```
<html>
<head>
<script type="text/javascript" src="jquery.js"></script>
<script type="text/javascript">
$(document).ready(function(){
  $("button").click(function(){
    $("#test").hide();
  });
});
</script>
</head>

<body>
<h2>This is a heading</h2>
<p>This is a paragraph.</p>
<p id="test">This is another paragraph.</p>
<button>Click me</button>
</body>

</html>
```

Example : $("p ").hide()

```
<html>
<head>
<script type="text/javascript" src="jquery.js"></script>
<script type="text/javascript">
$(document).ready(function(){
  $("button").click(function(){
    $("p").hide();
  });
});
</script>
</head>

<body>
<h2>This is a heading</h2>
<p>This is a paragraph.</p>
<p>This is another paragraph.</p>
<button>Click me</button>
</body>
</html>
```

Example : $(".test").hide()

```
        alert("Hello world!");
    });
});
```

Now we can include **custom.js** file in our HTML file as follows:

```
<html>
<head>
<title>The jQuery Example</title>
   <script type="text/javascript"
   src="jquery.js"></script>
   <script type="text/javascript"
   src="/jquery/custom.js"></script>
</head>
<body>
<div id="newdiv">
Click on this to see a dialogue box.
</div>
</body>
</html>
```

Using Multiple Libraries:

You can use multiple libraries all together without conflicting each others. For example you can use **jQuery** and **MooTool** javascript libraries together.

jQuery - noConflict() Method

Many JavaScript libraries use $ as a function or variable name, just as jQuery does. In jQuery's case, $ is just an alias for jQuery.

Run **$.noConflict()** method to give control of the **$** variable back to whichever library first implemented it. This helps to make sure that jQuery doesn't conflict with the **$** object of other libraries.

Definition and Usage

The **noConflict()** method releases jQuery's control of the **$** variable.

This method can also be used to specify a new custom name for the jQuery variable.

Tip: This method is useful when other JavaScript libraries use the $ for their functions.

Syntax

$.noConflict(*removeAll*)

```
<html>
<head>
<script type="text/javascript" src="jquery.js"></script>
<script type="text/javascript">
$(document).ready(function(){
 $("button").click(function(){
  $(".test").hide();
 });
});
</script>
</head>
<body>

<h2 class="test">This is a heading</h2>
<p class="test">This is a paragraph.</p>
<p>This is another paragraph.</p>
<button>Click me</button>
</body>
</html>
```

NOTE: jQuery uses a combination of XPath and CSS selector syntax

The Document Ready Function

You might have noticed that all jQuery methods, in our examples, are inside a document.ready() function:

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

  // jQuery functions go here...

});
```

This is to prevent any jQuery code from running before the document is finished loading (is ready).

Here are some examples of actions that can fail if functions are run before the document is fully loaded:

- Trying to hide an element that doesn't exist
- Trying to get the size of an image that is not loaded

How to use Custom Scripts?

It is better to write our custom code in the custom JavaScript file : **custom.js**, as follows:

```
/* Filename: custom.js */
$(document).ready(function() {
  $("div").click(function() {
```

Parameter	Description
removeAll	Optional. A Boolean value that specifies whether or not to release jQuery's control of ALL jQuery variables (including "jQuery")

```
<html>
<head>
<script type="text/javascript" src="jquery.js"></script>
<script type="text/javascript">
var jq=$.noConflict();
jq(document).ready(function(){
 jq("button").click(function(){
  jq("p").hide();
 });
});
</script>
</head>

<body>
<h2>This is a heading</h2>
<p>This is a paragraph.</p>
<p>This is another paragraph.</p>
<button>Click me</button>
</body>
</html>
```

Here is simple way of avoiding any conflict:

```
// Import other library
// Import jQuery
$.noConflict();
// Code that uses other library's $ can follow here.
```

This technique is especially effective in conjunction with the .ready() method's ability to alias the jQuery object, as within the .ready() we can use $ if we wish without fear of conflicts later:

```
// Import other library
// Import jQuery
$.noConflict();
jQuery(document).ready(function($) {
// Code that uses jQuery's $ can follow here.
});
// Code that uses other library's $ can follow here.
DOM Element
```

jQuery - Basics

jQuery is a framework built using JavaScript capabilities. So you can use all the functions and other capabilities available in JavaScript.

This chapter would explain most basic concepts but frequently used in jQuery.

String:

A string in JavaScript is an immutable object that contains none, one or many characters.

Following are the valid examples of a JavaScript String:

```
"This is JavaScript String"
'This is JavaScript String'
'This is "really" a JavaScript String'
"This is 'really' a JavaScript String"
```

Numbers:

Numbers in JavaScript are double-precision 64-bit format IEEE 754 values. They are immutable, just as strings.

Following are the valid examples of a JavaScript Numbers:

```
5350
120.27
0.26
```

Boolean:

A boolean in JavaScript can be either **true** or **false**. If a number is zero, it defaults to false. If an empty string defaults to false:

Following are the valid examples of a JavaScript Boolean:

```
true       // true
false      // false
0          // false
1          // true
""         // false
"hello"    // true
```

Objects:

JavaScript supports Object concept very well. You can create an object using the object literal as follows:

```
var emp = {
   name: "Zara",
   age: 10
};
```

You can write and read properties of an object using the dot notation as follows:

```
// Getting object properties
emp.name  // ==> Zara
emp.age   // ==> 10

// Setting object properties
emp.name = "Daisy"  // <== Daisy
emp.age  =  20      // <== 20
```

Arrays:

You can define arrays using the array literal as follows:

```
var x = [];
var y = [1, 2, 3, 4, 5];
```

An array has a **length** property that is useful for iteration:

```
var x = [1, 2, 3, 4, 5];
for (var i = 0; i < x.length; i++) {
   // Do something with x[i]
 }
```

Functions:

A function in JavaScript can be either named or anonymous. A named function can be defined using *function* keyword as follows:

```
function named(){
  // do some stuff here
}
```

An anonymous function can be defined in similar way as a normal function but it would not have any name.

A anonymous function can be assigned to a variable or passed to a method as shown below.

```
var handler = function (){
  // do some stuff here
}
```

JQuery makes a use of anonymous functions very frequently as follows:

```
$(document).ready(function(){
  // do some stuff here
});
```

Arguments:

JavaScript variable *arguments* is a kind of array which has *length* property. Following example explains it very well:

```
function func(x){
  console.log(typeof x, arguments.length);
}
func();                 //==> "undefined", 0
func(1);                //==> "number", 1
func("1", "2", "3");    //==> "string", 3
```

The arguments object also has a *callee* property, which refers to the function you're inside of. For example:

```
function func() {
   return arguments.callee;
}
func();                 // ==> func
```

Scope:

The scope of a variable is the region of your program in which it is defined. JavaScript variable will have only two scopes.

- **Global Variables:** A global variable has global scope which means it is defined everywhere in your JavaScript code.
- **Local Variables:** A local variable will be visible only within a function where it is defined. Function parameters are always local to that function.

Within the body of a function, a local variable takes precedence over a global variable with the same name:

```
var myVar = "global";      // ==> Declare a global variable

function ( ) {
   var myVar = "local";    // ==> Declare a local variable
   document.write(myVar); // ==> local
}
```

Built-in Functions:

JavaScript comes along with a useful set of built-in functions. These methods can be used to manipulate Strings, Numbers and Dates.

Following are important JavaScript functions:

Method	Description
charAt()	Returns the character at the specified index.
concat()	Combines the text of two strings and returns a new string.
forEach()	Calls a function for each element in the array.
indexOf()	Returns the index within the calling String object of the first occurrence of the specified value, or -1 if not found.
length()	Returns the length of the string.
pop()	Removes the last element from an array and returns that element.
push()	Adds one or more elements to the end of an array and returns the new length of the array.
reverse()	Reverses the order of the elements of an array -- the first becomes the last, and the last becomes the first.
sort()	Sorts the elements of an array.
substr()	Returns the characters in a string beginning at the specified location through the specified number of characters.

toLowerCase()	Returns the calling string value converted to lower case.
toString()	Returns the string representation of the number's value.
toUpperCase()	Returns the calling string value converted to uppercase.

jQuery - Selectors

A jQuery Selector is a function which makes use of expressions to find out matching elements from a DOM based on the given criteria.

How to use Selectors?

The selectors are very useful and would be required at every step while using jQuery. They get the exact element that you want from your HTML document.

Following table lists down few basic selectors and explains them with examples.

Selector	Description
Name	Selects all elements which match with the given element **Name**.
#ID	Selects a single element which matches with the given **ID**
.Class	Selects all elements which match with the given **Class**.
Universal (*)	Selects all elements available in a DOM.
Multiple Elements E, F, G	Selects the combined results of all the specified selectors **E, F** or **G**.

jQuery - CSS Element Selector

Description:

The element selector selects all the elements that have a tag name of T.

Syntax:

Here is the simple syntax to use this selector:

```
$('tagname')
```

Parameters:

Here is the description of all the parameters used by this selector:

- **tagname:** Any standard HTML tag name like div, p, em, img , li etc.

Returns:

Like any other jQuery selector, this selector also returns an array filled with the found elements.

Example:

- **$('p')** selects all elements with a tag name of **p** in the document.
- **$('div')** selects all elements with a tag name of **div** in the document.

Following example would select all the divisions and display them one by one:

```
<html>
<head>
<title>The Selecter Example</title>
<script type="text/javascript" src="jquery.js">
</script>

<script type="text/javascript" language="javascript">

   $(document).ready(function() {
      /* This would select all the divisions */
      var divs = $("div");
      for( i=0; i<divs.length; i++ ){
         alert("Found Division: " + divs[i].innerHTML);
      }
   });
```

```
 </script>
 </head>
 <body>

<div class="div1" id="divid1">
  this is div one
</div>
<br />

<div class="div2" id="divid2">
  this is div two
</div>
<br />

<div class="div3" id="divid3">
  this is div three
</div>

 </body>
 </html>
```

```
<html>
<head>
<title>The Selecter Example</title>
<script type="text/javascript" src="jquery.js">
</script>

<script type="text/javascript" language="javascript">

  $(document).ready(function() {
    /* This would select all the divisions */
    $("button").click(function(){
          var divs = $("div");
          for(var i=0 ;i<divs.length ; i++){
                  if(i==0){
                          $(divs[i]).addClass("ybg");
                  }

                  if(i==1){
                          $(divs[i]).addClass("pbg");
                  }

                  if(i==2){

                          $(divs[i]).addClass("obg");
                  }

          }
```

```
 });
 });

</script>

<style type="text/css" >

.ybg{
 background-color : yellow;
}

.pbg{
 background-color : pink;
}

.obg{
 background-color : orange;
}


.div1{ color : red }
.div2{ color : green }
.div3{ color : blue }

</style>

</head>
<body>
<button> Click </button>

<div class="div1" id="divid1">
  this is div one
</div>
<br />

<div class="div2" id="divid2">
  this is div two
</div>
<br />

<div class="div3" id="divid3">
  this is div three
</div>

</body>
</html>
```

```
<html>
<head>
<title>The Selecter Example</title>
```

```
<script type="text/javascript" src="jquery.js">
</script>

<script type="text/javascript" language="javascript">

  $(document).ready(function() {

          $("button").click(function(){

                       var outputText = "";
                       $("div.div1, div.div2, div.div3").each(function(index){
                                outputText = outputText + index + "- "+$(this).text()+" <br/>";
                       });
                       $("div#output").html(outputText);

           });

           });

</script>

<style type="text/css" >

.div1{ color : red }
.div2{ color : green }
.div3{ color : blue }

</style>

</head>
<body>
<button> Click </button>

<div id="output" style="border : 2px solid red; height : 100px; width:500px;" >
</div>

<div class="div1" id="divid1">
 <p>this is div one </p>
</div>
<br />

<div class="div2" id="divid2">
 this is div two
</div>
<br />

<div class="div3" id="divid3">
 this is div three
</div>
```

```
</body>
</html>
```

jQuery - CSS Element ID Selector

Description:

The element ID selector selects a single element with the given id attribute.

Syntax:

Here is the simple syntax to use this selector:

```
$('#elementid')
```

Parameters:

Here is the description of all the parameters used by this selector:

- **elementid:** This would be an element ID. If the id contains any special characters like periods or colons you have to escape those characters with backslashes.

Returns:

Like any other jQuery selector, this selector also returns an array filled with the found element.

Example:

- **$('#myid')** selects a single element with the given id myid.
- **$('div#yourid')** selects a single division with the given id yourid.

Following example would select second division and display its content:

```
<html>
<head>
<title>The Selecter Example</title>
<script type="text/javascript" src="jquery.js">
</script>

<script type="text/javascript" language="javascript">

   $(document).ready(function() {
      var divs = $("#divid2");
```

```
        alert("Found Division: " + divs[0].innerHTML);
    });

</script>
</head>
<body>

<div class="div1" id="divid1">
  <p class="para1" id="pid1">This is first paragraph.</p>
  <p class="para2" id="pid2">This is second paragraph.</p>
</div>
<br />

<div class="div2" id="divid2">
  <p>This is second division of the DOM.</p>
</div>
<br />

<div class="div3" id="divid3">
  <p>This is a para inside third division</p>
</div>

</body>
</html>
```

jQuery - CSS Element Class Selector

Description:

The element class selector selects all the elements which match with the given class of the elements.

Syntax:

Here is the simple syntax to use this selector:

```
$('.classid')
```

Parameters:

Here is the description of all the parameters used by this selector:

- **classid:** This is class ID available in the document.

Returns:

Like any other jQuery selector, this selector also returns an array filled with the found elements.

Example:

- **$('.big')** selects all the elements with the given class ID **big**.
- **$('p.small')** selects all the paragraphs with the given class ID **small**.
- **$('.big.small')** selects all the elements with a class of **big** and **small**.

Following example would select all divisions with class **.big** and display its content:

```
<html>
<head>
<title>The Selecter Example</title>
<script type="text/javascript" src="jquery.js">
</script>

<script type="text/javascript" language="javascript">

  $(document).ready(function() {
      var divs = $(".big");
      for( i=0; i<divs.length; i++ ){
         alert("Found Division: " + divs[i].innerHTML);
      }
   });

</script>
</head>
<body>

<div class="big" id="divid1">
  <p class="para1" id="pid1">This is first paragraph.</p>
  <p class="para2" id="pid2">This is second paragraph.</p>
  <p class="para3" id="pid3">This is third paragraph.</p>
</div>
<br />

<div class="big" id="divid2">
  <p>This is second division of the DOM.</p>
  <p>This is second para inside second division.</p>
</div>
<br />

<div class="medium" id="divid3">
  <p>This is a para inside third division</p>
</div>

</body>
</html>
```

jQuery - CSS Universal Selector

Description:

The universal selector selects all the elements available in the document.

Syntax:

Here is the simple syntax to use this selector:

```
$('*')
```

Parameters:

Here is the description of all the parameters used by this selector:

- ***:** A symbolic star.

Returns:

Like any other jQuery selector, this selector also returns an array filled with the found elements.

Example:

- **$('*')** selects all the elements available in the document.

Following example would select all the elements available and would display them one by one:

```
<html>
<head>
<title>The Selecter Example</title>
<script type="text/javascript" src="jquery.js">
</script>

<script type="text/javascript" language="javascript">

  $(document).ready(function() {
      var elements = $("*");
      for( i=0; i<elements.length; i++ ){
         alert("Found element: " + elements[i].innerHTML);
      }
   });

</script>
</head>
```

```
<body>

<div class="big" id="divid1">
  <p class="para1" id="pid1">This is first paragraph.</p>
  <p class="para2" id="pid2">This is second paragraph.</p>
  <p class="para3" id="pid3">This is third paragraph.</p>
</div>
<br />

<div class="big" id="divid2">
  <p>This is second division of the DOM.</p>
  <p>This is second para inside second division.</p>
</div>
<br />

<div class="medium" id="divid3">
  <p>This is a para inside third division</p>
</div>

</body>
</html>
```

jQuery - CSS Multiple Elements E, F, G Selector

Description:

This Multiple Elements selector selects the combined results of all the specified selectors E, F or G.

You can specify any number of selectors to combine into a single result. Here order of the DOM elements in the jQuery object aren't necessarily identical.

Syntax:

Here is the simple syntax to use this selector:

```
$('E, F, G,....')
```

Parameters:

Here is the description of all the parameters used by this selector:

- **E:** Any valid selector
- **F:** Any valid selector
- **G:** Any valid selector
- ....

Returns:

Like any other jQuery selector, this selector also returns an array filled with the found elements.

Example:

- **$('div, p'):** selects all the elements matched by **div** or **p**.
- **$('p strong, .myclass'):** selects all elements matched by **strong** that are descendants of an element matched by **p** as well as all elements that have a class of **myclass**.
- **$('p strong, #myid'):** selects a single elements matched by **strong** that is descendant of an element matched by **p** as well as element whose id is **myid**.

Following example would select elements with class ID **big** and element with ID **divid3**:

```
<html>
<head>
<title>The Selecter Example</title>
<script type="text/javascript" src="jquery.js">
</script>

<script type="text/javascript" language="javascript">

  $(document).ready(function() {
      var elements = $(".big, #divid3");
      for( i=0; i<elements.length; i++ ){
         alert("Found element: " + elements[i].innerHTML);
      }
   });

</script>
</head>
<body>

<div class="big" id="divid1">
  <p class="para1" id="pid1">This is first paragraph.</p>
  <p class="para2" id="pid2">This is second paragraph.</p>
  <p class="para3" id="pid3">This is third paragraph.</p>
</div>
<br />

<div class="big" id="divid2">
  <p>This is second division of the DOM.</p>
  <p>This is second para inside second division.</p>
</div>
<br />

<div class="medium" id="divid3">
  <p>This is a para inside third division</p>
</div>

</body>
</html>
```

jQuery Selectors

Selector	Example	Selects
*	$("*")	All elements
#id	$("#lastname")	The element with id=lastname
.class	$(".intro")	All elements with class="intro"
element	$("p")	All p elements
.*class*.*class*	$(".intro.demo")	All elements with the classes "intro" and "demo"
:first	$("p:first")	The first p element
:last	$("p:last")	The last p element
:even	$("tr:even")	All even tr elements
:odd	$("tr:odd")	All odd tr elements
:eq(*index*)	$("ul li:eq(3)")	The fourth element in a list (index starts at 0)
:gt(*no*)	$("ul li:gt(3)")	List elements with an index greater than 3
:lt(*no*)	$("ul li:lt(3)")	List elements with an index less than 3
:not(*selector*)	$("input:not(:empty)")	All input elements that are not empty
:header	$(":header")	All header elements h1, h2 ...
:animated	$(":animated")	All animated elements
:contains(*text*)	$(":contains(sekharit)")	All elements which contains the text
:empty	$(":empty")	All elements with no child (elements) nodes
:hidden	$("p:hidden")	All hidden p elements

:visible	$("table:visible")	All visible tables
s1,s2,s3	$("th,td,.intro")	All elements with matching selectors
[attribute]	$("[href]")	All elements with a href attribute
[attribute=value]	$("[href='default.htm']")	All elements with a href attribute value equal to "default.htm"
[attribute!=value]	$("[href!='default.htm']")	All elements with a href attribute value not equal to "default.htm"
[attribute$=value]	$("[href$='.jpg']")	All elements with a href attribute value ending with ".jpg"
:input	$(":input")	All input elements
:text	$(":text")	All input elements with type="text"
:password	$(":password")	All input elements with type="password"
:radio	$(":radio")	All input elements with type="radio"
:checkbox	$(":checkbox")	All input elements with type="checkbox"
:submit	$(":submit")	All input elements with type="submit"
:reset	$(":reset")	All input elements with type="reset"
:button	$(":button")	All input elements with type="button"
:image	$(":image")	All input elements with type="image"
:file	$(":file")	All input elements with type="file"
:enabled	$(":enabled")	All enabled input elements
:disabled	$(":disabled")	All disabled input elements
:selected	$(":selected")	All selected input elements

:checked	$(":checked")	All checked input elements

Example: :first, :last

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

        $("button").click(function(){
                $("p:first").addClass("obg");
                $("p:last").addClass("pbg");
        });

        });
```

Example: :first, :last like example with our own logic

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

        $("button").click(function(){
                var count = $("p").size();
                $("p").each(function(index){
                        if(index == 0){
                                $(this).addClass("obg");
                        }

                        if(index == (count-1)){
                                $(this).addClass("pbg");
                        }
                } );
        });

        });
```

Example: :even, :odd

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

        $("button").click(function(){
                $("tr:even").addClass("obg");
                $("tr:odd").addClass("pbg");
        });

        });
```

Similar to above syntax and examples, following examples would give you understanding on using different type of other useful selectors:

- **$('*'):** This selector selects all elements in the document.
- **$("p > *"):** This selector selects all elements that are children of a paragraph element.
- **$("#specialID"):** This selector function gets the element with id="specialID".
- **$(".specialClass"):** This selector gets all the elements that have the class of *specialClass*.
- **$("li:not(.myclass)"):** Selects all elements matched by <li> that do not have class="myclass".
- **$("a#specialID.specialClass"):** This selector matches links with an id of *specialID* and a class of *specialClass*.
- **$("p a.specialClass"):** This selector matches links with a class of *specialClass* declared within <p> elements.
- **$("ul li:first"):** This selector gets only the first <li> element of the <ul>.
- **$("#container p"):** Selects all elements matched by <p> that are descendants of an element that has an id of *container*.
- **$("li > ul"):** Selects all elements matched by <ul> that are children of an element matched by <li>
- **$("strong + em"):** Selects all elements matched by <em> that immediately follow a sibling element matched by <strong>.
- **$("p ~ ul"):** Selects all elements matched by <ul> that follow a sibling element matched by <p>.
- **$("code, em, strong"):** Selects all elements matched by <code> or <em> or <strong>.
- **$("p strong, .myclass"):** Selects all elements matched by <strong> that are descendants of an element matched by <p> as well as all elements that have a class of *myclass*.
- **$(":empty"):** Selects all elements that have no children.
- **$("p:empty"):** Selects all elements matched by <p> that have no children.
- **$("div[p]"):** Selects all elements matched by <div> that contain an element matched by <p>.
- **$("p[.myclass]"):** Selects all elements matched by <p> that contain an element with a class of *myclass*.
- **$("a[@rel]"):** Selects all elements matched by <a> that have a rel attribute.
- **$("input[@name=myname]"):** Selects all elements matched by <input> that have a name value exactly equal to *myname*.
- **$("input[@name^=myname]"):** Selects all elements matched by <input> that have a name value beginning with *myname*.
- **$("a[@rel$=self]"):** Selects all elements matched by <p> that have a class value ending with *bar*
- **$("a[@href*=domain.com]"):** Selects all elements matched by <a> that have an href value containing domain.com.
- **$("li:even"):** Selects all elements matched by <li> that have an even index value.
- **$("tr:odd"):** Selects all elements matched by <tr> that have an odd index value.
- **$("li:first"):** Selects the first <li> element.
- **$("li:last"):** Selects the last <li> element.
- **$("li:visible"):** Selects all elements matched by <li> that are visible.
- **$("li:hidden"):** Selects all elements matched by <li> that are hidden.
- **$(":radio"):** Selects all radio buttons in the form.
- **$(":checked"):** Selects all checked boxex in the form.
- **$(":input"):** Selects only form elements (input, select, textarea, button).
- **$(":text"):** Selects only text elements (input[type=text]).
- **$("li:eq(2)"):** Selects the third <li> element
- **$("li:eq(4)"):** Selects the fifth <li> element
- **$("li:lt(2)"):** Selects all elements matched by <li> element before the third one; in other words, the first two <li> elements.

- **$("p:lt(3)"):** selects all elements matched by <p> elements before the fourth one; in other words the first three <p> elements.
- **$("li:gt(1)"):** Selects all elements matched by <li> after the second one.
- **$("p:gt(2)"):** Selects all elements matched by <p> after the third one.
- **$("div/p"):** Selects all elements matched by <p> that are children of an element matched by <div>.
- **$("div//code"):** Selects all elements matched by <code>that are descendants of an element matched by <div>.
- **$("//p//a"):** Selects all elements matched by <a> that are descendants of an element matched by <p>
- **$("li:first-child"):** Selects all elements matched by <li> that are the first child of their parent.
- **$("li:last-child"):** Selects all elements matched by <li> that are the last child of their parent.
- **$(":parent"):** Selects all elements that are the parent of another element, including text.
- **$("li:contains(second)"):** Selects all elements matched by <li> that contain the text second.

You can use all the above selectors with any HTML/XML element in generic way. For example if selector **$("li:first")** works for <li> element then **$("p:first")** would also work for <p> element.

jQuery Callback Functions

A callback function is executed after the current animation is 100% finished.

jQuery Callback Functions

JavaScript statements are executed line by line. However, with animations, the next line of code can be run even though the animation is not finished. This can create errors.

To prevent this, you can create a callback function.

A callback function is executed after the current animation (effect) is finished.

jQuery Callback Example

Typical syntax: **$(selector).hide(speed,callback)**

The callback parameter is a function to be executed after the hide effect is completed:

Example with Callback

```
$("p").hide(1000,function(){
  alert("The paragraph is now hidden");
});
```

Without a callback parameter, the alert box is displayed before the hide effect is completed:

Example without Callback

```
$("p").hide(1000);
```

```
alert("The paragraph is now hidden");
```

Example:

```
<html>
<head>
<title>the title</title>
  <script type="text/javascript"   src="jquery.js"></script>
  <script type="text/javascript" language="javascript">
   $(document).ready(function() {
                $("button").click(function(){
                        $("div#abc").hide(2000, function(){
                                alert("Hiding is over");

                        } );
                } );
        });
  </script>
</head>
<body>
        <button>Click Me</button>
        <div id="abc" >
          <ul>
                <li>list item 1</li>
                <li>list item 2</li>
                <li>list item 3</li>
                <li>list item 4</li>
                <li>list item 5</li>
                <li>list item 6</li>
          </ul>
  </div>
</body>
</html>
```

jQuery - DOM Attributes

Some of the most basic components we can manipulate when it comes to DOM elements are the properties and attributes assigned to those elements.

Most of these attributes are available through JavaScript as DOM node properties. Some of the more common properties are:

- className
- tagName
- id
- href

- title
- rel
- src

Consider the following HTML markup for an image element:

```
<img id="myImage" src="image.gif" alt="An image"
class="someClass" title="This is an image"/>
```

In this element's markup, the tag name is img, and the markup for id, src, alt, class, and title represents the element's attributes, each of which consists of a name and a value.

jQuery gives us the means to easily manipulate an element's attributes and gives us access to the element so that we can also change its properties.

Get Attribute Value:

The **attr()** method can be used to either fetch the value of an attribute from the first element in the matched set or set attribute values onto all matched elements.

Example: selector. attr(name)

Following is a simple example which fetches title attribute of <em> tag and set <div id="divid"> value with the same value:

```
<html>
<head>
<title>the title</title>
   <script type="text/javascript"
   src="jquery.js"></script>
   <script type="text/javascript" language="javascript">

   $(document).ready(function() {
      var title = $("em").attr("title");
      $("#divid").text(title);
   });

   </script>
</head>
<body>
   <div>
      <em title="Bold and Brave">This is first paragraph.</em>
      <p id="myid">This is second paragraph.</p>
      <div id="divid"></div>
   </div>
</body>
</html>
```

ɛt Attribute Value:

The **attr(name, value)** method can be used to set the named attribute onto all elements in the wrapped set using the passed value.

Example: selector. attr(name, value)

Following is a simple example which set **src** attribute of an image tag to a correct location:

```
<html>
<head>
<title>the title</title>
   <script type="text/javascript"
   src="jquery.js"></script>
   <script type="text/javascript" language="javascript">

   $(document).ready(function() {
      $("#myimg").attr("src", "/images/jquery.jpg");
   });

   </script>
</head>
<body>
   <div>
      <img id="myimg" src="/wongpath.jpg" alt="Sample image" />
   </div>
</body>
</html>
```

Example: selector.attr({ json })

```
<html>
<head>
<title>the title</title>
  <script type="text/javascript"   src="jquery.js"></script>
  <script type="text/javascript" language="javascript">

  $(document).ready(function() {
                $("button").click(function(){
                        $("div#abc").attr(
                                {
                                        title :"This is Title",
                                        style :"color : red; border: 1px solid green"

                                }
                        );
                });
  });

  </script>
</head>
<body>
```

```
  <button>Give default value</button>
  <div id="abc" >
        This is some div
  </div>
</body>
</html>
```

Applying Styles:

The **addClass(classes)** method can be used to apply defined style sheets onto all the matched elements. You can specify multiple classes separated by space.

Example: selector.addClass(classs)

Following is a simple example which set **src** attribute of an image tag to a correct location:

```
<html>
<head>
<title>the title</title>
   <script type="text/javascript"
   src="jquery.js"></script>
   <script type="text/javascript" language="javascript">

   $(document).ready(function() {
      $("em").addClass("selected");
      $("#myid").addClass("highlight");
   });

   </script>
   <style>
      .selected { color:red; }
      .highlight { background:yellow; }
  </style>
</head>
<body>
   <em title="Bold and Brave">This is first paragraph.</em>
   <p id="myid">This is second paragraph.</p>
</body>
</html>
```

Useful Attribute Methods:

Following table lists down few useful methods which you can use to manipulate attributes and properties:

Methods	Description

attr(properties)	Set a key/value object as properties to all matched elements.
attr(key, fn)	Set a single property to a computed value, on all matched elements.
removeAttr(name)	Remove an attribute from each of the matched elements.
hasClass(class)	Returns true if the specified class is present on at least one of the set of matched elements.
removeClass(class)	Removes all or the specified class(es) from the set of matched elements.
toggleClass(class)	Adds the specified class if it is not present, removes the specified class if it is present.
html( )	Get the html contents (innerHTML) of the first matched element.
html(val)	Set the html contents of every matched element.
text( )	Get the combined text contents of all matched elements.
text(val)	Set the text contents of all matched elements.
val( )	Get the input value of the first matched element.
val(val)	Set the value attribute of every matched element if it is called on <input> but if it is called on <select> with the passed <option> value then passed option would be selected, if it is called on check box or radio box then all the matching check box and radiobox would be checked.

Similar to above syntax and examples, following examples would give you understanding on using various attribute methods in different situation:

- **$("#myID").attr("custom") :** This would return value of attribute *custom* for the first element matching with ID myID.
- **$("img").attr("alt", "Sample Image"):** This sets the **alt** attribute of all the images to a new value "Sample Image".

- **$("input").attr({ value: "", title: "Please enter a value" }); :** Sets the value of all <input> elements to the empty string, as well as sets the title to the string *Please enter a value.*
- **$("a[href^=http://]").attr("target","_blank"):** Selects all links with an href attribute starting with *http://* and set its target attribute to *_blank*
- **$("a").removeAttr("target") :** This would remove *target* attribute of all the links.
- **$("form").submit(function() {$(":submit",this).attr("disabled", "disabled");}); :** This would modify the disabled attribute to the value "disabled" while clicking Submit button.
- **$("p:last").hasClass("selected"):** This return true if last <p> tag has associated class*selected.*
- **$("p").text():** Returns string that contains the combined text contents of all matched <p> elements.
- **$("p").text("<i>Hello World</i>"):** This would set "<I>Hello World</I>" as text content of the matching <p> elements
- **$("p").html() :** This returns the HTML content of the all matching paragraphs.
- **$("div").html("Hello World") :** This would set the HTML content of all matching <div> to *Hello World.*
- **$("input:checkbox:checked").val() :** Get the first value from a checked checkbox
- **$("input:radio[name=bar]:checked").val():** Get the first value from a set of radio buttons
- **$("button").val("Hello") :** Sets the value attribute of every matched element <button>.
- **$("input").val("on") :** This would check all the radio or check box button whose value is "on".
- **$("select").val("Orange") :** This would select Orange option in a dropdown box with options Orange, Mango and Banana.
- **$("select").val("Orange", "Mango") :** This would select Orange and Mango options in a dropdown box with options Orange, Mango and Banana.

jQuery - DOM Traversing

jQuery is a very powerful tool which provides a variety of DOM traversal methods to help us select elements in a document randomly as well as in sequential method.

Most of the DOM Traversal Methods do not modify the jQuery object and they are used to filter out elements from a document based on given conditions.

Find Elements by index:

Consider a simple document with the following HTML content:

```
<html>
<head>
<title>the title</title>
</head>
<body>
   <div>
   <ul>
     <li>list item 1</li>
     <li>list item 2</li>
     <li>list item 3</li>
     <li>list item 4</li>
     <li>list item 5</li>
     <li>list item 6</li>
```

```
    </ul>
    </div>
</body>
</html>
```

- Above every list has its own index, and can be located directly by using **eq(index)** method as below example.
- Every child element starts its index from zero, thus, *list item 2* would be accessed by using **$("li").eq(1)** and so on.

Example: eq(index)

Following is a simple example which adds the color to second list item.

```
<html>
<head>
<title>the title</title>
   <script type="text/javascript"
   src="jquery.js"></script>
   <script type="text/javascript" language="javascript">

   $(document).ready(function() {
      $("li").eq(2).addClass("selected");
   });

   </script>
   <style>
      .selected { color:red; }
  </style>
</head>
<body>
   <div>
   <ul>
     <li>list item 1</li>
     <li>list item 2</li>
     <li>list item 3</li>
     <li>list item 4</li>
     <li>list item 5</li>
     <li>list item 6</li>
   </ul>
   </div>
</body>
</html>
```

Filtering out Elements:

The **filter(selector)** method can be used to filter out all elements from the set of matched elements that do not match the specified selector(s). The *selector* can be written using any selector syntax.

Example: filter(selector)

Following is a simple example which applies color to the lists associated with middle class:

```
<html>
<head>
<title>the title</title>
   <script type="text/javascript"   src="jquery.js"></script>
   <script type="text/javascript" language="javascript">

   $(document).ready(function() {

        $("button").click(function() {
               $("li").filter(".middle").addClass("selected");

   });
   });

   </script>
   <style>
      .selected { color:red; }
  </style>
</head>
<body>
<button>Click Me </button>
   <div>
   <ul>
     <li class="top">list item 1</li>
     <li class="top">list item 2</li>
     <li class="middle">list item 3</li>
     <li class="middle">list item 4</li>
     <li class="bottom">list item 5</li>
     <li class="bottom">list item 6</li>
   </ul>
   </div>
</body>
</html>
```

Locating Descendent Elements :

The **find(selector)** method can be used to locate all the descendent elements of a particular type of elements. The *selector* can be written using any selector syntax.

Example: find(selector)

Following is an example which selects all the <span> elements available inside different <p> elements:

```
<html>
<head>
<title>the title</title>
   <script type="text/javascript"   src="jquery.js"></script>
```

```
    <script type="text/javascript" language="javascript">

    $(document).ready(function() {

         $("button").click(function() {
                $("p").find("span").addClass("selected");

    });
    });

    </script>
    <style>
       .selected { color:red; }
   </style>
</head>
<body>
<button>Click Me </button>
    <p>This is 1st paragraph and <span>THIS IS RED</span> </p>
    <p>This is 2nd paragraph and <span>THIS IS ALSO RED</span> </p>
    <p> this is 3rd paragraph <strong> I am strong </strong>  </p>
</body>
</html>
```

JQuery DOM Traversing Methods:

Following table lists down useful methods which you can use to filter out various elements from a list of DOM elements:

Selector	Description
eq(index)	Reduce the set of matched elements to a single element.
filter(selector)	Removes all elements from the set of matched elements that do not match the specified selector(s).
filter(fn)	Removes all elements from the set of matched elements that do not match the specified function.
is(selector)	Checks the current selection against an expression and returns true, if at least one element of the selection fits the given selector.
map(callback)	Translate a set of elements in the jQuery object into another set of values in a jQuery array (which may, or may not contain elements).

not(selector)	Removes elements matching the specified selector from the set of matched elements.
slice(start, [end])	Selects a subset of the matched elements.

Following table lists down other useful methods which you can use to locate various elements in a DOM:

Selector	Description
add(selector)	Adds more elements, matched by the given selector, to the set of matched elements.
andSelf( )	Add the previous selection to the current selection.
children([selector])	Get a set of elements containing all of the unique immediate children of each of the matched set of elements.
closest(selector)	Get a set of elements containing the closest parent element that matches the specified selector, the starting element included.
contents( )	Find all the child nodes inside the matched elements (including text nodes), or the content document, if the element is an iframe.
end( )	Revert the most recent 'destructive' operation, changing the set of matched elements to its previous state .
find(selector)	Searches for descendent elements that match the specified selectors.
next([selector])	Get a set of elements containing the unique next siblings of each of the given set of elements.
nextAll([selector])	Find all sibling elements after the current element.
offsetParent( )	Returns a jQuery collection with the positioned parent of the first matched element.

parent([selector])	Get the direct parent of an element. If called on a set of elements, parent returns a set of their unique direct parent elements.
parents([selector])	Get a set of elements containing the unique ancestors of the matched set of elements (except for the root element).
prev([selector])	Get a set of elements containing the unique previous siblings of each of the matched set of elements.
prevAll([selector])	Find all sibling elements in front of the current element.
siblings([selector])	Get a set of elements containing all of the unique siblings of each of the matched set of elements.

jQuery - CSS Methods

The jQuery library supports nearly all of the selectors included in Cascading Style Sheet (CSS) specifications 1 through 3, as outlined on the World Wide Web Consortium's site.

Using JQuery library developers can enhance their websites without worrying about browsers and their versions as long as the browsers have JavaScript enabled.

Most of the JQuery CSS Methods do not modify the content of the jQuery object and they are used to apply CSS properties on DOM elements.

Apply CSS Properties:

This is very simple to apply any CSS property using JQuery method **css(PropertyName, PropertyValue)**.

Here is the syntax for the method:

```
selector.css( PropertyName,. PropertyValue );
```

Here you can pass *PropertyName* as a javascript string and based on its value, *PropertyValue* could be string or integer.

Example:selector.css(properyName, properyValue)

Following is an example which adds font color to the second list item.

```
<html>
<head>
<title>the title</title>
   <script type="text/javascript"   src="jquery.js"></script>
   <script type="text/javascript" language="javascript">

   $(document).ready(function() {
               $("button").click(function(){
                       $("div#abc").css("border", "1px solid green");
               } );
   });

   </script>
   <style type="text/css" >
               .pbg { background-color : pink }
   </style>
</head>
<body>

        <button>Click Me</button>
   <div id="abc"  >
           This is some div
   </div>
</body>
</html>
```

Apply Multiple CSS Properties:

You can apply multiple CSS properties using a single JQuery method **CSS({key1:val1, key2:val2....)**. You can apply as many properties as you like in a single call.

Here is the syntax for the method:

```
selector.css( {key1:val1, key2:val2....keyN:valN})
```

Here you can pass key as property and val as its value as described above.

Example: selector.css({ json object})

Following is an example which adds font color as well as background color to the second list item.

```
<html>
<head>
<title>the title</title>
   <script type="text/javascript"   src="jquery.js"></script>
   <script type="text/javascript" language="javascript">

   $(document).ready(function() {
               $("button").click(function(){
                       $("div#abc").css(
                                   {
                                          border : "1px solid green",
                                          color : "red",
                                          width : "400px",
                                          height : "300px"
                                   }
                           );
                   } );
     });

     </script>
     <style type="text/css" >
                   .pbg { background-color : pink }
     </style>
  </head>
  <body>

          <button>Click Me</button>
     <div id="abc"  >
             This is some div
     </div>
  </body>
  </html>
```

Setting Element Width & Height:

The **width(val)** and **height(val)** method can be used to set the width and hieght respectively of any element.

Example: selector.width(val), selector.height(val)

Following is a simple example which sets the width of first division element where as rest of the elements have width set by style sheet:

```
  <html>
  <head>
  <title>the title</title>
     <script type="text/javascript"   src="jquery.js"></script>
     <script type="text/javascript" language="javascript">

     $(document).ready(function() {
                   $("div").click(function(){
                           $(this).width(100);
                           $(this).height(150);
                           $(this).css("background-color", "blue");

                   } );
     });

     </script>
     <style type="text/css" >

          div{
                width:70px;
                height:50px;
                float:left;
                margin:5px;
                background:red;
                cursor:pointer;
        }

   </style>
</head>
<body>

         <div>ONE</div>
         <div>TWO</div>
         <div>THREE</div>
         <div>FOUR</div>
         <div>FIVE</div>

</body>
</html>
```

Example : Function chaining

```
<html>
<head>
<title>the title</title>
  <script type="text/javascript"   src="jquery.js"></script>
  <script type="text/javascript" language="javascript">

  $(document).ready(function() {
                  $("button").click(function(){

                         $("div#abc").css("border", "1px solid green")
                                                .css("color", "red")
                                                .width("400px")
                                                .height("500px")
                                                .addClass("pbg");


                  });
  });

  </script>
  <style type="text/css" >
         .pbg { background-color : pink }
  </style>
</head>
<body>

<button>Click Me</button>
```

```
  <div id="abc"  >
            This is some div
  </div>
</body>
</html>
```

Example:selector.toggleClass(css-class)

```
<html>
<head>
<title>the title</title>
  <script type="text/javascript"   src="jquery.js"></script>
  <script type="text/javascript" language="javascript">

   function MyFocusBlur(tb){
                 $(tb).toggleClass("ybg");
  }

  </script>
  <style type="text/css" >
          .ybg{ background-color : yellow; }

  </style>
</head>
<body>

        First Name :  <input type="text" name="fname" onblur="MyFocusBlur(this)" onfocus="MyFocusBlur(this)" />
<br/>
        Last Name :  <input type="text" name="lname"  onblur="MyFocusBlur(this)" onfocus="MyFocusBlur(this)" />
<br/>
</body>
</html>
```

JQuery CSS Methods:

Following table lists down all the methods which you can use to play with CSS properties:

Method	Description
css(name)	Return a style property on the first matched element.
css(name, value)	Set a single style property to a value on all matched elements.
css(properties)	Set a key/value object as style properties to all matched elements.

height(val)	Set the CSS height of every matched element.
height( )	Get the current computed, pixel, height of the first matched element.
innerHeight( )	Gets the inner height (excludes the border and includes the padding) for the first matched element.
innerWidth( )	Gets the inner width (excludes the border and includes the padding) for the first matched element.
offset( )	Get the current offset of the first matched element, in pixels, relative to the document
offsetParent( )	Returns a jQuery collection with the positioned parent of the first matched element.
outerHeight([margin])	Gets the outer height (includes the border and padding by default) for the first matched element.
outerWidth([margin])	Get the outer width (includes the border and padding by default) for the first matched element.
position( )	Gets the top and left position of an element relative to its offset parent.
scrollLeft(val)	When a value is passed in, the scroll left offset is set to that value on all matched elements.
scrollLeft( )	Gets the scroll left offset of the first matched element.
scrollTop(val)	When a value is passed in, the scroll top offset is set to that value on all matched elements.
scrollTop( )	Gets the scroll top offset of the first matched element.
width(val)	Set the CSS width of every matched element.

width( )	Get the current computed, pixel, width of the first matched element.

jQuery - DOM Manipulation Methods

JQuery provides methods to manipulate DOM in efficient way. You do not need to write big code to modify the value of any element's attribute or to extract HTML code from a paragraph or division.

JQuery provides methods such as .attr(), .html(), and .val() which act as getters, retrieving information from DOM elements for later use.

Content Manipulation:

The **html( )** method gets the html contents (innerHTML) of the first matched element.

Here is the syntax for the method:

***selector*.html( )**

Example:selector.html()

Following is an example which makes use of .html() and .text(val) methods. Here .html() retrieves HTML content from the object and then .text(val) method sets value of the object using passed parameter:

```
<html>
<head>
<title>the title</title>
   <script type="text/javascript"   src="jquery.js"></script>
   <script type="text/javascript" language="javascript">

   $(document).ready(function() {

     $("div").click(function () {
      var content = $(this).html();
      $("#result").text( content );
    });

   });

   </script>
   <style>
      #division{
               margin:10px;
               padding:12px;
        border:2px solid #666;
        width:60px;
               background-color:blue;
```

```
       }
    </style>
  </head>
  <body>
     <p>Click on the square below:</p>

     <span id="result"> </span>

     <div id="division" >
       This is Blue Square!!
     </div>
  </body>
  </html>
```

DOM Element Replacement:

You can replace a complete DOM element with the specified HTML or DOM elements. The **replaceWith(content)** method serves this purpose very well.

Here is the syntax for the method:

```
selector.replaceWith( content )
```

Here content is what you want to have instead of original element. This could be HTML or simple text.

Example: *selector*.replaceWith(content)

Following is an example which would replace division element with "<h1>JQuery is Great</h1>":

```
<html>
<head>
<title>the title</title>
  <script type="text/javascript"   src="jquery.js"></script>
  <script type="text/javascript" language="javascript">

  $(document).ready(function() {

   $("div").click(function () {
    $(this).replaceWith("<h1>JQuery is Great</h1>");
   });

  });

  </script>
  <style>
    #division{
                  margin:10px;
                  padding:12px;
```

```
                    border:2px solid #666;
                    width:60px;
                    background-color:blue;
    }
  </style>
</head>
<body>
  <p>Click on the square below:</p>
  <div id="division" >
    This is Blue Square!!
  </div>
</body>
</html>
```

Removing DOM Elements:

There may be a situation when you would like to remove one or more DOM elements from the document. JQuery provides two methods to handle the situation.

The **empty( )** method remove all child nodes from the set of matched elements where as the method **remove(expr)** method removes all matched elements from the DOM.

Here is the syntax for the method:

```
selector.remove( [ expr ])

or

selector.empty( )
```

You can pass optional paramter *expr* to filter the set of elements to be removed.

Example: *selector*.remove()

Following is an example where elements are being removed as soon as they are clicked:

```
<html>
<head>
<title>the title</title>
  <script type="text/javascript"   src="jquery.js"></script>
  <script type="text/javascript" language="javascript">

  $(document).ready(function() {

    $("div").click(function () {
      $(this).remove( );
  });

 });

 </script>
 <style>
   .myclass{
        margin:10px;
        padding:12px;
        border:2px solid #666;
        width:60px;
     }
 </style>
</head>
<body>
 <p>Click on any square below:</p>
 <span id="result"> </span>
 <div class="myclass" style="background-color:blue;"></div>
 <div class="myclass" style="background-color:green;"></div>
 <div class="myclass" style="background-color:red;"></div>
</body>
</html>
```

Inserting DOM elements:

There may be a situation when you would like to insert new one or more DOM elements in your existing document. JQuery provides various methods to insert elements at various locations.

The **after(content)** method insert content after each of the matched elements where as the method **before(content)** method inserts content before each of the matched elements.

Here is the syntax for the method:

```
selector.after( content )

or

selector.before( content )
```

Here content is what you want to insert. This could be HTML or simple text.

Example: *selector*.before(content)

Following is an example where <div> elements are being inserted just before the clicked element:

```
<html>
<head>
<title>the title</title>
   <script type="text/javascript"
   src="jquery.js"></script>
   <script type="text/javascript" language="javascript">

   $(document).ready(function() {

     $("div").click(function () {
       $(this).before('<div class="div"></div>' );
    });

   });

   </script>
   <style>
      .myclass{ margin:10px;padding:12px;
             border:2px solid #666;
             width:60px;
            }
  </style>
</head>
<body>
   <p>Click on any square below:</p>
   <span id="result"> </span>
   <div class="myclass" style="background-color:blue;"></div>
   <div class="myclass" style="background-color:green;"></div>
   <div class="myclass" style="background-color:red;"></div>
</body>
</html>
```

Example: *selector*.append(content)

```
<html>
<head>
<title>the title</title>
  <script type="text/javascript"
  src="jquery.js"></script>
  <script type="text/javascript" language="javascript">

  $(document).ready(function() {
   $("div").click(function () {
    $(this).append('<p> this is next paragraph</p>');
   });
  });

  </script>
  <style type="text/css" >
        .myclass{
                height:150px;
                width:200px;
                border:2px solid red;
                overflow:auto;
    <style>
    .sample{
          margin:10px;
          padding:12px;
          border:2px solid #666;
          width:150px;
        }
  </style>

</head>
<body>

   <p> this is first paragraph </p>

</body>
</html>
```

Example: ***selector*.clone(boolean)**

```
<html>
<head>
<title>the title</title>
  <script type="text/javascript"
  src="jquery.js"></script>
  <script type="text/javascript" language="javascript">

  $(document).ready(function() {
   $("div").click(function () {
                    var divElement = $(this).clone(true);
                    $(this).after(divElement);
   });
  });

  </script>
    <style>
    .sample{
                          margin:10px;
                          padding:12px;
         border:2px solid #666;
         width:150px;
                          background-color:pink;
        }
  </style>

</head>
<body>

   <div class="sample">
                    this is DIV
        </div>

</body>
</html>
```

DOM Manipulation Methods:

Following table lists down all the methods which you can use to manipulate DOM elements:

Method	Description
after(content)	Insert content after each of the matched elements.
append(content)	Append content to the inside of every matched element.
appendTo(selector)	Append all of the matched elements to another, specified, set of elements.
before(content)	Insert content before each of the matched elements.
clone(bool)	Clone matched DOM Elements, and all their event handlers, and select the clones.
clone( )	Clone matched DOM Elements and select the clones.
empty( )	Remove all child nodes from the set of matched elements.
html(val)	Set the html contents of every matched element.
html( )	Get the html contents (innerHTML) of the first matched element.
insertAfter(selector)	Insert all of the matched elements after another, specified, set of elements.
insertBefore(selector)	Insert all of the matched elements before another, specified, set of elements.

prepend(content)	Prepend content to the inside of every matched element.
prependTo(selector)	Prepend all of the matched elements to another, specified, set of elements.
remove(expr)	Removes all matched elements from the DOM.
replaceAll(selector)	Replaces the elements matched by the specified selector with the matched elements.
replaceWith(content)	Replaces all matched elements with the specified HTML or DOM elements.
text(val)	Set the text contents of all matched elements.
text( )	Get the combined text contents of all matched elements.
wrap(elem)	Wrap each matched element with the specified element.
wrap(html)	Wrap each matched element with the specified HTML content.
wrapAll(elem)	Wrap all the elements in the matched set into a single wrapper element.
wrapAll(html)	Wrap all the elements in the matched set into a single wrapper element.
wrapInner(elem)	Wrap the inner child contents of each matched element (including text nodes) with a DOM element.
wrapInner(html)	Wrap the inner child contents of each matched element (including text nodes) with an HTML structure.

jQuery - Events Handling

We have the ability to create dynamic web pages by using events. Events are actions that can be detected by your Web Application.

Following are the examples events:

- A mouse click
- A web page loading
- Taking mouse over an element
- Submitting an HTML form
- A keystroke on your keyboard
- etc.

When these events are triggered you can then use a custom function to do pretty much whatever you want with the event. These custom functions call Event Handlers.

Binding event handlers:

Using the jQuery Event Model, we can establish event handlers on DOM elements with the **bind()** method as follows:

```
$('div').bind('click', function( event ){
   alert('Hi there!');
});
```

This code will cause the division element to respond to the click event; when a user clicks inside this division thereafter, the alert will be shown.

The full syntax of the bind() command is as follows:

```
selector.bind( eventType[, eventData], handler)
```

Following is the description of the parameters:

- **eventType:** A string containing a JavaScript event type, such as **click** or **submit**. Refer to the next section for a complete list of event types.
- **eventData:** This is optional parameter is a map of data that will be passed to the event handler.
- **handler:** A function to execute each time the event is triggered.

Removing event handlers:

Typically, once an event handler is established, it remains in effect for the remainder of the life of the page. There may be a need when you would like to remove event handler.

jQuery provides the **unbind()** command to remove an exiting event handler. The syntax of unbind() is as follows:

```
selector.unbind(eventType, handler)

or
```

```
selector.unbind(eventType)
```

Following is the description of the parameters:

- **eventType:** A string containing a JavaScript event type, such as **click** or **submit**. Refer to the next section for a complete list of event types.
- **handler:** If provided, identifies the specific listener that is to be removed.

Event Types:

The following are cross platform and recommended event types which you can bind using JQuery:

Event Type	Description
blur	Occurs when the element loses focus
change	Occurs when the element changes
click	Occurs when a mouse click
dblclick	Occurs when a mouse double-click
error	Occurs when there is an error in loading or unloading etc.
focus	Occurs when the element gets focus
keydown	Occurs when key is pressed
keypress	Occurs when key is pressed and released
keyup	Occurs when key is released
load	Occurs when document is loaded
mousedown	Occurs when mouse button is pressed
mouseenter	Occurs when mouse enters in an element region

mouseleave	Occurs when mouse leaves an element region
mousemove	Occurs when mouse pointer moves
mouseout	Occurs when mouse pointer moves out of an element
mouseover	Occurs when mouse pointer moves over an element
mouseup	Occurs when mouse button is released
resize	Occurs when window is resized
scroll	Occurs when window is scrolled
select	Occurs when a text is selected
submit	Occurs when form is submitted
unload	Occurs when documents is unloaded

The Event Object:

The callback function takes a single parameter; when the handler is called the JavaScript event object will be passed through it.

The event object is often unneccessary and the parameter is omitted, as sufficient context is usually available when the handler is bound to know exactly what needs to be done when the handler is triggered, however there are certail attributes which you would need to be accessed.

The Event Attributes:

The following event properties/attributes are available and safe to access in a platform independent manner:

Property	Description
altKey	Set to true if the Alt key was pressed when the event was triggered, false if not. The Alt key is labeled Option on most Mac keyboards.

ctrlKey	Set to true if the Ctrl key was pressed when the event was triggered, false if not.
data	The value, if any, passed as the second parameter to the bind() command when the handler was established.
keyCode	For keyup and keydown events, this returns the key that was pressed.
metaKey	Set to true if the Meta key was pressed when the event was triggered, false if not. The Meta key is the Ctrl key on PCs and the Command key on Macs.
pageX	For mouse events, specifies the horizontal coordinate of the event relative from the page origin.
pageY	For mouse events, specifies the vertical coordinate of the event relative from the page origin.
relatedTarget	For some mouse events, identifies the element that the cursor left or entered when the event was triggered.
screenX	For mouse events, specifies the horizontal coordinate of the event relative from the screen origin.
screenY	For mouse events, specifies the vertical coordinate of the event relative from the screen origin.
shiftKey	Set to true if the Shift key was pressed when the event was triggered, false if not.
target	Identifies the element for which the event was triggered.
timeStamp	The timestamp (in milliseconds) when the event was created.
Type	For all events, specifies the type of event that was triggered (for example, click).
Which	For keyboard events, specifies the numeric code for the key that caused the event, and for mouse events, specifies which button was pressed (1 for left, 2 for middle, 3 for right)

Event Helper Methods:

jQuery also provides a set of event helper functions which can be used either to trigger an event to bind any event types mentioned above.

Trigger Methods:

Following is an example which would triggers the blur event on all paragraphs:

```
$("p").blur();
```

Binding Methods:

Following is an example which would bind a **click** event on all the <div>:

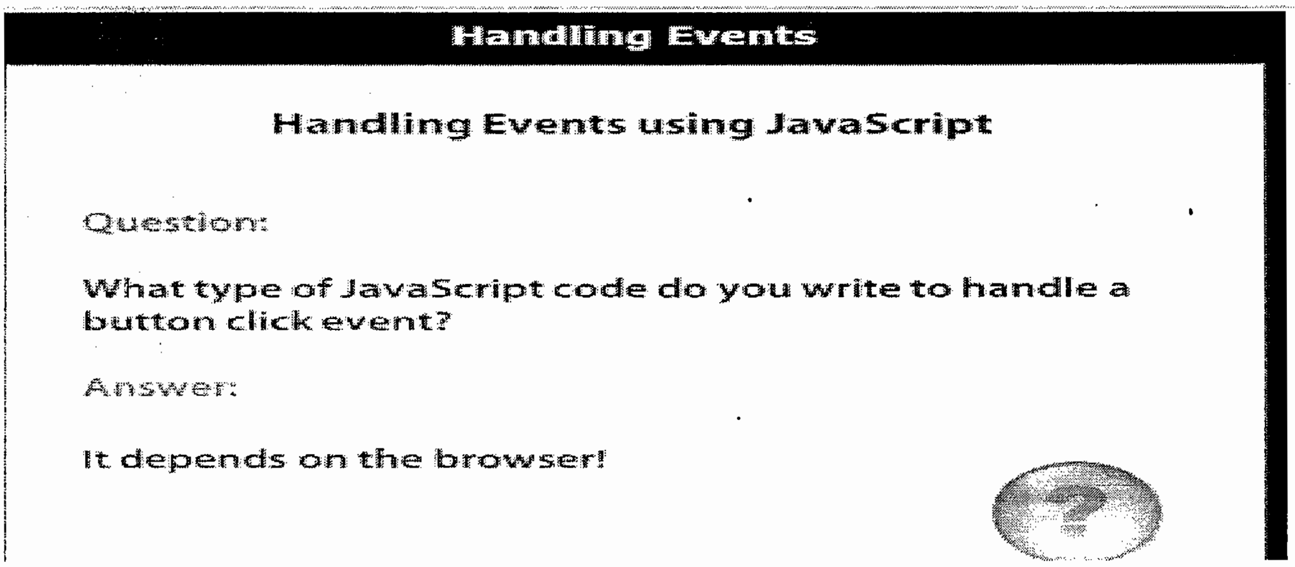
```
$("div").click( function () {
   // do something here
});
```

Here is a complete list of all the support methods provided by jQuery:

Method	Description
blur( )	Triggers the blur event of each matched element.
blur(fn)	Bind a function to the blur event of each matched element.
change( )	Triggers the change event of each matched element.
change(fn)	Binds a function to the change event of each matched element.
click( )	Triggers the click event of each matched element.
click(fn)	Binds a function to the click event of each matched element.
dblclick( )	Triggers the dblclick event of each matched element.
dblclick(fn)	Binds a function to the dblclick event of each matched element.

error( )	Triggers the error event of each matched element.
error(fn)	Binds a function to the error event of each matched element.
focus( )	Triggers the focus event of each matched element.
focus(fn)	Binds a function to the focus event of each matched element.
keydown( )	Triggers the keydown event of each matched element.
keydown(fn)	Bind a function to the keydown event of each matched element.
keypress( )	Triggers the keypress event of each matched element.
keypress(fn)	Binds a function to the keypress event of each matched element.
keyup( )	Triggers the keyup event of each matched element.
keyup(fn)	Bind a function to the keyup event of each matched element.
load(fn)	Binds a function to the load event of each matched element.
mousedown(fn)	Binds a function to the mousedown event of each matched element.
mouseenter(fn)	Bind a function to the mouseenter event of each matched element.
mouseleave(fn)	Bind a function to the mouseleave event of each matched element.
mousemove(fn)	Bind a function to the mousemove event of each matched element.
mouseout(fn)	Bind a function to the mouseout event of each matched element.
mouseover(fn)	Bind a function to the mouseover event of each matched element.

mouseup(fn)	Bind a function to the mouseup event of each matched element.
resize(fn)	Bind a function to the resize event of each matched element.
scroll(fn)	Bind a function to the scroll event of each matched element.
select( )	Trigger the select event of each matched element.
select(fn)	Bind a function to the select event of each matched element.
submit( )	Trigger the submit event of each matched element.
submit(fn)	Bind a function to the submit event of each matched element.
unload(fn)	Binds a function to the unload event of each matched element.



Handling Events

Handling Events using JavaScript

Question:

What type of JavaScript code do you write to handle a button click event?

Answer:

It depends on the browser!

Handling Events

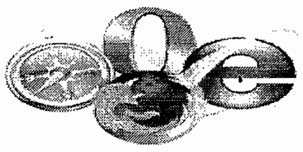
Event Attachment Techniques

Most Browsers:

```
myButton.addEventListener('click', function() { },false);
```

Internet Explorer:

```
myButton.attachEvent('onclick', function() { });
```



Handling Events

jQuery Event Model Benefits

- Events notify a program that a user performed some type of action
- jQuery provides a cross-browser event model that works in IE, Chrome, Opera, FireFox, Safari and more
- jQuery event model is simple to use and provides a compact syntax

Example : Event Handling complexity in Javascript

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

<button  id="buttonId"  >Click</button>

<script type="text/javascript">
        var myButton = document.getElementById("buttonId");
                if(document.addEventListener){
                        myButton.addEventListener("click",function(){alert("Click Button");}, false);
                }else{
                        myButton.attachEvent("onclick",function(){alert("Click IE Button");});
                }
</script>

</body>
</html>
```

Example : blur(fn) , focus(fn)

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

$(document).ready( function(){

 $("input[type='text']").blur(function(){
        $(this).toggleClass("ybg");
 });

 $("input[type='text']").focus(function(){
        $(this).toggleClass("ybg");
 });
});

</script>
<style type="text/css" >
 .ybg{
 background-color:yellow;
 }

</style>
</head>
<body>

 First Name : <input type="text" name="fname" /> <br/>
 Last Name : <input type="text" name="fname" /> <br/>

</body>
</html>
```

Example : click() , click(fn)

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

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

 $('#buttonId1').click(function(){
        // raising the event using jquery function
        alert("button1 is clicked");
        $("#buttonId2").click();
 });

 $("#buttonId2").click(function(){
        alert("button2 is clicked");
 });

});

</script>
</head>
<body>

<button  id="buttonId1"  >Click1</button>
<button  id="buttonId2"  >Click2</button>

</body>
</html>
```

Example : change(fn)

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

        $("#cityId").change(function(){
                $("#result").text($(this).val());
        });

});

</script>
</head>
<body>

<select id="cityId" >
        <option value="hyd">Hyderabad</option>
        <option value="bang">Banglore</option>
        <option value="chennai">Chennai</option>
        <option value="mumbai">Mumbai</option>
</select>
<span id="result" style="color:red"> <span>
```

```
</body>
</html>
```

Example : change(fn)

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

        $("#cityId1").change(function(){
                $("#cityId2").val($(this).val());
        });

});

</script>
</head>
<body>

<select id="cityId1" >
        <option value="hyd">Hyderabad</option>
        <option value="bang">Banglore</option>
        <option value="chennai">Chennai</option>
        <option value="mumbai">Mumbai</option>
</select>

<select id="cityId2" >
        <option value="hyd">Hyderabad</option>
        <option value="bang">Banglore</option>
        <option value="chennai">Chennai</option>
        <option value="mumbai">Mumbai</option>
</select>

</body>
</html>
```

Example : $(window).unload(fn)

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

 $(window).unload(function(){
        alert("Do you want to leave this page");
 });

});

</script>
</head>
<body>

<h1> Refresh the page, to call unload functionality</h1>

</body>
</html>
```

Example : error(fn)

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

$("img").error(function(){
   $(this).attr("src", "images/noimage.jpg");

  });
});

</script>
</head>
<body>

<img src="imags/home.jpg" > </img>

</body>
</html>
```

xample: click(function(event){ })

```
html>
head>
style type="text/css">
        .Hilight{
                  background-color:pink;
        }
 style>
 cript type="text/javascript" src="jquery.js"></script>
 cript type="text/javascript">
 locument).ready( function(){
        $("div#myDiv").click(function(event){
                $(this).addClass("Hilight");

                var result="";
                result= result+" X:"+event.pageX+"<br/>";
                result= result+" Y:"+event.pageY+"<br/>";
                result= result+" keyCode:"+event.keyCode+"<br/>";
                result= result+" TimeStamp:"+event.timeStamp+"<br/>";
                result= result+" Target element Id :"+$(event.target).attr("id")+"<br/>";
                result= result+" Event type :"+event.type+"<br/>";
                $(this).html(result);
        });

});

</script>
</head>
<body>

<div id="myDiv" style="width:400px;height:200px;border:2px solid black;" >
This is my div
This is my div
This is my div
This is my div
</div>

</body>
</html>
```

Example: mouseenter(fn), mouseleave(fn)

```
<html>
<head>
<style type="text/css">
        .Hilight{
                background-color:pink;
        }
</style>
<script type="text/javascript" src="jquery.js"></script>
<script type="text/javascript">
$(document).ready( function(){

        $("div#myDiv").mouseenter(function(){
                $(this).toggleClass("Hilight");
        }).mouseleave(function(){
                $(this).toggleClass("Hilight");
        }).click(function(event){
                $(this).text("X : "+event.pageX + " Y:"+event.pageY);
        });

});

</script>
</head>
<body>

<div id="myDiv" style="width:100px;height:100px;border:2px solid black;" >
        This is my div
        This is my div
        This is my div
        This is my div
</div>

</body>
</html>
```

Handling Events

Using bind()

- .bind(eventType, handler(eventObject)) attaches a handler to an event for the selected element(s)

```
$('#MyDiv').bind('click', function() {
  //Handle click event
});
```

.click() is the same as .bind('click')

Handling Events

Using unbind()

- .unbind(event) is used to remove a handler previously bound to an element:

```
$('#test').click(handler); can be unbound using
$("#test").unbind();
```

- Specific events can also be targeted using unbind():

```
$('#test').unbind('click');
```

Handling Events

Binding Multiple Events

- bind() allows multiple events to be bound to one or more elements
- Event names to bind are separated with a space:

```
$('#MyDiv').bind('mouseenter mouseleave',
   function() {
      $(this).toggleClass('entered');
   }
);
```

Example: bind(), unbind()

```
<html>
<head>
<style type="text/css">
        .Hilight{
                background-color:pink;
        }
</style>
<script type="text/javascript" src="jquery.js"></script>
<script type="text/javascript">
$(document).ready( function(){

        $("div#myDiv").bind("mouseenter mouseleave click",function(event){
                $(this).html(event.type);
        });

        $("button").click(function() {
                $("div#myDiv").unbind("click");
        });

});

</script>
</head>
<body>
<button> Retire Click event </button>
<div id="myDiv" style="width:400px;height:200px;border:2px solid black;" >
This is my div
This is my div
This is my div
This is my div
</div>

</body>
</html>
```

Example: bind(), unbind()

```
<html>
<head>
<style type="text/css">
        .Hilight{
                background-color:pink;
        }
</style>
<script type="text/javascript" src="jquery.js"></script>
<script type="text/javascript">
$(document).ready( function(){

        $("div#myDiv").bind("mouseenter mouseleave click", function(event){

                if(event.type=='click'){
                        $(this).text("X : "+event.pageX + " Y:"+event.pageY);
                }

                if(event.type=='mouseenter' || event.type=='mouseleave'){
                        $(this).toggleClass("Hilight");
                }

        });

        $("button").click(function(){
                //$("div#myDiv").unbind();
                $("div#myDiv").unbind("click");
        });

});

</script>
</head>
<body>

<div id="myDiv" style="width:100px;height:100px;border:2px solid black;" >
This is my div
This is my div
This is my div
This is my div
</div>
<button >Unbind event</button>
</body>
</html>
```

Key events occur in the following order:

1. KeyDown
2. KeyPress
3. KeyUp

NOTE: **In order to understand the difference between keydown and keypress, it is useful to understand the difference between a "character" and a "key". A "key" is a physical button on the computer's keyboard while a "character" is a symbol typed by pressing a button. In theory, the keydown and keyup events represent keys being pressed or released, while the keypress event represents a character being typed. The implementation of the theory is not same in all browsers.**

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

        var down=0;
        var up=0;
        var press=0;


        $("input#phoneId").bind("keydown keypress keyup", function(event){
                if(event.type=="keydown"){
                        $("#downId").text("Keydown : "+(++down));
                }

                if(event.type=="keyup"){
                        $("#upId").text("keyup : "+(++up));
                }

                if(event.type=="keypress"){
                        $("#pressId").text("keypress : "+(++press));
                }
        });


});

</script>
</head>
<body>
 Phone : <input type="text" id="phoneId" name="phone" />
 <br/>
 <span id="downId" style="color:red" > </span> <br/>
 <span id="pressId" style="color:red" > </span> <br/>
 <span id="upId" style="color:red" > </span>
</body>
</html>
```

Example: mousedown(fn), mouseup(fn)

```
<html>
<head>
 <script src="http://code.jquery.com/jquery-latest.js"></script>
</head>
<body>
 <p>Press mouse and release here.</p>

<script>
   $("p").mouseup(function(){
    $(this).append('<span style="color:blue;">Mouse up.</span>');
   }).mousedown(function(){
    $(this).append('<span style="color:red;">Mouse down.</span>');
   });

</script>

</body>
</html>
```

Example: mouseenter(fn), mouseleave(fn)

```
<html>
<head>
<script type="text/javascript" src="jquery.js"></script>
<script type="text/javascript">
$(document).ready(function(){
 $("div").mouseenter(function(){
  $(this).css("background-color","yellow");
 });
 $("div").mouseleave(function(){
  $(this).css("background-color","pink");
 });
});
</script>
<style>
.myclass{
 height:50px;
 width:200px;
 border:2px solid red;
}
</style>
</head>
<body>
<div class="myclass">
Hover the mouse pointer over this paragraph.
</div>
</body>
</html>
```

Example: mouseover(fn), mouseout(fn)

```
<html>
<head>
<script type="text/javascript" src="jquery.js"></script>
<script type="text/javascript">
$(document).ready(function(){
 $("div").mouseover(function(){
  $("div").css("background-color","yellow");
 });
 $("div").mouseout(function(){
  $("div").css("background-color","pink");
 });
});
</script>
<style>
.myclass{
 height:50px;
 width:200px;
 border:2px solid red;
}
</style>
</head>
<body>
<div class="myclass">
Hover the mouse pointer over this paragraph.
</div>
</body>
```

```
</html>
```

Example:Difference among mouseover(fn), mouseout(fn), mouseenter(fn),mouseleave(fn)

```
<html>
<head>
<script type="text/javascript" src="jquery.js"></script>
<script type="text/javascript">
var overCount=0;
var outCount=0;
var enterCount=0;
var leaveCount=0;
$(document).ready(function(){

 $("div#oneId").mouseover(function(){
          $("span#overResult").text(++overCount);

 });

 $("div#oneId").mouseout(function(){
          $("span#outResult").text(++outCount);

 });

 $("div#twoId").mouseenter(function(){
          $("span#enterResult").text(++enterCount);
 });

 $("div#twoId").mouseleave(function(){
          $("span#leaveResult").text(++leaveCount);
 });

  });
</script>
<style>
div.out {
          width:45%;
          height:120px;
          margin:0 15px;
          background-color:#D6EDFC;
          float:left;
}
div.in {
          width:60%;
          height:60%;
          background-color:#FFCC00;
          margin:10px auto;
          font-size:25px;
}
```

```
</style>
</head>
<body>
        <h1>The mouseover() event triggers if a mouse pointer enters the child elements as well as the selected
element.</h1>
        <h1>The mouseenter() event is only triggered when the the mouse pointer enters the selected element. </h1>

        <div class="out" id="oneId" >
                <div class="in" >
                        Mouseover: <span id="overResult" ></span> <br/>
                        Mouseout: <span id="outResult" ></span>
                </div>
        </div>

        <div class="out" id="twoId" >
                <div class="in" >
                        Mouseenter : <span id="enterResult" ></span> <br/>
                        Mouseleave : <span id="leaveResult" ></span>
                </div>
        </div>

</body>
</html>
```

Example: mousemove(fn)

```
<html>
<head>
<script type="text/javascript" src="jquery.js"></script>
<script type="text/javascript">
$(document).ready(function(){
  $(document).mousemove(function(e){
    $("span").text(e.pageX + ", " + e.pageY);
  });
});
</script>
</head>
<body>
<h1>Mouse is at coordinates: <span></span>.</h1>
</body>
</html>
```

Example: Selector of some other selector

```
<html>
<head>
<script type="text/javascript" src="jquery.js"></script>
<script type="text/javascript">
$(document).ready(function(){
 $("button").click(function(){
	//var oneSelector = $("div#oneId");
	//$("div.myclass", oneSelector).css("background-color", "pink");

	var twoSelector = $("div#twoId");
	$("div.myclass", twoSelector).css("background-color", "pink");
 });

});
</script>
<style type="text/css" >
.myclass{
	width: 200px;
	height: 100px;
	border: 2px solid green;
	margin : 20px;
}
</style>
</head>
<body>

<div id="oneId" >
 <div class="myclass" >
	DIV ONE
 </div>
</div>

<div id="twoId" >
 <div class="myclass" >
	DIV TWO
 </div>
</div>

<button> Click </button>

</body>
</html>
```

Handling Events

live() and delegate() Functions

- live() and delegate() allow new elements added into the DOM to automatically be "attached" to an event handler

live() –Allows binding of event handlers to elements that match a selector, including future elements. Events bubble up to the document object.

delegate() – Replacement for live() in jQuery 1.4. Attaches an event handler directly to the selector context.

Handling Events

Using live()

- Event handlers can be set using live()
- The document object handles events by default
- Works even when new objects are added into the DOM:

```
$('.someClass').live('click',
    someFunction);
```

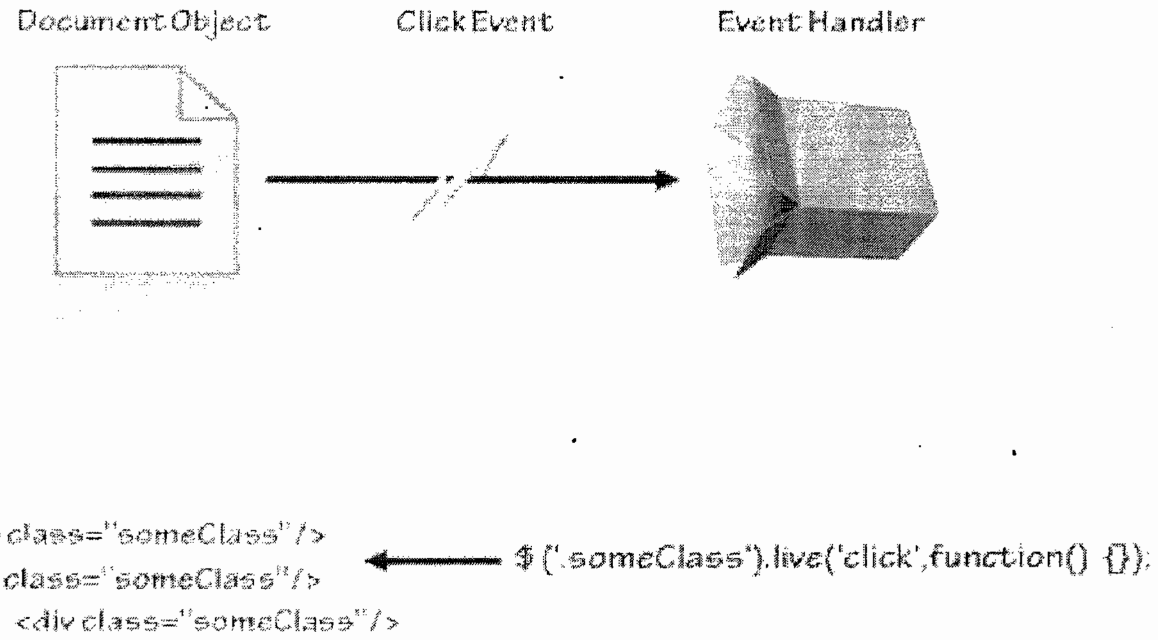
Any element added with .someClass will have a click event

- Stop live event handling using die():

```
$('.someClass').die('click', someFunction);
```

Handling Events

How live() Works



Handling Events

Using delegate()

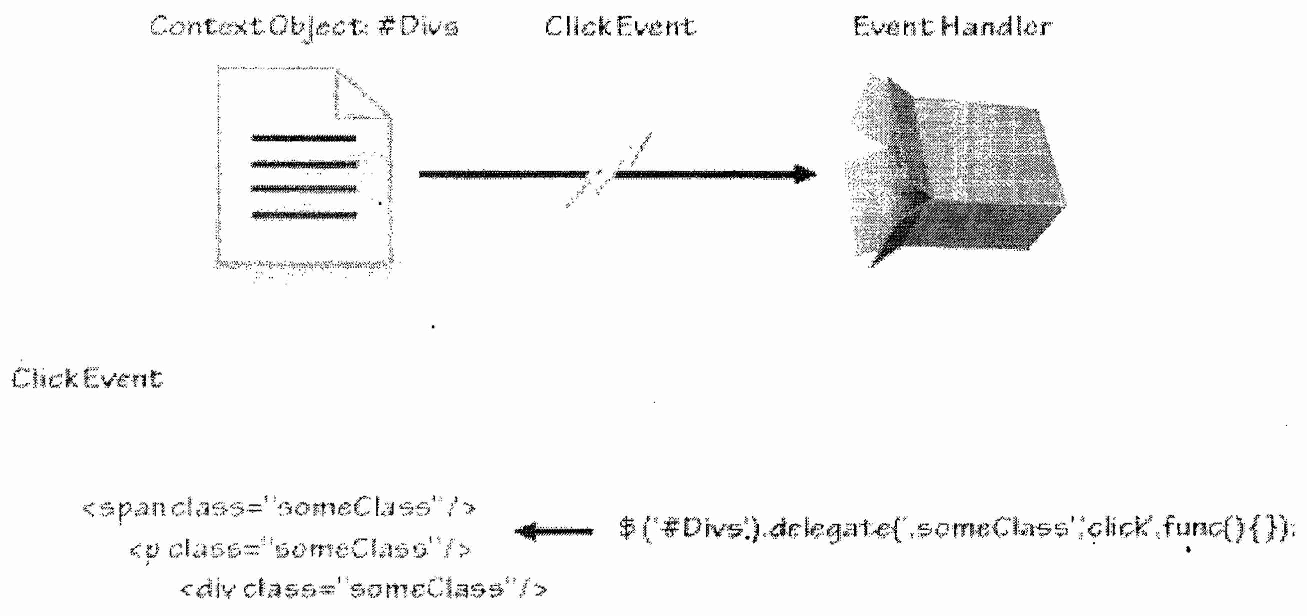
- Newer version of live() added in jQuery 1.4
- A context object (#Divs in the sample below) handles events by default rather than the document object
- Works even when new objects are added into the DOM:

```
$('#Divs').delegate('div','click',someFunction);
```

- Stop delegate event handling using undelegate()

Handling Events

How delegate() Works



Example: live(fn), die(fn), delegate(fn), undelegate(fn)

```
<html>
<head>
<script type="text/javascript" src="jquery.js"></script>
<script type="text/javascript">
$(document).ready( function(){
        var PhoneDiv = $("div#PhoneDiv");

        $("#AddPhoneButton").click(function(){
                var PhoneTypeDivClone= $("div:eq(0)",PhoneDiv).clone();
                var PhoneNumberDivClone= $("div:eq(1)",PhoneDiv).clone();
                $("select", PhoneTypeDivClone).val('');
                $("input", PhoneNumberDivClone).val('');
                PhoneDiv.append("<div style='clear:both' />").append(PhoneTypeDivClone)
                                                      .append(PhoneNumberDivClone);
        });

        $("input", PhoneDiv).live("keypress", function(event){
                if(event.keyCode<48 || event.keyCode>57){
                        //event.stopPropagation();
                        return false;
                }
```

```
                                        <div id="PhoneTypeDiv" style="float:left" >
                                                <select name="PhoneType" >
                                                        <option value="">Select One</option>
                                                        <option value="Mobile">Mobile</option>
                                                        <option value="Office">Office</option>
                                                        <option value="Home">Home</option>
                                                </select>
                                        </div>
                                        <div id="PhoneNumberDiv" style="float:left" >
                                                <input type="text" name="PhoneNumber"/>
                                        </div>
                                </div>
                        </td>
                </tr>
                <tr>
                        <td>City</td>
                        <td>
                                <select id="cityId" >
                                        <option value="hyd">Hyderabad</option>
                                        <option value="bang">Banglore</option>
                                        <option value="chennai">Chennai</option>
                                        <option value="mumbai">Mumbai</option>
                                </select>
                        </td>
                </tr>
                <tr>
                        <td colspan="2" align="center" >
                                <input type="submit" value="Submit"/>
                        </td>
                </tr>

</table>
 <button>Release Event Handlers</button>
</body>
</html>
```

Handling Events

Handling Hover Events

- Hover events can be handled using hover():

```
$(selector).hover(handlerIn, handlerOut)
```

- handlerIn is equivalent to mouseenter and handlerOut is equivalent to mouseleave

Handling Events

Using hover()

- This example highlights #target on mouseenter and sets it back to white on mouseleave

```
$('#target').hover(
    function(){
        $(this).css('background-color', '#00FF99');
    },
    function(){
        $(this).css('background-color', '#FFFFFF');
    }
);
```

Handling Events

Alternate Hover Example

- Another option is $(selector).hover(handlerInOut)
- Fires the same handler for mouseenter and mouseleave events
- Used with jQuery's toggle methods:

```
$('p').hover(function() {
    $(this).toggleClass('over');
});
```

This code will toggle the class applied to a paragraph element

Example: hover(fn)

```
<html>
<head>
<style type="text/css" >
        .Hilight{
                background-color:yellow;
        }

        table, td, th{
                border-collapse:collapse;
                border : 2px solid green;
                width : 200px;
        }

</style>
<script type="text/javascript" src="jquery.js"></script>
<script type="text/javascript">
$(document).ready( function(){
        /*
        $("table#MyTable tr").hover(
                function(){
                        // mouseenter
                        $(this).css("background-color","#efefef");
                },
                function(){
                        // mouseleave
                        $(this).css("background-color","#ffffff");
                }
        );
        */
```

```
		$("table#MyTable tr").hover(function(){
			$(this).toggleClass("Hilight");
		});
});
</script>
<style type="text/css" >
</style>
</head>
<body>

<table  id="MyTable" >
		<tr>
			<th>ENO</th>
			<th>Name</th>
			<th>Sal</th>
		</tr>
		<tr>
			<td>1001</td>
			<td>sekhar1</td>
			<td>459.09</td>
		</tr>
		<tr>
			<td>1002</td>
			<td>sekhar2</td>
			<td>459.09</td>
		</tr>
		<tr>
			<td>1003</td>
			<td>sekhar3</td>
			<td>459.09</td>
		</tr>
		<tr>
			<td>1004</td>
			<td>sekhar4</td>
			<td>459.09</td>
		</tr>
		<tr>
			<td>1005</td>
			<td>sekhar5</td>
			<td>459.09</td>
		</tr>

</table>

</body>
</html>
```

ıle: hover(fn, fn, fn, fn)

```
<html>
<head>
<style type="text/css" >
 .Hilight{
        background-color:#efefef;
 }
 table, td, th{
        border-collapse:collapse;
        border : 2px solid green;
        width : 200px;
 }

</style>
<script type="text/javascript" src="jquery.js"></script>
<script type="text/javascript">
$(document).ready( function(){

 $("table#MyTable tr").toggle(
        function(){
                // click
                $(this).css("background-color","#efefef");
        },
        function(){
                // click
                $(this).css("background-color","purple");
        },
        function(){
                // click
                $(this).css("background-color","yellow");
        },
        function(){
                // click
                $(this).css("background-color","pink");
        }
 );



});
</script>
</head>
<body>

<table id="MyTable" border="1" >
 <tr>
        <th>ENO</th>
        <th>Name</th>
        <th>Sal</th>
 </tr>
```

```
<tr>
        <td>1001</td>
        <td>sekhar1</td>
        <td>459.09</td>
</tr>
<tr>
        <td>1002</td>
        <td>sekhar2</td>
        <td>459.09</td>
</tr>
<tr>
        <td>1003</td>
        <td>sekhar3</td>
        <td>459.09</td>
</tr>

</table>

</body>
</html>
```

jQuery - Effects

jQuery provides a trivially simple interface for doing various kind of amazing effects. jQuery methods allow us to quickly apply commonly used effects with a minimum configuration.

JQuery Effect Methods:

You have seen basic concept of jQuery Effects. Following table lists down all the important methods to create different kind of effects:

jQuery Hide and Show

With jQuery, you can hide and show HTML elements with the hide() and show() methods:

Example: hide(), show()

```
<html>
<head>
<script type="text/javascript" src="jquery.js"></script>
<script type="text/javascript">
$(document).ready(function(){
 $("#hide").click(function(){
  $("h1").hide();
 });
 $("#show").click(function(){
  $("h1").show();
 });
});
</script>
</head>
<body>
<h1>If you click on the "Hide" button, I will disappear. <br/>
   If you click on the "Show" button, I will appear </h1>
<button id="hide">Hide</button>
<button id="show">Show</button>
</body>
</html>
```

Both hide() and show() can take.the two optional parameters: speed and callback.

Syntax:

$(selector).hide(speed,callback)

$(selector).show(speed,callback)

The speed parameter specifies the speed of the hiding/showing, and can take the following values: "slow", "fast", "normal", or milliseconds:

Example: hide(time)

```
<html>
<head>
<script type="text/javascript" src="jquery.js"></script>
<script type="text/javascript">
$(document).ready(function(){
 $("button").click(function(){
  $("p").hide(1000);
 });
});
</script>
</head>
<body>
<button>Hide</button>
<p>This is a paragraph with little content.</p>
<p>This is another small paragraph.</p>
</body>
</html>
```

Example: hide("slow")

```
<html>
<head>
<script type="text/javascript" src="jquery.js"></script>
<script type="text/javascript">
        $(document).ready(function(){
                $(".ex .hide").click(function(){
                        $(this).parents(".ex").hide("slow");
                });
        });
</script>
<style type="text/css">
        div.ex
        {
                background-color:#e5eecc;
                padding:7px;
                border:solid 1px #c3c3c3;
        }
</style>
</head>
<body>

        <h3>Island Trading</h3>
        <div class="ex">
                <button class="hide">Hide me</button>
                <p>Contact: Helen Bennett<br />
                  Garden House Crowther Way<br />
                  London</p>
        </div>

        <h3>Paris spécialités</h3>
        <div class="ex">
                <button class="hide">Hide me</button>
                <p>Contact: Marie Bertrand<br />
                  265, Boulevard Charonne<br />
                  Paris</p>
        </div>

</body>
</html>
```

The callback parameter is the name of a function to be executed after the hide (or show) function completes.

jQuery Toggle

The jQuery toggle() method toggles the visibility of HTML elements using the show() or hide() methods.

Shown elements are hidden and hidden elements are shown.

Syntax:

$(selector).toggle(speed,callback)

The speed parameter can take the following values: "slow", "fast", "normal", or milliseconds.

Example: toggle()

```
<html>
<head>
<script type="text/javascript" src="jquery.js"></script>
<script type="text/javascript">
$(document).ready(function(){
  $("button").click(function(){
    $("p").toggle();
  });
});
</script>
</head>
<body>

<button>Toggle</button>
<p>This is a paragraph with little content.</p>
<p>This is another small paragraph.</p>
</body>
</html>
```

The callback parameter is the name of a function to be executed after the hide (or show) method completes.

jQuery Slide - slideDown, slideUp, slideToggle

The jQuery slide methods gradually change the height for selected elements.

jQuery has the following slide methods:

$(selector).slideDown(speed,callback)

$(selector).slideUp(speed,callback)

$(selector).slideToggle(speed,callback)

The speed parameter can take the following values: "slow", "fast", "normal", or milliseconds.

The callback parameter is the name of a function to be executed after the function completes.

Example : slideDown(), slideUp(), slideToggle()

```
<html>
```

```
<head>

<script type="text/javascript" src="jquery.js"></script>
<script type="text/javascript">
$(document).ready(function(){

        $(".flip").click(function(){
                $(".panel").slideDown("slow");
        });
/*
        $(".flip").click(function(){
                $(".panel").slideUp("slow");
        });
*/
/*
        $(".flip").click(function(){
                $(".panel").slideToggle("slow");
        });
*/
});
</script>

<style type="text/css">
        div.panel,p.flip
        {
                margin:0px;
                padding:5px;
                text-align:center;
                background:#e5eecc;
                border:solid 1px #c3c3c3;
        }
        div.panel
        {
                height:120px;
                display:none;
        }
</style>
</head>

<body>

<div class="panel">
        <p>Because time is valuable, we deliver quick and easy learning.</p>
        <p>At Sekharit, you can study everything you need to learn, in an accessible and handy
format.</p>
</div>

<p class="flip">Show/Hide Panel</p>

</body>
```

```
</html>
```

jQuery Fade - fadeIn, fadeOut, fadeTo

The jQuery fade methods gradually change the opacity for selected elements.

jQuery has the following fade methods:

$(selector).fadeIn(speed,callback)

$(selector).fadeOut(speed,callback)

$(selector).fadeTo(speed,opacity,callback)

The speed parameter can take the following values: "slow", "fast", "normal", or milliseconds.

The opacity parameter in the fadeTo() method allows fading to a given opacity.

The callback parameter is the name of a function to be executed after the function completes.

Example : fadeIn(), fadeOut(), fadeTo()

```
<html>
<head>
<script type="text/javascript" src="jquery.js"></script>
<script type="text/javascript">
$(document).ready(function(){
 $("button#fadeOutId").click(function(){
  $("div#myId").fadeOut(4000);
 });

 $("button#fadeInId").click(function(){
  $("div#myId").fadeIn(4000);
 });

 $("button#fadeToId").click(function(){
  $("div#myId").fadeTo("slow",0.25);
 });

});
</script>
</head>

<body>
<div id="myId" style="background:green;width:300px;height:300px">ME AWAY!</div>
<button id="fadeOutId" > Fade Out </button>
<button id="fadeInId" > Fade In </button>
```

```
<button id="fadeToId" > Fade To </button>
</body>

</html>
```

jQuery Custom Animations

The syntax of jQuery's method for making custom animations is:

$(selector).animate({params},[duration],[easing],[callback])

The key parameter is **params**. It defines the CSS properties that will be animated. Many properties can be animated at the same time:

animate({width:"70%",opacity:0.4,marginLeft:"0.6in",fontSize:"3em"});

The second parameter is **duration**. It specifies the speed of the animation. Possible values are "fast", "slow", "normal", or milliseconds.

NOTE: Only numeric values can be animated (like "margin:30px"). String values cannot be animated (like "background-color:red").

NOTE: The styles are set with DOM names (like "fontSize"), not CSS names (like "font-size").

Example : animate()

```
<html>
<head>
<script type="text/javascript" src="jquery.js"></script>
<script type="text/javascript">
$(document).ready(function(){
 $("button").click(function(){
  $("div").animate({height:300},"slow");
  $("div").animate({width:300},"slow");
  $("div").animate({height:100},"slow");
  $("div").animate({width:100},"slow");
 });
});
</script>
</head>

<body>
<button>Start Animation</button>
<br /><br />
<div style="background:#98bf21;height:100px;width:100px;position:relative">
</div>
</body>
</html>
```

Example : animate()

```
<html>
<head>
<script type="text/javascript" src="jquery.js"></script>
<script type="text/javascript">
$(document).ready(function(){
 $("button").click(function(){
  $("div").animate({left:"100px"},"slow");
  $("div").animate({fontSize:"3em", borderWidth: "10px" },"slow", function() {
                $(this).css("background-color", "yellow");
        });

 });
});
</script>
 <style>
        div {
                background-color:#98bf21;
                width:200px;
                height:100px;
                border:1px solid green;
                position:relative;
        }
</style>
</head>
<body>

<button>Start Animation</button>
<br /><br />
<div>HELLO</div>
</body>
</html>
```

NOTE: HTML elements are positioned static by default and cannot be moved.
To make elements moveable, set the CSS position property to fixed, relative or absolute.

Methods and Description

animate(params, [duration, easing, callback])
A function for making custom animations.

fadeIn(speed, [callback])
Fade in all matched elements by adjusting their opacity and firing an optional callback after completion.

fadeOut(speed, [callback])

Fade out all matched elements by adjusting their opacity to 0, then setting display to "none" and firing an optional callback after completion.

fadeTo(speed, opacity, callback)

Fade the opacity of all matched elements to a specified opacity and firing an optional callback after completion.

hide( )

Hides each of the set of matched elements if they are shown.

hide(speed, [callback])

Hide all matched elements using a graceful animation and firing an optional callback after completion.

show( )

Displays each of the set of matched elements if they are hidden.

show(speed, [callback])

Show all matched elements using a graceful animation and firing an optional callback after completion.

slideDown(speed, [callback])

Reveal all matched elements by adjusting their height and firing an optional callback after completion.

slideToggle(speed, [callback])

Toggle the visibility of all matched elements by adjusting their height and firing an optional callback after completion.

slideUp(speed, [callback])

Hide all matched elements by adjusting their height and firing an optional callback after completion.

stop([clearQueue, gotoEnd])

Stops all the currently running animations on all the specified elements.

toggle( )

Toggle displaying each of the set of matched elements.

toggle(speed, [callback])

Toggle displaying each of the set of matched elements using a graceful animation and firing an optional callback after completion.

toggle(switch)
Toggle displaying each of the set of matched elements based upon the switch (true shows all elements, false hides all elements).

jQuery.fx.off
Globally disable all animations.

jQuery - AJAX

AJAX is an acronym standing for Asynchronous JavaScript and XML and this technology help us to load data from the server without a browser page refresh.

JQuery is a great tool which provides a rich set of AJAX methods to develop next generation web application.

Loading simple data:

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

Syntax:

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

```
[selector].load( URL, [data], [callback] );
```

Here is the description of all the parameters:

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

Example:

Consider the following HTML file with a small JQuery coding:

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

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

```
<h1>THIS IS RESULT...</h1>
```

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

Getting JSON data:

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

Syntax:

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

```
[selector].getJSON( URL, [data], [callback] );
```

Here is the description of all the parameters:

- **URL:** The URL of the server-side resource contacted via the GET method.
- **data:** An object whose properties serve as the name/value pairs used to construct a query string to be appended to the URL, or a preformatted and encoded query string.

- **callback:** A function invoked when the request completes. The data value resulting from digesting the response body as a JSON string is passed as the first parameter to this callback, and the status as the second.

Example:

Consider the following HTML file with a small JQuery coding:

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

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

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

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

Passing data to the Server:

Many times you collect input from the user and you pass that input to the server for further processing. JQuery AJAX made it easy enough to pass collected data to the server using **data** parameter of any available Ajax method.

Example:

This example demonstrate how can pass user input to a web server script which would send the same result back and we would print it:

```
<html>
<head>
<title>the title</title>
   <script type="text/javascript"
   src="jquery.js"></script>
   <script type="text/javascript" language="javascript">
   $(document).ready(function() {
      $("#driver").click(function(event){
          var name = $("#name").val();
          $("#stage").load('/jquery/result.php', {"name":name} );
      });
   });
   </script>
</head>
<body>
   <p>Enter your name and click on the button:</p>
   <input type="input" id="name" size="40" /><br />
   <div id="stage" style="background-color:blue;">
          STAGE
   </div>
   <input type="button" id="driver" value="Show Result" />
</body>
</html>
```

Here is the code written in **result.php** script:

```
<?php
if( $_REQUEST["name"] )
{
   $name = $_REQUEST['name'];
   echo "Welcome ". $name;
}
?>
```

Now you can enter any text in the given input box and then click "Show Result" button to see what you have entered in the input box.

JQuery AJAX Methods:

You have seen basic concept of AJAX using JQuery. Following table lists down all important JQuery AJAX methods which you can use based your programming need:

Methods and Description
jQuery.ajax(options) Load a remote page using an HTTP request.
jQuery.ajaxSetup(options) Setup global settings for AJAX requests.
jQuery.get(url, [data], [callback], [type]) Load a remote page using an HTTP GET request.
jQuery.getJSON(url, [data], [callback]) Load JSON data using an HTTP GET request.
jQuery.getScript(url, [callback]) Loads and executes a JavaScript file using an HTTP GET request.
jQuery.post(url, [data], [callback], [type]) Load a remote page using an HTTP POST request.
load(url, [data], [callback]) Load HTML from a remote file and inject it into the DOM.
serialize( ) Serializes a set of input elements into a string of data.
serializeArray( ) Serializes all forms and form elements like the .serialize() method but returns a JSON data structure for you to work with.

JQuery AJAX Events:

You can call various JQuery methods during the life cycle of AJAX call progress. Based on different events/stages following methods are available:

You can go through all the AJAX Events.

Methods and Description
ajaxComplete(callback) Attach a function to be executed whenever an AJAX request completes.
ajaxStart(callback) Attach a function to be executed whenever an AJAX request begins and there is none already active.
ajaxError(callback) Attach a function to be executed whenever an AJAX request fails.
ajaxSend(callback) Attach a function to be executed before an AJAX request is sent.
ajaxStop(callback) Attach a function to be executed whenever all AJAX requests have ended.
ajaxSuccess(callback) Attach a function to be executed whenever an AJAX request completes successfully.

Working with Ajax Features

Agenda

- jQuery AJAX Functions
- Loading HTML Content from the Server
- Making GET Requests
- Making POST Requests
- Introduction to the ajax() Function

Working with Ajax Features

jQuery Ajax Features

- **jQuery allows Ajax requests to be made from a page:**
  - Allows parts of a page to be updated
  - Cross-Browser Support
  - Simple API
  - GET and POST supported
  - Load JSON, XML, HTML or even scripts

NOTE: In the case of javascript, to implement ajax functionality first we need to identify the browser as follows. So javascript ajax functionality is browser dependent.

```
var xmlHttp = null;
if (window.XMLHttpRequest) {
  // If IE7, Mozilla, Safari, and so on: Use native object.
  xmlHttp = new XMLHttpRequest();
}
else
{
  if (window.ActiveXObject) {
     // ...otherwise, use the ActiveX control for IE5.x and IE6.
     xmlHttp = new ActiveXObject('MSXML2.XMLHTTP.3.0');
   }
}
```

jQuery Ajax Functions

- **jQuery provides several functions that can be used to send and receive data:**
  - $(selector).load(): Loads HTML data from the server
  - $.get() and $.post(): Get raw data from the server
  - $.getJSON(): Get/Post and return JSON
  - $.ajax(): Provides core functionality

- **jQuery Ajax functions work with REST APIs, Web Services and more**

Using load()

- $(selector).load(url,data,callback) allows HTML content to be loaded from a server and added into a DOM object:

```
$(document).ready(function(){
    $('#HelpButton').click(function(){
        $('#MyDiv').load('HelpDetails.html');
    });
});
```

Using load() With a Selector

- A selector can be added after the URL to filter the content that is returned from calling load():

```
$('#MyDiv').load('HelpDetails.html #MainTOC');
```

Passing Data using load()

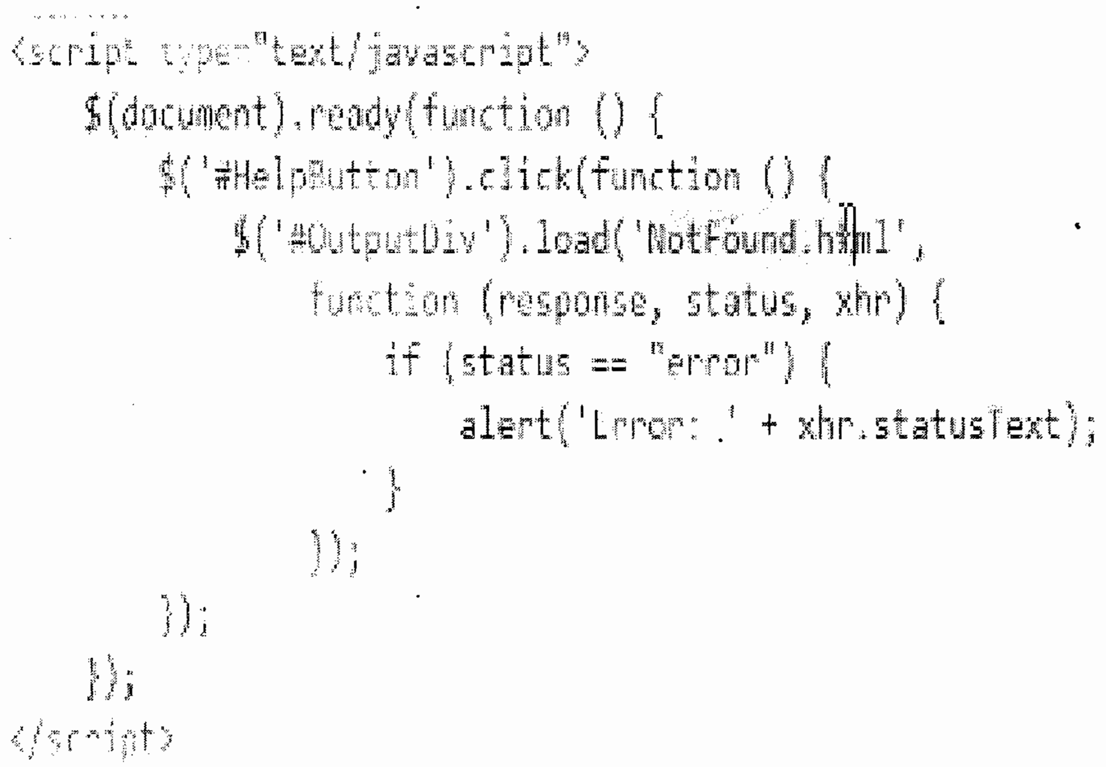
- Data can be passed to the server using load(url,data):

```
$('#MyDiv').load('GetCustomers.aspx',
 {PageSize:25});
```

Using a Callback Function with load()

- load() can be passed a callback function:

```
$('#OutputDiv').load('NotFound.html',
    function (response, status, xhr) {
        if (status == "error") {
            alert(xhr.statusText);
        }
});
```

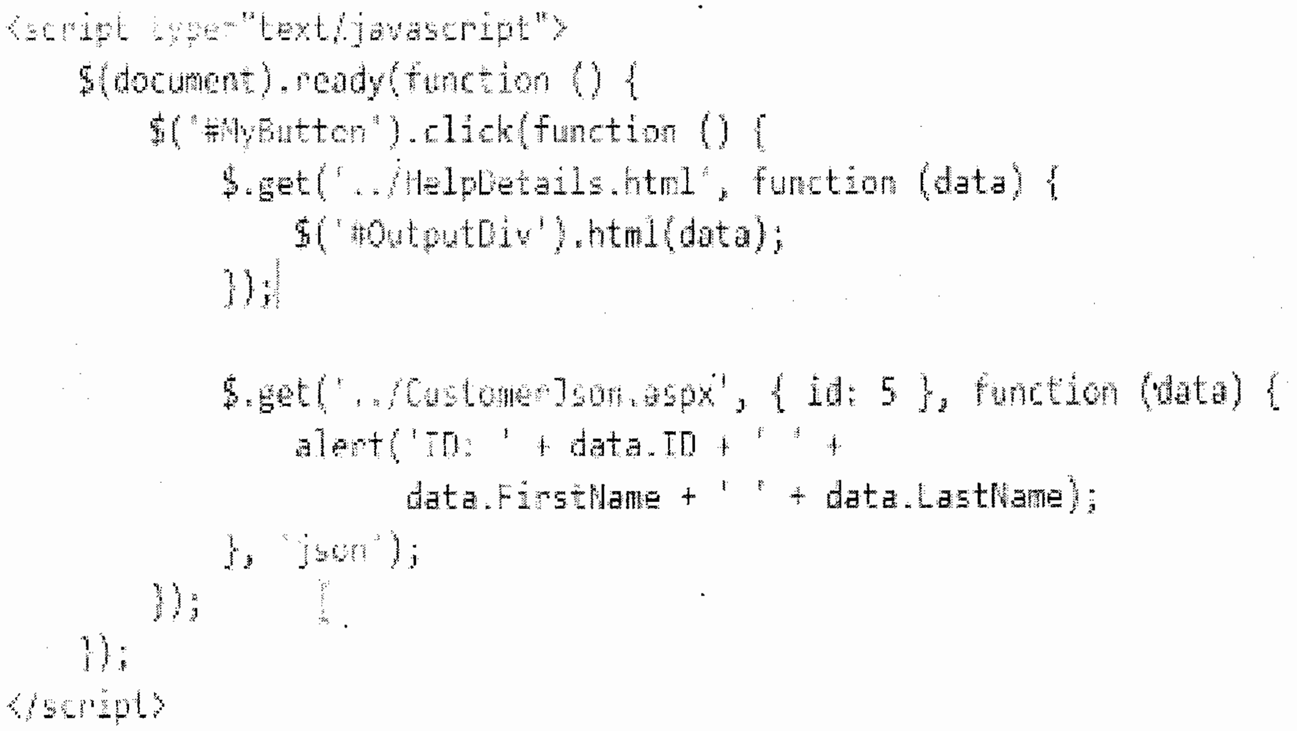


```
<script type="text/javascript">
    $(document).ready(function () {
        $('#HelpButton').click(function () {
            $('#OutputDiv').load('NotFound.html',
                function (response, status, xhr) {
                    if (status == "error") {
                        alert('Error: ' + xhr.statusText);
                    }
                });
        });
    });
</script>
```

Using get()

- $.get(url,data,callback,datatype) can retrieve data from a server:

```
$.get('HelpDetails.html', function (data) {
    $('#OutputDiv').html(data);
});
```



```
<script type="text/javascript">
    $(document).ready(function () {
        $('#MyButton').click(function () {
            $.get('../HelpDetails.html', function (data) {
                $('#OutputDiv').html(data);
            });

            $.get('../CustomerJson.aspx', { id: 5 }, function (data) {
                alert('ID: ' + data.ID + ' ' +
                      data.FirstName + ' ' + data.LastName);
            }, 'json');
        });
    });
</script>
```

```
<script type="text/javascript">
    $(document).ready(function () {
        $('#MyButton').click(function () {
            $.getJSON('../CustomerJson.aspx', { id: 5 },
              function (data) {
                  alert('ID: ' + data.ID + ' ' +
                    data.FirstName + ' ' + data.LastName);
              });
        });
    });
</script>
```

Using post()

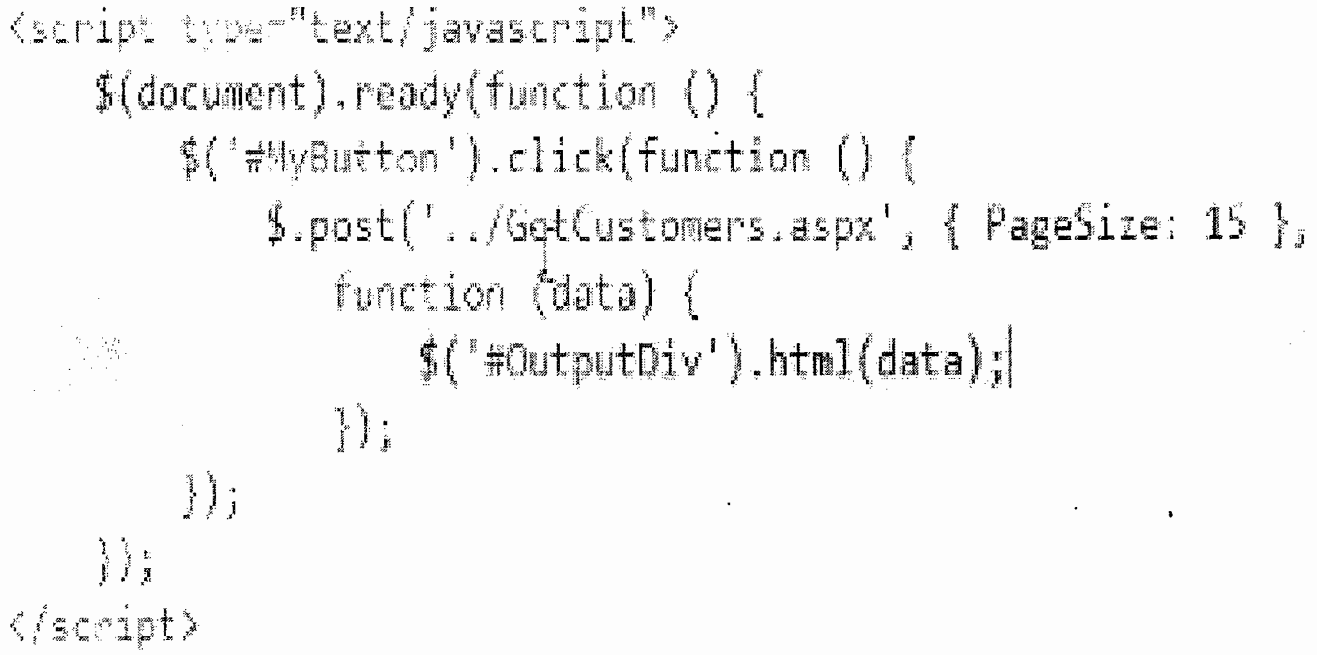
- $.post(url,data,callback,datatype) can post data to a server and retrieve results:

```
$.post('GetCustomers.aspx', {PageSize:15},
    function (data) {
        $('#OutputDiv').html(data);
    }
);
```

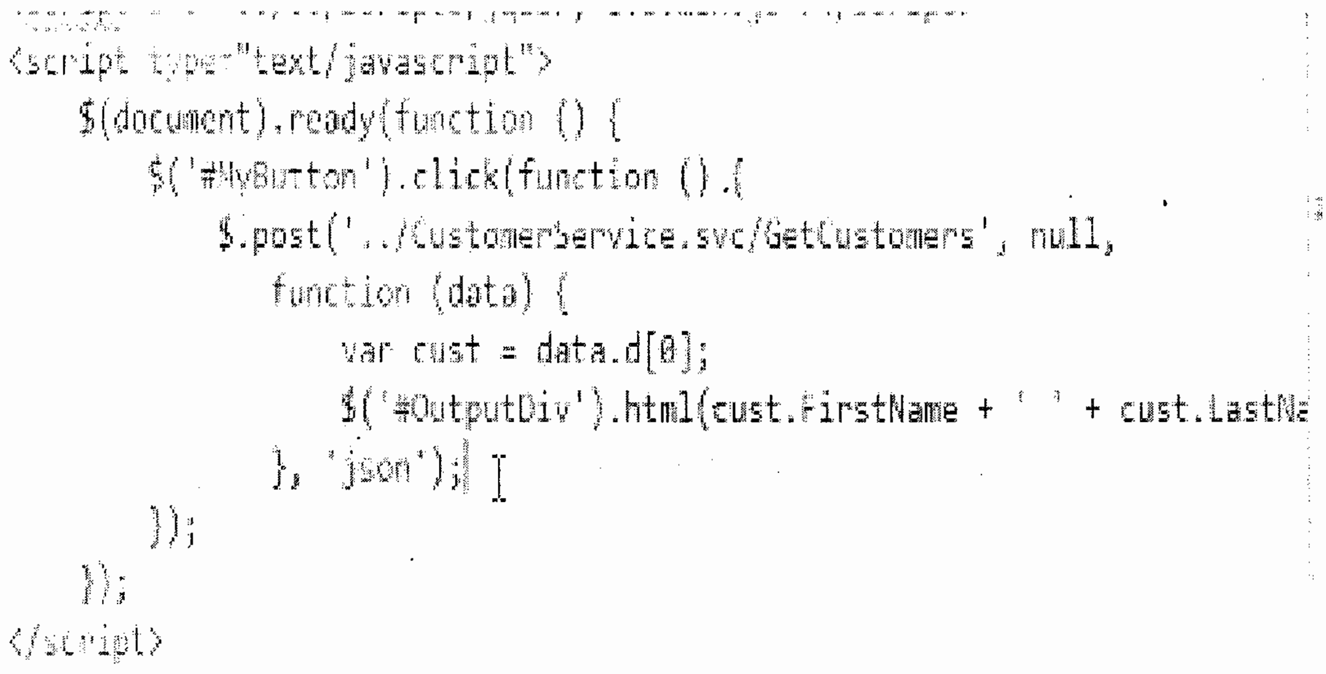
Using post() to Call a WCF Service

- post() can also be used to interact with an Ajax-enabled WCF service:

```
$.post('CustomerService.svc/GetCustomers',
  null, function (data) {
       var cust = data.d[0];
       alert(cust.FirstName + ' ' +
       cust.LastName);
  }, 'json');
```



```
<script type="text/javascript">
    $(document).ready(function () {
        $('#MyButton').click(function () {
            $.post('../GetCustomers.aspx', { PageSize: 15 },
                function (data) {
                    $('#OutputDiv').html(data);
                });
        });
    });
</script>
```



```
<script type="text/javascript">
    $(document).ready(function () {
        $('#MyButton').click(function () {
            $.post('../CustomerService.svc/GetCustomers', null,
                function (data) {
                    var cust = data.d[0];
                    $('#OutputDiv').html(cust.FirstName + ' ' + cust.LastNa
                }, 'json');
        });
    });
</script>
```

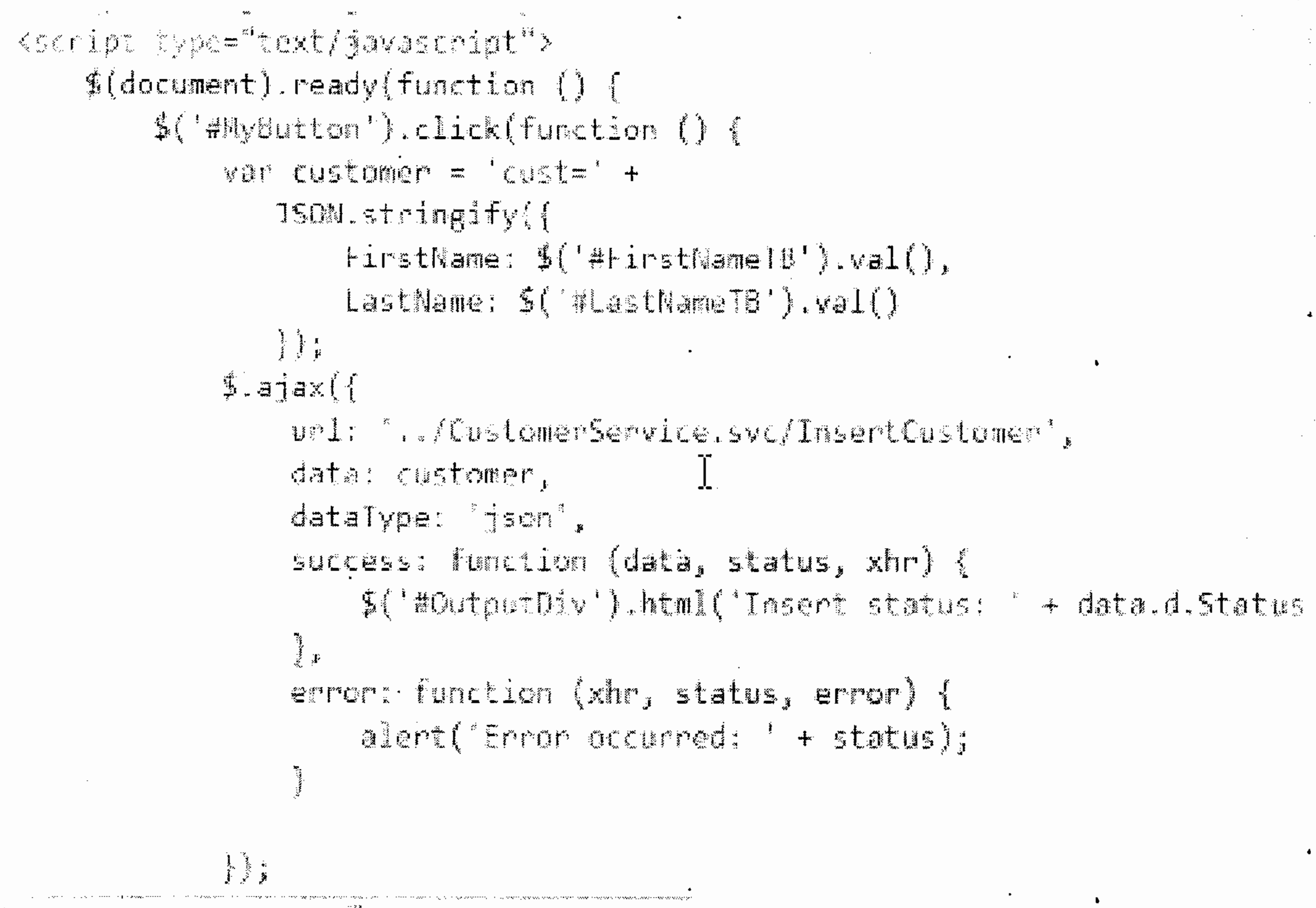
The ajax() Function

- **The ajax() function provides extra control over making Ajax calls to a server**
- **Configure using JSON properties:**
  - contentType
  - data
  - dataType
  - error
  - success
  - type (GET or POST)

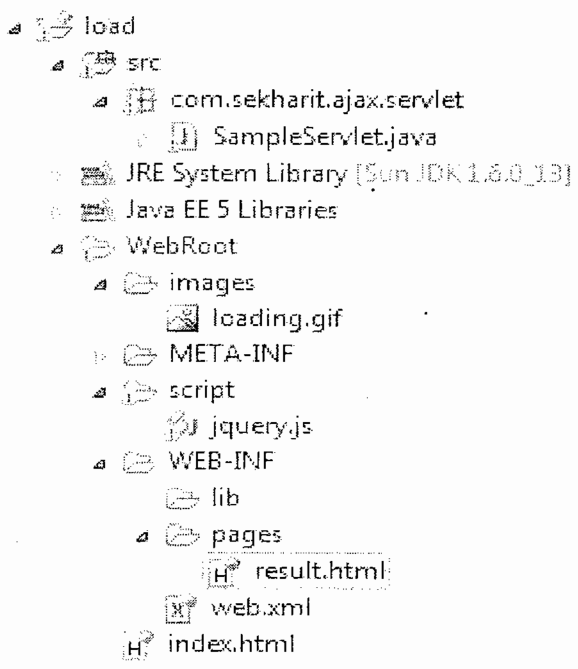
Using the ajax() Function

The ajax() function is configured by assigning values to JSON properties:

```
$.ajax({
   url: '../CustomerService.svc/InsertCustomer',
   data: customer,
   dataType: 'json',
   success: function (data, status, xhr) {
      alert("Insert status: " + data.d.Status + '\n' +
            data.d.Message);
   },
   error: function (xhr, status, error) {
      alert('Error occurred: ' + status);
   }
```



```
<script type="text/javascript">
    $(document).ready(function () {
        $('#MyButton').click(function () {
            var customer = 'cust=' +
               JSON.stringify({
                   FirstName: $('#FirstNameTB').val(),
                   LastName: $('#LastNameTB').val()
               });
            $.ajax({
               url: '../CustomerService.svc/InsertCustomer',
               data: customer,
               dataType: 'json',
               success: function (data, status, xhr) {
                   $('#OutputDiv').html('Insert status: ' + data.d.Status
               },
               error: function (xhr, status, error) {
                   alert('Error occurred: ' + status);
               }

            });
```



```
load
  src
    com.sekharit.ajax.servlet
      SampleServlet.java
  JRE System Library [Sun JDK 1.6.0_13]
  Java EE 5 Libraries
  WebRoot
    images
      loading.gif
    META-INF
    script
      jquery.js
    WEB-INF
      lib
      pages
        result.html
      web.xml
    index.html
```

SampleServlet.java

```
1. package com.sekharit.ajax.servlet;
```

```

import java.io.IOException;

import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

public class SampleServlet extends HttpServlet {

        public void doGet(HttpServletRequest request, HttpServletResponse response)
                    throws ServletException, IOException {
                try {
                        Thread.sleep(5000);
                } catch (InterruptedException e) {
                        e.printStackTrace();
                }
                request.getRequestDispatcher(
                                "/WEB-INF/pages/result.html").forward(request, response);
        }

}
```

web.xml

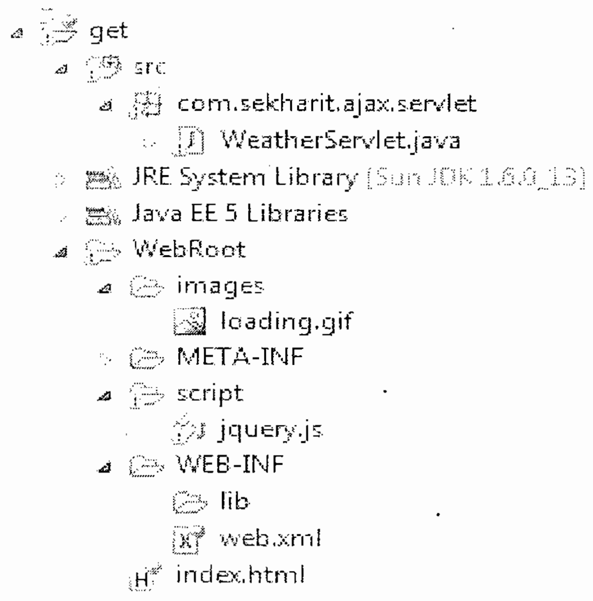
```
<web-app>
    <servlet>
        <servlet-name>SampleServlet</servlet-name>
        <servlet-class>com.sekharit.ajax.servlet.SampleServlet</servlet-class>
    </servlet>

    <servlet-mapping>
        <servlet-name>SampleServlet</servlet-name>
        <url-pattern>/SampleServlet</url-pattern>
        </servlet-mapping>
        <welcome-file-list>
            <welcome-file>index.html</welcome-file>
        </welcome-file-list>
    </web-app>
```

index.html

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

        $("button").click(function(event) {
            $('#resultId').html('<img  src="images/loading.gif"> </img>
                                                    <br/> Please Wait...');
                $('#resultId').load('./SampleServlet');

            });

            });
    </script>
    <style type="text/css">
      .myclass{
            background-color: pink;
            font-size: 50px;
            border: 2px solid green;
            height: 300px;
            width: 400px;
      }
    </style>
    </head>
    <body bgcolor="#c0f9fc">
    <center>
            <h1>Ajax Web Application</h1>
            <h3>By using <i>selector.load(...)</i></h3>
            <hr/> <hr/>

            <div id="resultId" class="myclass">
                  Initial Content
            </div>
            <button>Load Data</button>
    </center>
    </body>
    </html>
```

result.html

```
<div>This is the data from result.html</div>
```



```
get
  src
    com.sekharit.ajax.servlet
      WeatherServlet.java
  JRE System Library [Sun JDK 1.6.0_13]
  Java EE 5 Libraries
  WebRoot
    images
      loading.gif
    META-INF
    script
      jquery.js
    WEB-INF
      lib
      web.xml
    index.html
```

WeatherServlet.java

```
package com.sekharit.ajax.servlet;

import java.io.IOException;
import java.io.PrintWriter;

import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

   public class WeatherServlet extends HttpServlet {

         public void doGet(HttpServletRequest request, HttpServletResponse response)
                     throws ServletException, IOException {
               try {
                     Thread.sleep(5000);
               } catch (InterruptedException e) {
                     e.printStackTrace();
               }
               String city = request.getParameter("cityName");
               String report = getWeather(city);
               response.setContentType("text/xml");
               PrintWriter out = response.getWriter();
               out.println("<weather>");
               out.println("<city>"+city+"</city>");
               out.println("<report>" + report + "</report>");
               out.println("</weather>");
               out.flush();
               out.close();
         }

         private String getWeather(String city) {
               String report;

               if (city.equalsIgnoreCase("hyderabad")) {
                     report = "Currently it is not raining in hyderabad.
                                                         Average temperature is 20";
               } else if (city.equalsIgnoreCase("chennai")) {
                     report = "It's a rainy season in Chennai now.
                                           Better get a umbrella before going out.";
               } else if (city.equalsIgnoreCase("bangalore")) {
                     report = "It's mostly cloudy in Bangalore. Good weather
                                                              for a cricket match.";
               } else {
                     report = "The City you have entered is not present in our system.
                               May be it has been destroyed in last World War or not
                                                           yet built by the mankind";
               }
               return report;
         }

   }
```

web.xml

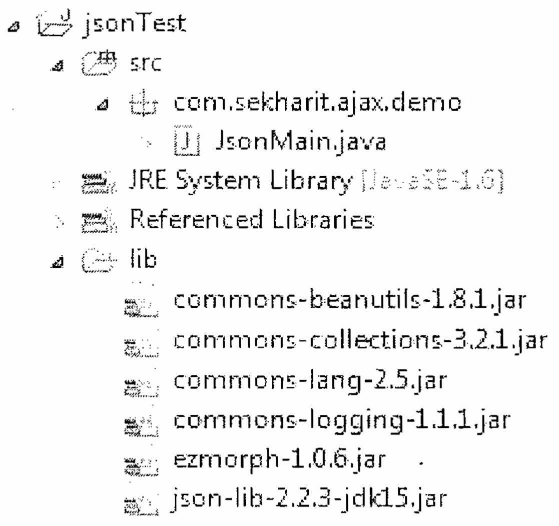
```
<web-app >
   <servlet>
         <servlet-name>WeatherServlet</servlet-name>
         <servlet-class>com.sekharit.ajax.servlet.WeatherServlet</servlet-class>
   </servlet>
```

```
    <servlet-mapping>
          <servlet-name>WeatherServlet</servlet-name>
          <url-pattern>/WeatherServlet</url-pattern>
    </servlet-mapping>
          <welcome-file-list>
                <welcome-file>index.html</welcome-file>
          </welcome-file-list>
    </web-app>
```

inex.html

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

    $("button").click(function(){

          $('div#reportResultId').html('<img  src="images/loading.gif"> </img>
                                                        <br/> Please Wait...');
                 var cityName = $("input#cityName").val();

                 $.get("./WeatherServlet", {"cityName": cityName}, function(xml) {
                       var city = $("city", xml).text();
                       var report = $("report", xml).text();
                       var result = city + " : "+ report;
                 $("div#reportResultId").html(result);
                 });

          });


    });
    </script>
    <style type="text/css">
      .myclass{
          background-color: pink;
          border: 2px solid green;
          font-size:25px;
          height: 300px;
          width: 400px;
      }
    </style>
    </head>
    <body bgcolor="#c0f9fc" >
       <center>
                <h1>Ajax Web Application</h1>
                <h3>By using <i>$.get(...)</i></h3>
                <hr/> <hr/>
                Enter City :
                <input type="text" id="cityName" size="30" />
                <button>Get Weather Report</button>
                <br/>
                <div id="reportResultId" class="myclass" >
                </div>
```

```
        </center>
    </body>
</html>
```

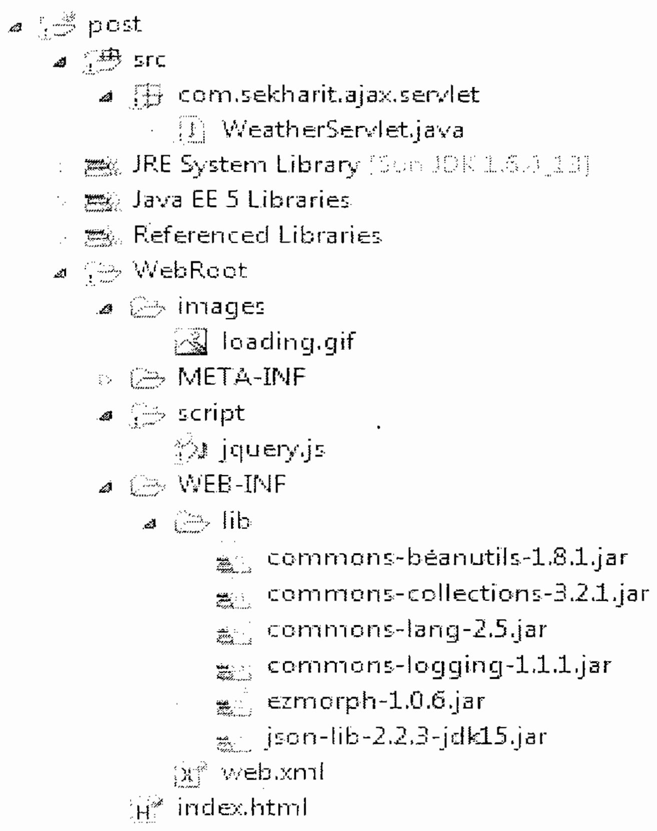


```
jsonTest
    src
        com.sekharit.ajax.demo
            JsonMain.java
    JRE System Library [JavaSE-1.6]
    Referenced Libraries
    lib
        commons-beanutils-1.8.1.jar
        commons-collections-3.2.1.jar
        commons-lang-2.5.jar
        commons-logging-1.1.1.jar
        ezmorph-1.0.6.jar
        json-lib-2.2.3-jdk15.jar
```

JsonMain.java

```
package com.sekharit.ajax.demo;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

import net.sf.json.JSONObject;

public class JsonMain {
        public static void main(String[] args) {

                Map<String, Long> map = new HashMap<String, Long>();
                map.put("A", 10L);
                map.put("B", 20L);
                map.put("C", 30L);

                List<String> list = new ArrayList<String>();
                list.add("Sunday");
                list.add("Monday");
                list.add("Tuesday");

                JSONObject json = new JSONObject();

                // adding properties to json object
                json.put("city", "Mumbai");
                json.put("country", "India");

                // adding map to json
                json.accumulateAll(map);

                // adding list to json
                json.accumulate("weekdays", list);

```

```
                        System.out.println(json.toString());
                }
        }
```



```
post
  src
    com.sekharit.ajax.servlet
      WeatherServlet.java
  JRE System Library [Sun JDK 1.6.0_13]
  Java EE 5 Libraries
  Referenced Libraries
  WebRoot
    images
      loading.gif
    META-INF
    script
      jquery.js
    WEB-INF
      lib
        commons-beanutils-1.8.1.jar
        commons-collections-3.2.1.jar
        commons-lang-2.5.jar
        commons-logging-1.1.1.jar
        ezmorph-1.0.6.jar
        json-lib-2.2.3-jdk15.jar
      web.xml
    index.html
```

WeatherServlet.java

```
package com.sekharit.ajax.servlet;

import java.io.IOException;
import java.io.PrintWriter;

import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import net.sf.json.JSONObject;

public class WeatherServlet extends HttpServlet {

        public void doPost(HttpServletRequest request, HttpServletResponse response)
                        throws ServletException, IOException {
                try {
                        Thread.sleep(5000);
                } catch (InterruptedException e) {
                        e.printStackTrace();
                }
                String city = request.getParameter("cityName");
                String report = getWeather(city);
                response.setContentType("application/json");
```

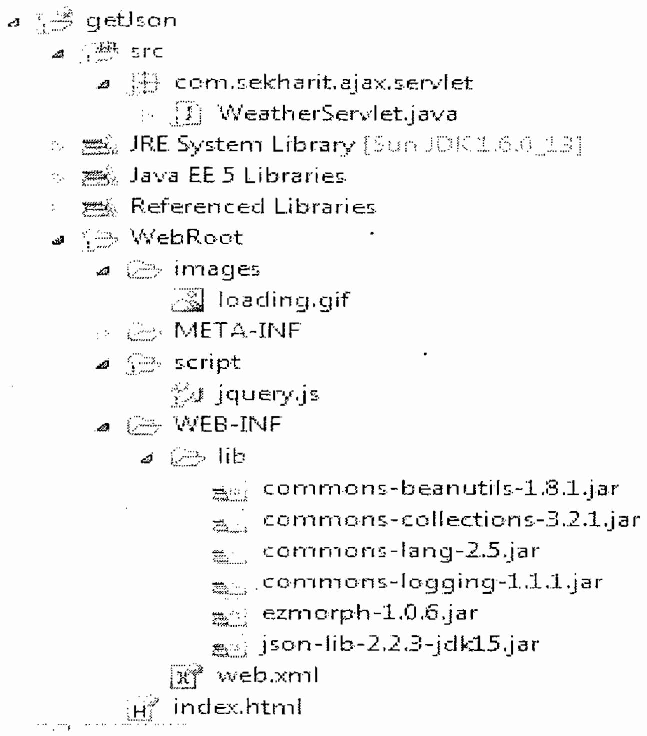
```
            PrintWriter out = response.getWriter();

            JSONObject json = new JSONObject();
            // adding properties to json object
            json.put("city", city);
            json.put("report", report);
            out.println(json);
            out.flush();
            out.close();
        }

        private String getWeather(String city) {
            String report;

            if (city.equalsIgnoreCase("hyderabad")) {
                report = "Currently it is not raining in hyderabad.
                                        Average temperature is 20";
            } else if (city.equalsIgnoreCase("chennai")) {
                report = "It's a rainy season in Chennai now.
                                Better get a umbrella before going out.";
            } else if (city.equalsIgnoreCase("bangalore")) {
                report = "It's mostly cloudy in Bangalore. Good weather
                                            for a cricket match.";
            } else {
                report = "The City you have entered is not present in our system.
                      May be it has been destroyed in last World War or not
                                           yet built by the mankind";
            }
            return report;
        }

    }
```

web.xml

```
<web-app >
   <servlet>
         <servlet-name>WeatherServlet</servlet-name>
         <servlet-class>com.sekharit.ajax.servlet.WeatherServlet</servlet-class>
   </servlet>
   <servlet-mapping>
         <servlet-name>WeatherServlet</servlet-name>
         <url-pattern>/WeatherServlet</url-pattern>
   </servlet-mapping>
         <welcome-file-list>
               <welcome-file>index.html</welcome-file>
         </welcome-file-list>
</web-app>
```

index.html

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

    $("button").click(function(){

        $('div#reportResultId').html('<img  src="images/loading.gif"> </img>
                                                        <br/> Please Wait...');
            var cityName = $("#cityName").val();

            $.post("./WeatherServlet", {"cityName": cityName}, function(json)  {
                var city = json.city;
                var report = json.report;
                var result = city + " : "+ report;
            $("div#reportResultId").html(result);
            });

    });


});
</script>
<style type="text/css">
  .myclass{
     background-color: pink;
     border: 2px solid green;
     font-size:25px;
     height: 300px;
     width: 400px;
  }
</style>
</head>
<body bgcolor="#c0f9fc" >
   <center>
          <h1>Ajax Web Application</h1>
          <h3>By using <i>$.post(...)</i></h3>
          <hr/> <hr/>
          Enter City :
          <input type="text" name="cityName" id="cityName" size="30" />
        <button>Get Weather Report</button>
          <br/>
          <div id="reportResultId" class="myclass" >
          </div>
   </center>
</body>
</html>
```



```
getJson
    src
        com.sekharit.ajax.servlet
            WeatherServlet.java
    JRE System Library [Sun JDK 1.6.0_13]
    Java EE 5 Libraries
    Referenced Libraries
    WebRoot
        images
            loading.gif
        META-INF
        script
            jquery.js
        WEB-INF
            lib
                commons-beanutils-1.8.1.jar
                commons-collections-3.2.1.jar
                commons-lang-2.5.jar
                commons-logging-1.1.1.jar
                ezmorph-1.0.6.jar
                json-lib-2.2.3-jdk15.jar
            web.xml
        index.html
```

WeatherServlet.java

```java
package com.sekharit.ajax.servlet;

import java.io.IOException;
import java.io.PrintWriter;

import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import net.sf.json.JSONObject;

public class WeatherServlet extends HttpServlet {

        public void doGet(HttpServletRequest request, HttpServletResponse response)
                        throws ServletException, IOException {
                try {
                        Thread.sleep(5000);
                } catch (InterruptedException e) {
                        e.printStackTrace();
                }
                String city = request.getParameter("cityName");
                String report = getWeather(city);
                response.setContentType("application/json");
                PrintWriter out = response.getWriter();

                JSONObject json = new JSONObject();
                // adding properties to json object
                json.put("city", city);
                json.put("report", report);
                out.println(json);
```

```
            out.flush();
            out.close();
      }

      private String getWeather(String city) {
            String report;

            if (city.equalsIgnoreCase("hyderabad")) {
                  report = "Currently it is not raining in hyderabad.
                                                Average temperature is 20";
            } else if (city.equalsIgnoreCase("chennai")) {
                  report = "It's a rainy season in Chennai now. Better
                                         get a umbrella before going out.";
            } else if (city.equalsIgnoreCase("bangalore")) {
                  report = "It's mostly cloudy in Bangalore. Good weather
                                                     for a cricket match.";
            } else {
                  report = "The City you have entered is not present in our system.
                            May be it has been destroyed in last World War or not
                                                     yet built by the mankind";
            }
            return report;
      }

}
```

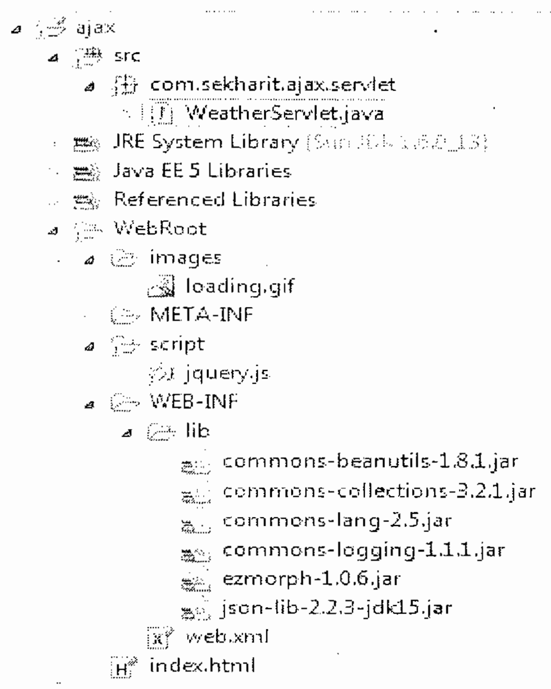
web.xml

```
<web-app >
   <servlet>
         <servlet-name>WeatherServlet</servlet-name>
         <servlet-class>com.sekharit.ajax.servlet.WeatherServlet</servlet-class>
   </servlet>
   <servlet-mapping>
         <servlet-name>WeatherServlet</servlet-name>
         <url-pattern>/WeatherServlet</url-pattern>
   </servlet-mapping>
         <welcome-file-list>
               <welcome-file>index.html</welcome-file>
         </welcome-file-list>
</web-app>
```

index.html

```
<html>
<head>
<script type="text/javascript" src="script/jquery.js" ></script>
<script type="text/javascript">
$(document).ready(function(){
   $("button").click(function(){

          $('div#reportResultId').html('<img  src="images/loading.gif"> </img> <br/>
Please Wait...');
          var cityName = $("#cityName").val();
               $.getJSON("./WeatherServlet", {"cityName": cityName}, function(json) {
                var city = json.city;
```

```
                        var report = json.report;
                        var result = city + " : "+ report;
                  $("div#reportResultId").html(result);
                  });

           });
     });
     </script>
     <style type="text/css">
       .myclass{
           background-color: pink;
           border: 2px solid green;
           font-size:25px;
           height: 300px;
           width: 400px;
       }
     </style>
     </head>
     <body bgcolor="#c0f9fc" >
        <center>
                 <h1>Ajax Web Application</h1>
                 <h3>By using <i>$.getJson(...)</i></h3>
                 <hr/> <hr/>
                 Enter City :
                 <input type="text" id="cityName" size="30" />
                 <button>Get Weather Report</button>
                 <br/>
                 <div id="reportResultId" class="myclass" >
                 </div>
        </center>
     </body>
     </html>
```



```
ajax
    src
        com.sekharit.ajax.servlet
            WeatherServlet.java
    JRE System Library [Sun JDK 1.6.0_13]
    Java EE 5 Libraries
    Referenced Libraries
    WebRoot
        images
            loading.gif
        META-INF
        script
            jquery.js
        WEB-INF
            lib
                commons-beanutils-1.8.1.jar
                commons-collections-3.2.1.jar
                commons-lang-2.5.jar
                commons-logging-1.1.1.jar
                ezmorph-1.0.6.jar
                json-lib-2.2.3-jdk15.jar
            web.xml
        index.html
```

WeatherServlet.java

```java
package com.sekharit.ajax.servlet;

import java.io.IOException;
import java.io.PrintWriter;

import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import net.sf.json.JSONObject;

public class WeatherServlet extends HttpServlet {

    public void doPost(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        try {
            Thread.sleep(5000);
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
        String city = request.getParameter("cityName");
        String report = getWeather(city);
        response.setContentType("application/json");
        PrintWriter out = response.getWriter();

        JSONObject json = new JSONObject();
        // adding properties to json object
        json.put("city", city);
        json.put("report", report);
        System.out.println(json);
        out.println(json);
```

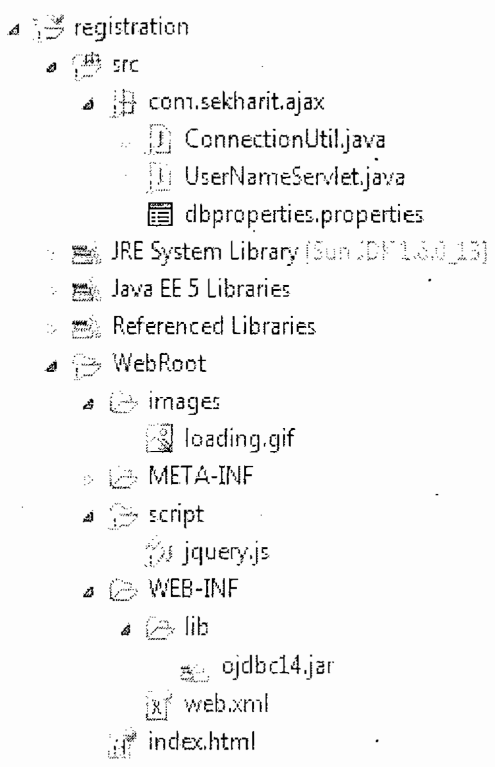
```
            out.flush();
            out.close();
    }

    private String getWeather(String city) {
            String report;

            if (city.equalsIgnoreCase("hyderabad")) {
                    report = "Currently it is not raining in hyderabad.
                                                    Average temperature is 20";
            } else if (city.equalsIgnoreCase("chennai")) {
                    report = "It's a rainy season in Chennai now. Better
                                            get a umbrella before going out.";
            } else if (city.equalsIgnoreCase("bangalore")) {
                    report = "It's mostly cloudy in Bangalore. Good weather
                                                            for a cricket match.";
            } else {
                    report = "The City you have entered is not present in our system.
                            May be it has been destroyed in last World War or not
                                                    yet built by the mankind";
            }
            return report;
    }

}
```

web.xml

```
<web-app >
    <servlet>
          <servlet-name>WeatherServlet</servlet-name>
          <servlet-class>com.sekharit.ajax.servlet.WeatherServlet</servlet-class>
    </servlet>
    <servlet-mapping>
          <servlet-name>WeatherServlet</servlet-name>
          <url-pattern>/WeatherServlet</url-pattern>
    </servlet-mapping>
          <welcome-file-list>
                <welcome-file>index.html</welcome-file>
          </welcome-file-list>
</web-app>

```

index.html

```
<html>
<head>
<script type="text/javascript" src="script/jquery.js" ></script>
<script type="text/javascript">
$(document).ready(function(){
   $("button").click(function(){
          $('div#reportResultId').html('<img  src="images/loading.gif"> </img>
                                                          <br/> Please Wait...');
          var cityName = $("#cityName").val();
                $.ajax(
                {
                      url:"./WeatherServlet",
                        data:{"cityName": cityName},
                        dataType:'json',
                        type:'POST',
                        success: function(data, status, xhr){
                               var city = data.city;
                               var report = data.report;
                               var result = city + " : "+ report;
                               $("div#reportResultId").html(result);
                        },
                        error: function(xhr, status, error){
                              alert("success"+error);
                              $("div#reportResultId").html("Processing Error...");
                        }
                   });

            });
      });
      </script>
      <style type="text/css">
        .myclass{
            background-color: pink;
            border: 2px solid green;
            font-size:25px;
            height: 300px;
            width: 400px;
        }
      </style>
      </head>
      <body bgcolor="#c0f9fc" >
         <center>
                  <h1>Ajax Web Application</h1>
                  <h3>By using <i>$.ajax(...)</i></h3>
                  <hr/> <hr/>
                  Enter City :
                  <input type="text" name="cityName" id="cityName" size="30" />
                <button>Get Weather Report</button>
                  <br/>
                  <div id="reportResultId" class="myclass" >
                  </div>
         </center>
      </body>
      </html>
```



```
registration
    src
        com.sekharit.ajax
            ConnectionUtil.java
            UserNameServlet.java
            dbproperties.properties
    JRE System Library [Sun JDK 1.6.0_13]
    Java EE 5 Libraries
    Referenced Libraries
    WebRoot
        images
            loading.gif
        META-INF
        script
            jquery.js
        WEB-INF
            lib
                ojdbc14.jar
            web.xml
        index.html
```

dbproperties.properties

```
driverClass=oracle.jdbc.driver.OracleDriver
url=jdbc:oracle:thin:@localhost:1521:XE
username=system
password=tiger
```

ConnectionUtil.java

```
package com.sekharit.ajax;

import java.io.IOException;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.HashMap;
import java.util.Properties;

public class ConnectionUtil {

      private static HashMap dbProps;
      static {
            try {
                  dbProps = new HashMap();
                  Properties properties = new Properties();
                  properties
                              .load(ConnectionUtil.class
                                          .getClassLoader()
                                          .getResourceAsStream(
"com/sekharit/ajax/dbproperties.properties"));
                dbProps.putAll(properties);
                Class.forName((String) dbProps.get("driverClass"));

            } catch (ClassNotFoundException e) {
                e.printStackTrace();
            } catch (IOException e) {
                e.printStackTrace();
            }
        }

        public static Connection getConnection() {
            Connection connection = null;
            try {
                connection = DriverManager.getConnection((String) dbProps
                        .get("url"), (String) dbProps.get("username"),
                        (String) dbProps.get("password"));
            } catch (SQLException e) {
                e.printStackTrace();
            }
            return connection;
        }

        public static void closeConnection(Connection con) {
            if (con != null) {
                try {
                    con.close();
                } catch (SQLException e) {
                    e.printStackTrace();
                }
            }

        }

        public static void closeConnection(Connection con, Statement st) {
            closeConnection(con);
            if (st != null) {
                try {
                    st.close();
                } catch (SQLException e) {
                    e.printStackTrace();
                }
            }

        }

        public static void closeConnection(Connection con, Statement st,
                ResultSet rs) {
            closeConnection(con, st);
            if (rs != null) {
                try {
                    rs.close();
                } catch (SQLException e) {
                    e.printStackTrace();
                }
            }
        }

}
```

UserNameServlet.java

```
package com.sekharit.ajax;

import java.io.IOException;
import java.io.PrintWriter;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;

import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

public class UserNameServlet extends HttpServlet {
      private String QUERY = "SELECT * FROM USER_TAB WHERE USER_ID=?";

      public void doPost(HttpServletRequest request, HttpServletResponse response)
                  throws ServletException, IOException {

            try {
                  Thread.sleep(5000);
            } catch (InterruptedException e) {
                  e.printStackTrace();
            }
            response.setContentType("text/xml");
            String userId = request.getParameter("userId");
            PrintWriter out = response.getWriter();
            StringBuffer buffer = new StringBuffer();
            buffer.append("<resp>");
            buffer.append("<valid>");
            if (userId == null || userId.trim().length() == 0) {
                  buffer.append("EMPTY");
            } else {
                  Connection connection = null;
                  PreparedStatement ps = null;
                  ResultSet rs = null;
                  try {
                        connection = ConnectionUtil.getConnection();
                        ps = connection.prepareStatement(QUERY);
                        ps.setString(1, userId);
                        rs = ps.executeQuery();
                        if (rs.next()) {
                              buffer.append("NO");
                        } else {
                              buffer.append("YES");
                        }
                  } catch (Exception e) {
                        buffer.append("ERROR");
```

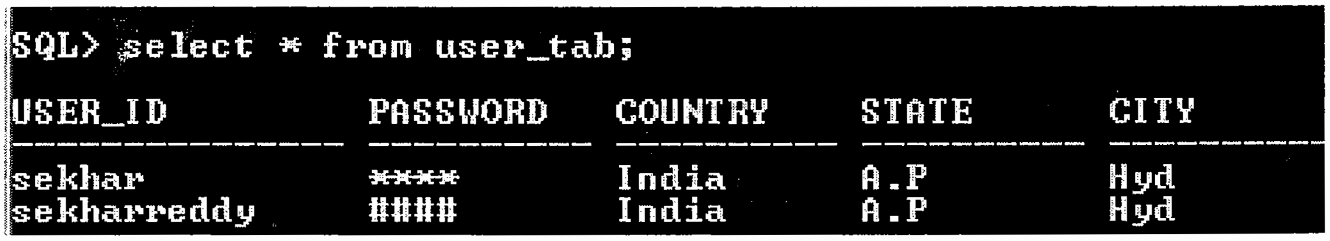
```
                    }
                }
                buffer.append("</valid>");
                buffer.append("</resp>");
                out.println(buffer.toString());
                out.flush();
                out.close();
        }

}
```

web.xml

```
<web-app >
  <servlet>
    <servlet-name>UserNameServlet</servlet-name>
    <servlet-class>com.sekharit.ajax.UserNameServlet</servlet-class>
  </servlet>

  <servlet-mapping>
    <servlet-name>UserNameServlet</servlet-name>
    <url-pattern>/userNameServlet</url-pattern>
     </servlet-mapping>
     <welcome-file-list>
       <welcome-file>index.html</welcome-file>
     </welcome-file-list>
   </web-app>
```

index.html

```
<html>
<head>
<script type="text/javascript" src="script/jquery.js" ></script>
<script type="text/javascript">
$(document).ready(function(){
   $("#usernameId").change(function(){
          hideAll();
          $('span#loadingId').show();
          var username = $("#usernameId").val();
               $.ajax(
               {
                    url:"./userNameServlet",
                    data:{"userId": username},
                    dataType:'xml',
                    type:'POST',
                    success: function(data, status, xhr){
                           var result = $("valid", $(data)).text();
                           if(result == 'YES'){
                                hideAll();.
                                $("span#rightId").show();
                           } else if(result == 'NO'){
                                hideAll();
                                $("span#wrongId").show();
                           } else if(result == 'EMPTY'){
                                hideAll();
                                $("span#emptyId").show();
                                } else if(result == 'ERROR'){
                                        hideAll();
                                        $("span#errorId").show();
                                }

                        },
                        error: function(xhr, status, error){
                                        hideAll();
                                        $("span#errorId").show();
                        }
                });

        });
});

function hideAll(){
        $('span#loadingId').hide();
        $('span#wrongId').hide();
        $('span#rightId').hide();
        $('span#emptyId').hide();
}
</script>
<style type="text/css">
  .myclass{
        background-color: pink;
        border: 2px solid green;
        font-size:25px;
        height: 300px;
        width: 400px;
        float: left;
  }
  img{
        height: 20px;
        width: 20px;
  }
</style>
</head>
<body bgcolor="#c0f9fc" >
   <center>
                <h1>Ajax Web Application</h1>
                <h3>By using <i>$.ajax(...)</i></h3>
                <hr/> <hr/>
                        <h3> Registraion Page </h3>
                <table border="1" >
                        <tr>
                                <td>Name</td>
                                <td>
                                        <input type="text" id="usernameId" name="name"/>
                                        <span id="loadingId" style="display: none" >
                                                <img  src="images/loading.gif"> </img>
                                        </span>
                                <span id="wrongId" style="display: none; color: red" >
                                                User Name not Available
                                        </span>
                                <span id="rightId" style="display: none; color: green" >
                                User Name Available
                           </span>
                     <span id="emptyId" style="display: none; color: red" >
                                User Name can't be empty
                           </span>
                     <span id="errorId" style="display: none; color: red" >
                                Processing Error...
                           </span>
                     </td>
               </tr>
               <tr>
                     <td>Password</td>
                     <td>
                           <input type="password" name="password"/>
                     </td>
               </tr>
               <tr>
                     <td>Address</td>
                     <td>
                           <textarea cols="50" rows="4" > </textarea>
                     </td>
               </tr>
               <tr>
                     <td colspan="2" align="center" >
                           <input type="submit"  value="Register">
                     </td>
               </tr>
         </table>
   </center>
</body>
</html>
```



```
SQL> select * from user_tab;

USER_ID              PASSWORD      COUNTRY       STATE         CITY
-------------------- ------------- ------------- ------------- ----------
sekhar               *****         India         A.P           Hyd
sekharreddy          ####          India         A.P           Hyd
```

Reference websites:

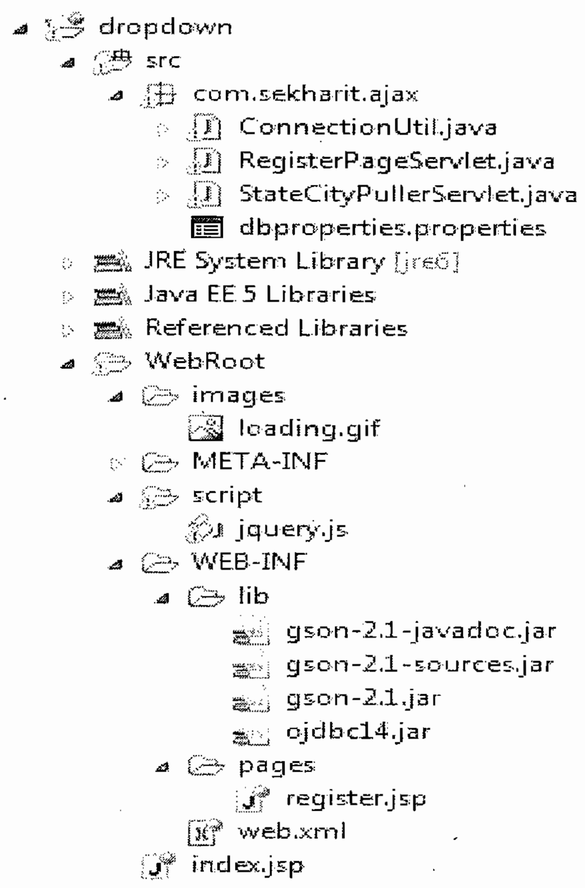
Jquery.com

Jqueryui.com

http://codylindley.com/jqueryselectors/

http://appendto.com/community/jquery-vsdoc

http://james.padolsey.com/jquery/#v=1.6.2&fn=jQuery.fn.hide



```
dropdown
  src
    com.sekharit.ajax
      ConnectionUtil.java
      RegisterPageServlet.java
      StateCityPullerServlet.java
      dbproperties.properties
  JRE System Library [jre6]
  Java EE 5 Libraries
  Referenced Libraries
  WebRoot
    images
      loading.gif
    META-INF
    script
      jquery.js
    WEB-INF
      lib
        gson-2.1-javadoc.jar
        gson-2.1-sources.jar
        gson-2.1.jar
        ojdbc14.jar
      pages
        register.jsp
      web.xml
    index.jsp
```

dbproperties.properties

--SAME AS ABOVE --

ConnectionUtil.java

--SAME AS ABOVE --

RegisterPageServlet.java

```java
package com.sekharit.ajax;

import java.io.IOException;
import java.sql.Connection;
import java.sql.ResultSet;
import java.sql.Statement;
import java.util.HashMap;
import java.util.Map;

import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

public class RegisterPageServlet extends HttpServlet {
      private String COUNTRIES_QUERY = "SELECT * FROM COUNTRIES";

      public void doGet(HttpServletRequest request, HttpServletResponse response)
                  throws ServletException, IOException {
            Map<String, String> countries = new HashMap<String, String>();
            Connection connection = null;
            Statement st = null;
```

```
        ResultSet rs = null;
        try {
            connection = ConnectionUtil.getConnection();
            st = connection.createStatement();
            rs = st.executeQuery(COUNTRIES_QUERY);
            while (rs.next()) {
                countries.put(rs.getString(1), rs.getString(2));
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
        request.setAttribute("countries", countries);
        request.getRequestDispatcher("/WEB-INF/pages/register.jsp").forward(
                request, response);
    }

}
```

StateCityPullerServlet.java

```
package com.sekharit.ajax;

import java.io.IOException;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.util.HashMap;
import java.util.Map;

import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import com.google.gson.Gson;

public class StateCityPullerServlet extends HttpServlet {
    private String STATES_QUERY = "SELECT * FROM STATES WHERE CID=?";
    private String CITIES_QUERY = "SELECT * FROM CITIES WHERE SID=?";

    public void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {

        try {
            Thread.sleep(5000);
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
        String type = request.getParameter("type");
// Returns "country" or "state".
        String value = request.getParameter("value");
// Value of selected country or state.
        Map<String, String> options = null;
        if (type.equalsIgnoreCase("state")) {
            options = getOptions(STATES_QUERY, value);
        } else if (type.equalsIgnoreCase("city")) {
```

```java
            options = getOptions(CITIES_QUERY, value);
        }
        String json = new Gson().toJson(options);
        // Convert Java object to JSON string.
        response.setContentType("application/json");
        // Inform client that
        response.setCharacterEncoding("UTF-8");
        // Important if you want world
        response.getWriter().write(json);
        // Write JSON string to response.
    }

    private Map<String, String> getOptions(String query, String value) {
        Map<String, String> options = new HashMap<String, String>();
        Connection connection = null;
        PreparedStatement ps = null;
        ResultSet rs = null;
        try {
            connection = ConnectionUtil.getConnection();
            ps = connection.prepareStatement(query);
            ps.setInt(1, Integer.parseInt(value));
            rs = ps.executeQuery();
            while (rs.next()) {
                options.put(rs.getString(1), rs.getString(2));
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
        return options;
    }

}
```

web.xml

```xml
<?xml version="1.0" encoding="UTF-8"?>
<web-app version="2.5" xmlns="http://java.sun.com/xml/ns/javaee"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
    http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd">
    <servlet>
        <servlet-name>RegisterPageServlet</servlet-name>
        <servlet-class>com.sekharit.ajax.RegisterPageServlet</servlet-class>
    </servlet>
        <servlet>
            <servlet-name>StateCityPullerServlet</servlet-name>
            <servlet-class>com.sekharit.ajax.StateCityPullerServlet</servlet-class>
        </servlet>

        <servlet-mapping>
            <servlet-name>RegisterPageServlet</servlet-name>
            <url-pattern>/registerPage</url-pattern>
        </servlet-mapping>
        <servlet-mapping>
            <servlet-name>StateCityPullerServlet</servlet-name>
            <url-pattern>/sateCityPullerServlet</url-pattern>
        </servlet-mapping>

        <welcome-file-list>
              <welcome-file>index.jsp</welcome-file>
        </welcome-file-list>
</web-app>
```

index.jsp

```
<html>
<body bgcolor="#c0f9fc" >
   <center>
              <h1>Ajax Web Application</h1>
        <hr/> <hr/>
        <a href="./registerPage" >Registration</a>
   </center>
</body>
</html>
```

register.jsp

```
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<html>
<head>
<script type="text/javascript" src="script/jquery.js" ></script>
<script type="text/javascript">
$(document).ready(function(){
    $('#country').change(function() {
        var value = $(this).val();
        var type = 'state';
              removeAllPrevOptions();
              fillOptions(type, value);
     });

       $('#state').change(function() {
        var value = $(this).val();
              var type = 'city';
              removeCityPrevOptions()
              fillOptions(type, value);
       });

  });

        function fillOptions(type, value) {
              var loadingId = 'span#'+type+'LoadingId';
              var dropdown = $('#'+type);

              $(loadingId).show();
              $.getJSON('./sateCityPullerServlet', { "type" : type, "value": value },
                                                                       function(opts) {
               if (opts) {
                     $.each(opts, function(key, value) {
                          dropdown.append($('<option/>').val(key).text(value));
                     });
               } else {
                                   alert("No options are available");
```

```
                    }
                    $(loadingId).hide();

        });
    }


    function removeAllPrevOptions(){
          removeStatePrevOptions();
          removeCityPrevOptions();
    }

    function removeStatePrevOptions(){
          var stateDropdown = $('#state');
          $('>option', stateDropdown).remove();
          stateDropdown.append($('<option/>').text('Please select '));
    }

    function removeCityPrevOptions(){
          var cityDropdown = $('#city');
          $('>option', cityDropdown).remove();
          cityDropdown.append($('<option/>').text('Please select '));
    }


    function hideAllLoading(){
          $('stateLoadingId').hide();
          $('cityLoadingId').hide();
    }
</script>
<style type="text/css" >

    table, td, th{
          border-collapse:collapse;
          border : 2px solid red;
          width : 400px;
    }
    img{
          height: 20px;
          width: 20px;
   }
</style>

</head>
<body  bgcolor="#c0f9fc" >
   <center>
          <h1>Ajax Web Application</h1>
          <hr/> <hr/>
          <form>
                <table >
                      <tr>
                            <td> Country </td>
                            <td>
                            <select id="country" name="country">
                              <option selected="selected" >Please select</opt
```

```
                                        <c:forEach items="${countries}" var="country">
                                         <option value="${country.key}" >
                                            ${country.value}
                                         </option>
                                        </c:forEach>
                                      </select>
                                      </td>
                                </tr>
                                <tr>
                                      <td> State </td>
                                      <td>
                                            <select id="state" name="state">
                                                  <option>Please select</option>
                                      </select>
                                      <span id="stateLoadingId" style="display: none" >
                                                  <img  src="images/loading.gif"> </img>
                                            </span>
                                      </td>
                                </tr>
                                <tr>
                                      <td> City </td>
                                      <td>
                                            <select id="city" name="city">
                                         <option>Please select</option>
                                      </select>
                                      <span id="cityLoadingId" style="display: none" >
                                                  <img  src="images/loading.gif"> </img>
                                            </span>
                                      </td>
                                </tr>
                          </table>

                </form>
           </center>
      </body>
      </html>
```

Reference websites:

Jquery.com

Jqueryui.com

http://codylindley.com/jqueryselectors/

http://appendto.com/community/jquery-vsdoc

http://james.padolsey.com/jquery/#v=1.6.2&fn=jQuery.fn.hide