.Data;

using System.Data.SqlClient;

using System.Web.Services;

namespace Demo

{

    [WebService(Namespace = "http://tempuri.org/")]

    [WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1\_1)]

    [System.ComponentModel.ToolboxItem(false)]

    [System.Web.Script.Services.ScriptService]

    public class EmployeeService : System.Web.Services.WebService

    {

        [WebMethod]

        public Employee GetEmployeeById(int employeeId)

        {

            Employee employee = new Employee();

            string cs = ConfigurationManager.ConnectionStrings["DBCS"].ConnectionString;

            using (SqlConnection con = new SqlConnection(cs))

            {

                SqlCommand cmd = new SqlCommand("spGetEmployeeById", con);

                cmd.CommandType = CommandType.StoredProcedure;

                SqlParameter parameter = new SqlParameter();

                parameter.ParameterName = "@Id";

                parameter.Value = employeeId;

                cmd.Parameters.Add(parameter);

                con.Open();

                SqlDataReader rdr = cmd.ExecuteReader();

                while (rdr.Read())

                {

                    employee.ID = Convert.ToInt32(rdr["Id"]);

                    employee.Name = rdr["Name"].ToString();

                    employee.Gender = rdr["Gender"].ToString();

                    employee.Salary = Convert.ToInt32(rdr["Salary"]);

                }

            }

            return employee;

        }

    }

}

**Step 7 :** Add an HTML page to the ASP.NET project. Copy and paste the following HTML and jQuery code

<html>

<head>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            $('#btnGetEmployee').click(function () {

                var empId = $('#txtId').val();

                $.ajax({

                    url: 'EmployeeService.asmx/GetEmployeeById',

                    data: { employeeId: empId },

                    method: 'post',

                    dataType: 'xml',

                    success: function (data) {

                        var jQueryXml = $(data);

                        $('#txtName').val(jQueryXml.find('Name').text());

                        $('#txtGender').val(jQueryXml.find('Gender').text());

                        $('#txtSalary').val(jQueryXml.find('Salary').text());

                    },

                    error: function (err) {

                        alert(err);

                    }

                });

            });

        });

    </script>

</head>

<body style="font-family:Arial">

    ID : <input id="txtId" type="text" style="width:100px" />

    <input type="button" id="btnGetEmployee" value="Get Employee" />

    <br /><br />

    <table border="1" style="border-collapse:collapse">

        <tr>

            <td>Name</td>

            <td><input id="txtName" type="text" /></td>

        </tr>

        <tr>

            <td>Gender</td>

            <td><input id="txtGender" type="text" /></td>

        </tr>

        <tr>

            <td>Salary</td>

            <td><input id="txtSalary" type="text" /></td>

        </tr>

    </table>

</body>

</html>

In our next video we will discuss **how to call an asp.net web service that returns JSON data using jQuery AJAX.**

### 62.Handling json data returned from asp.net web services

This is continuation to [Part 61](http://csharp-video-tutorials.blogspot.com/2015/06/calling-aspnet-web-services-using.html), please watch [Part 61](http://csharp-video-tutorials.blogspot.com/2015/06/calling-aspnet-web-services-using.html) before proceeding.  
  
There are 2 approaches to make an asp.net web service return JSON data and consume it using jQuery AJAX.  
  
**Approach 1** : With Approach 1, the web service code does not change in any way.   
  
**We need to add/change the following options of the jquery request object.**  
  
1. Add **contentType** option and set it to **application/json; charset=utf-8** to specify that we will be sending a JSON string.  
  
2. Use **JSON.stringify()** method to convert the data you are sending to the server to a JSON string  
  
3. Change the **dataType** to **json**, to specify that you are expecting JSON data from the server  
  
4. Finally modify the **success** function to display Name, Gender and Salary property values from the JSON object.  
  
**Please note :** By defaut, the JSON object returned by the web service has a **property d**. So to retrieve Name property value, use data.d.Name.

$(document).ready(function () {

    $('#btnGetEmployee').click(function () {

        var empId = $('#txtId').val();

        $.ajax({

            url: 'EmployeeService.asmx/GetEmployeeById',

            contentType: "application/json; charset=utf-8",

            data: JSON.stringify({ employeeId: empId }),

            dataType: "json",

            method: 'post',

            success: function (data) {

                $('#txtName').val(data.d.Name);

                $('#txtGender').val(data.d.Gender);

                $('#txtSalary').val(data.d.Salary);

            },

            error: function (err) {

                alert(err);

            }

        });

    });

});

**Approach 2 :** With Approach 2 both the web service code and the jQuery code need to change.  
  
**Modify the ASP.NET web service as shown below to return JSON data**  
  
1. **Serialize** the employee object to JSON string using **JavaScriptSerializer** and write it to the response stream.  
  
2. Since the method is not returning anything set the return type of the method to **void**.

using System;

using System.Configuration;

using System.Data;

using System.Data.SqlClient;

using System.Web.Script.Serialization;

using System.Web.Script.Services;

using System.Web.Services;

namespace Demo

{

    [WebService(Namespace = "http://tempuri.org/")]

    [WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1\_1)]

    [System.ComponentModel.ToolboxItem(false)]

    [System.Web.Script.Services.ScriptService]

    public class EmployeeService : System.Web.Services.WebService

    {

        [WebMethod]

        public void GetEmployeeById(int employeeId)

        {

            Employee employee = new Employee();

            string cs = ConfigurationManager.ConnectionStrings["DBCS"].ConnectionString;

            using (SqlConnection con = new SqlConnection(cs))

            {

                SqlCommand cmd = new SqlCommand("spGetEmployeeById", con);

                cmd.CommandType = CommandType.StoredProcedure;

                SqlParameter parameter = new SqlParameter();

                parameter.ParameterName = "@Id";

                parameter.Value = employeeId;

                cmd.Parameters.Add(parameter);

                con.Open();

                SqlDataReader rdr = cmd.ExecuteReader();

                while (rdr.Read())

                {

                    employee.ID = Convert.ToInt32(rdr["Id"]);

                    employee.Name = rdr["Name"].ToString();

                    employee.Gender = rdr["Gender"].ToString();

                    employee.Salary = Convert.ToInt32(rdr["Salary"]);

                }

            }

            JavaScriptSerializer js = new JavaScriptSerializer();

            Context.Response.Write(js.Serialize(employee));

        }

    }

}

**Modify the jQuery code in HtmlPage1.html as shown below.**  
  
1. **contentType** option is no longer required, so you may remove it.  
  
2. You don't have to convert the data that you are sending to the server to JSON string, so you may remove **JSON.stringify()** method  
  
3. The JSON object returned from the server will not have **property d**, so you don't have to use d on the data object. For example, you can now retrieve Name property simply by using data.Name.

$(document).ready(function () {

    $('#btnGetEmployee').click(function () {

        var empId = $('#txtId').val();

        $.ajax({

            url: 'EmployeeService.asmx/GetEmployeeById',

            data: { employeeId: empId },

            dataType: "json",

            method: 'post',

            success: function (data) {

                $('#txtName').val(data.Name);

                $('#txtGender').val(data.Gender);

                $('#txtSalary').val(data.Salary);

            },

            error: function (err) {

                alert(err);

            }

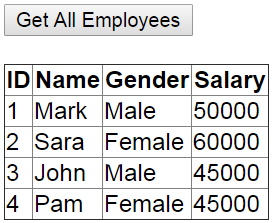
        });

    });

});

In our next video we will discuss **how to call an asp.net web service that returns JSON arrays using jQuery AJAX**.

### 63.Handling json arrays returned from asp.net web services with jquery

When we click **Get All Employees** button, we want to retrieve all the Employees from the database table and display on the page using jQuery.   
  
  
This is continuation to [Part 62](http://csharp-video-tutorials.blogspot.com/2015/06/handling-json-data-returned-from-aspnet.html), please watch [Part 62](http://csharp-video-tutorials.blogspot.com/2015/06/handling-json-data-returned-from-aspnet.html) before proceeding.  
  
**Step 1 :** Copy and paste the following code in EmployeeService.asmx.cs

using System;

using System.Collections.Generic;

using System.Configuration;

using System.Data.SqlClient;

using System.Web.Script.Serialization;

using System.Web.Script.Services;

using System.Web.Services;

namespace Demo

{

    [WebService(Namespace = "http://tempuri.org/")]

    [WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1\_1)]

    [System.ComponentModel.ToolboxItem(false)]

    [System.Web.Script.Services.ScriptService]

    public class EmployeeService : System.Web.Services.WebService

    {

        [WebMethod]

        public void GetAllEmployees()

        {

            List<Employee> listEmployees = new List<Employee>();

            string cs = ConfigurationManager.ConnectionStrings["DBCS"].ConnectionString;

            using (SqlConnection con = new SqlConnection(cs))

            {

                SqlCommand cmd = new SqlCommand("Select \* from tblEmployee", con);

                con.Open();

                SqlDataReader rdr = cmd.ExecuteReader();

                while (rdr.Read())

                {

                    Employee employee = new Employee();

                    employee.ID = Convert.ToInt32(rdr["Id"]);

                    employee.Name = rdr["Name"].ToString();

                    employee.Gender = rdr["Gender"].ToString();

                    employee.Salary = Convert.ToInt32(rdr["Salary"]);

                    listEmployees.Add(employee);

                }

            }

            JavaScriptSerializer js = new JavaScriptSerializer();

            Context.Response.Write(js.Serialize(listEmployees));

        }

    }

}

**Step 2 :** Copy and paste the following HTML and jQuery code in HtmlPage1.html

<html>

<head>

    <script src="jquery-1.11.2.js"></script>

    <script type="text/javascript">

        $(document).ready(function () {

            $('#btnGetAllEmployees').click(function () {

                $.ajax({

                    url: 'EmployeeService.asmx/GetAllEmployees',

                    dataType: "json",

                    method: 'post',

                    success: function (data) {

                        var employeeTable = $('#tblEmployee tbody');

                        employeeTable.empty();

                        $(data).each(function (index, emp) {

                            employeeTable.append('<tr><td>' + emp.ID + '</td><td>'

                                + emp.Name + '</td><td>' + emp.Gender

                                + '</td><td>' + emp.Salary + '</td></tr>');

                        });

                    },

                    error: function (err) {

                        alert(err);

                    }

                });

            });

        });

    </script>

</head>

<body style="font-family:Arial">

    <input type="button" id="btnGetAllEmployees" value="Get All Employees" />

    <br /><br />

    <table id="tblEmployee" border="1" style="border-collapse:collapse">

        <thead>

            <tr>

                <th>ID</th>

                <th>Name</th>

                <th>Gender</th>

                <th>Salary</th>

            </tr>

        </thead>

        <tbody></tbody>

    </table>

</body>

</html>