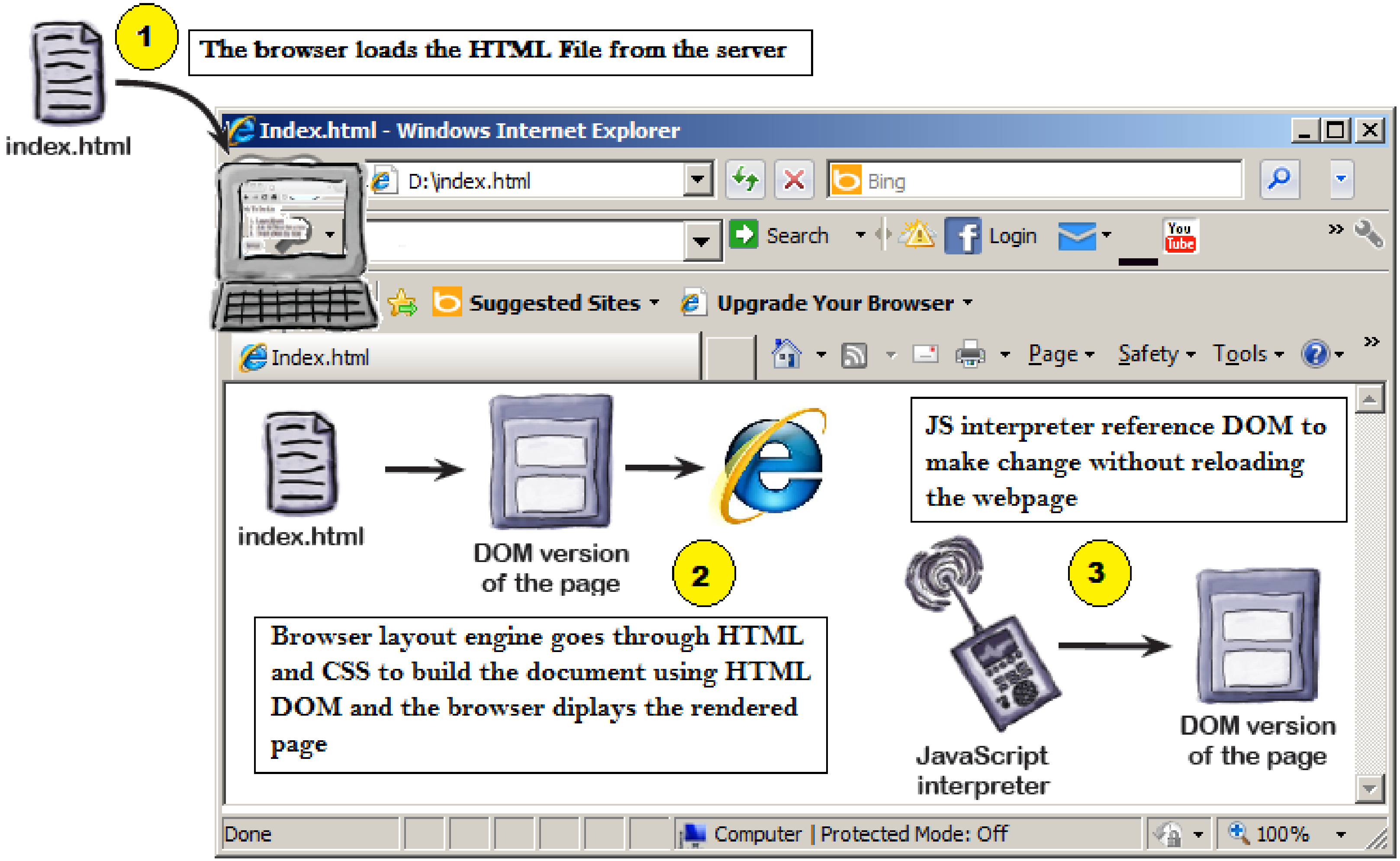
jQuery Fundamentals

How webpage works



The browser loads the HTML File from the server

Browser layout engine goes through HTML and CSS to build the document using HTML DOM and the browser diplays the rendered page

JS interpreter reference DOM to make change without reloading the webpage

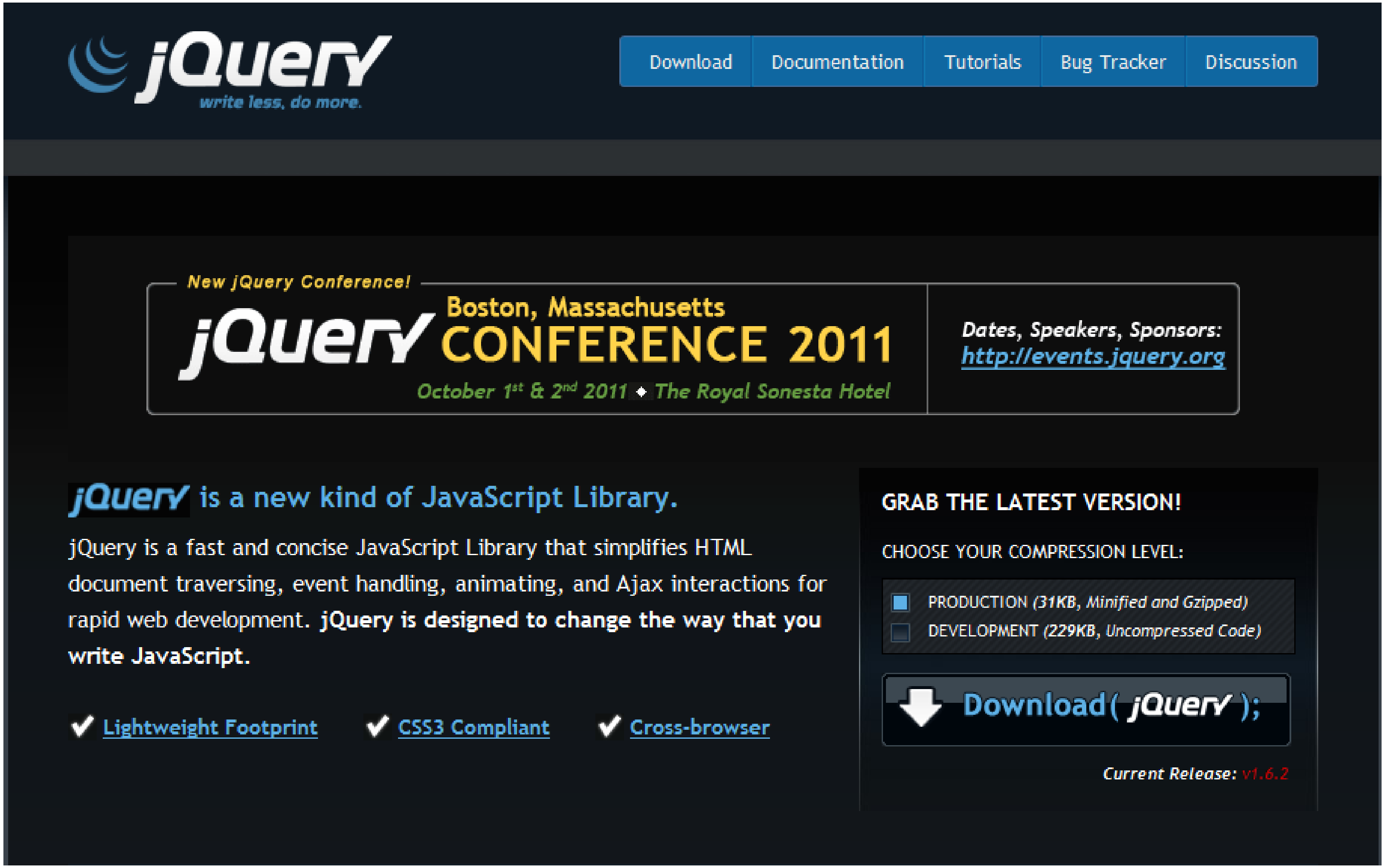
jQuery Introduction

- **jQuery is a JavaScript library (single file)**
- **It supports cross browser**
- **Select HTML elements**
- **Handle Events**
- **Animate HTML elements**
- **Make Ajax calls**
- **1000’s of plug-ins available**

Why jQuery?

- **JavaScript is great for a lot of things especially manipulating the DOM but it's pretty complex stuff. DOM manipulation is by no means straightforward at the base level, and that's where jQuery comes in. It abstracts away a lot of the complexity involved in dealing with the DOM, and makes creating effects super easy.**
- **It can locate elements with a specific class**
- **It can apply styles to multiple elements**
- **It solves the cross browser issues**
- **It supports method chaining**
- **It makes the client side development very easy**

jquery.com (Official Website for jQuery)



New jQuery Conference!

jQuery CONFERENCE 2011 Boston, Massachusetts

October 1st & 2nd 2011 • The Royal Sonesta Hotel

Dates, Speakers, Sponsors: http://events.jquery.org

jQuery is a new kind of JavaScript Library.

jQuery is a fast and concise JavaScript Library that simplifies HTML document traversing, event handling, animating, and Ajax interactions for rapid web development. **jQuery is designed to change the way that you write JavaScript.**

✓ Lightweight Footprint ✓ CSS3 Compliant ✓ Cross-browser

GRAB THE LATEST VERSION!

CHOOSE YOUR COMPRESSION LEVEL:

☑ PRODUCTION *(31KB, Minified and Gzipped)*

□ DEVELOPMENT *(229KB, Uncompressed Code)*

Download(jQuery);

Current Release: v1.6.2

Using jQuery

```
<html>
<head>
<title>Test jQuery</title>
<script type="text/javascript" src="jquery-1.6.4.js"></script>
<script type="text/javascript">
      $(document).ready(function(){
      alert('Hi');  });
</script>
</head>
<body>
      jQuery Enabled
</body>
</html>
```

jQuery code assistance

- **VS 2008 / 2010 provides jQuery Intellisense**
  - http://www.appendto.com/community/jquery-vsdoc
  - Place jQuery-vsdoc.js file into the same path as the jquery.js file
  - http://weblogs.asp.net/scottgu/archive/2008/11/21/jquery-intellisense-in-vs-2008.aspx
- **Other IDE’s**
  - Eclipse(with plug-in) - http://www.eclipse.org/
  - Aptana Studio - http://www.aptana.com/

Content Delivery Network(CDN)

- **Script can be also accessible from**
  - Microsoft - http://ajax.aspnetcdn.com/ajax/jQuery/jquery-1.6.2.min.js
  - jQuery - http://code.jquery.com/jquery-1.6.2.min.js
  - Google - http://ajax.googleapis.com/ajax/libs/jquery/1.6.2/jquery.min.js

- **A CDN short for Content Delivery Network distributes static content across servers in various, diverse physical locations. When a user's browser resolves the URL for these files, their download will automatically target the closest available server in the network.**

- **If jQuery is hosted locally then users must download it at least once. Each of your users probably already has dozens of identical copies of jQuery in their browser's cache, but those copies of jQuery are ignored when they visit your site. Even if someone visits hundreds of sites using the same CDN hosted version of jQuery, they will only need download it once!**

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, so all functionality is available without using $. If we need to use another JavaScript library alongside jQuery, we can return control of $ back to the other library with a call to $.noConflict**

```
<script type="text/javascript" src="other_lib.js"></script>
<script type="text/javascript" src="jquery.js"></script>
<script type="text/javascript">
$.noConflict();
//Code that uses other libraries $ can follow here.
</script>
```

jQuery Selectors

Introduction to selectors

- **jQuery uses same CSS selectors used to style our page to manipulate elements on the page.**
- **CSS selectors select elements to add style to those elements where as jQuery selectors select elements to add behavior to those elements.**
- **Selectors allow page elements to be selected.**
- **Single or Multiple elements are supported.**
- **A Selector identifies an HTML element / tag that will be manipulated with jQuery Code.**
- **Selector Syntax**
  - *$(selectorExpression)*
  - *jQuery(selectorExpression)*

Selecting by Tag Name

- **Selecting single tag takes the following syntax**
  - ***$(‘div’) – selects all <div> elements***
  - ***$(‘a’) – selects all <a> elements***
- **To reference multiple tags, use the (,) to separate the elements**
  - ***$(‘p, div, span’) - selects all paragraphs, divs and span elements***

Selecting Descendants

- **$(‘ancestor descendant’) - selects all the descendants of the ansector**
  - ***$(‘table tr’) - Selects all tr elements that are the descendants of the table element***
- **Descendants can be children, grand children etc of the designated ancestor element.**

Selecting by Element ID

- **It is used to locate the DOM element very fast.**
- **Use the # character to select elements by ID**
  - ***$(‘#myID’) – selects <div id=“myid”> element***

Selecting Elements by Class Name

- **Use the (.) character to select elements by class name**
  - *$('.myclass') - selects <div class="myclass"> element*
- **To reference multiple tags, use the (,) character to separate the class name.**
  - *$('.blueDiv,.redDiv') - selects all the elements containing the class blueDiv and redDiv*
- **Tag names can be combined with elements name as well.**
  - *$('div.myclass') – selects only <div> tags with class="myclass"*

Selecting by attribute values

- **Use brackets [attribute] to select based on attribute name and/or attribute value**
  - ***$(‘a[title]’) - selects all anchor elements that have a title attribute***
  - ***$(‘a[title=“trainer”]’) – selects all <a> elements that have a “trainer” title attribute value***

Selecting by input elements

- **To select all input elements**
  - ***$(':input') - selects input, select, textarea, button,image, radio etc***
  - ***$(':input[type="radio"]') – selects all radio buttons***

Additional Selectors

- **:contains() will select elements that match the contents.**
  - *$('div:contains("KLFS")') - selects div's which contains the text KLFS(match is case sensitive)*
- **$('element:odd') and $('element:even') is the jQuery syntax for selecting odd and even positions respectively.**
  - *Index is 0 based. Odd returns(1,3,5…) and Even returns (0,2,4…)*
- **$('element:first-child') and $('element:last-child') is the jQuery syntax for selecting the first child and last child of every element group.**
  - *$('span:first-child') returns the span which is a first child for all the groups*

Additional Selectors (Contd)

- **$(‘attribute^=“value’) and $(‘attribute$=“value’) is the jQuery syntax for selecting all the elements with an attribute that begins and end with stated value.**
  - **$(‘input[value^=“Company”]’) – selects any input element whose value starts with Company (‘^’ can be replaced with ‘*’ to retrieve the elements contains the value company**

Working with JSON

JSON Introduction

- **Java Script Object Notation**
- **JSON is a lightweight format for exchanging data between the client and server.**
- **JSON is a syntax for passing around objects that contain name/value pairs, arrays and other objects.**
- **It is often used in AJAX applications because of its simplicity and because its format is based on JavaScript object literals.**
- **JSON is language independent and text based. It is easy to parse and generate.**
- **Supported by most of the languages.**

JSON Types

- **Number : integer, real or floating point**
- **String : double-quoted Unicode with backslashes**
- **Boolean : true and false**
- **Array : ordered sequence of comma-separated values enclosed in square brackets**
- **Object : collection of comma-separated “key”:value pairs enclosed in curly braces**
- **null**

Using JSON in jQuery

- **A JSON object can be passed as input into jQuery Methods, so that we can avoid method chaining.**
- **We can parse a JSON string in jQuery using the following syntax**

```
var obj = jQuery.parseJSON(jsonString);
```

Interacting with DOM

Iterating through Nodes

- **.each(function(index,Element)) is used to iterate through jQuery objects**

```
$(‘div’).each (function (index){
alert(index+’=‘+$(this).text());
});
```

- **Iterates through each div element and returns its index number and text**

```
$(‘div’).each (function
(index,element){
alert(index+’=‘+$(element).text());
});
```

Modifying Object Properties

- **The this.ProperyName statement can be used to modify an object’s properties directly.**

```
$(‘div’).each (function (index){
this.title = “Index = “+index;
});
```

- **Iterates through each div and modifies the title. If the property does not exit, it will be added**

Working with Attributes

- **Object attributes can be used using attr():**
  - **var val = $(‘#customDiv’).attr(‘title’); - Retrieves the title attribute value**
- **.attr(attributeName,value) is the method used to access an object’s attributes and modify the values.**
  - **$(‘img’).attr(‘title’,’Image title’); - changes the title attribute value to Image title.**
- **To modify multiple attributes, pass JSON object.**

```
$(‘img’).attr({
“title”:  “image title”,
“style” : “border:2px solid black”
});
```

Adding and Removing Nodes

- **In traditional approach adding and removing nodes is tedious.**
- **To insert nodes four methods were available**
- **Appending adds children at the end of the matching elements**
  - .append()
  - .appendTo()
- **Prepending adds children at the beginning of the matching elements**
  - .prepend()
  - .prependTo()
- **To wrap the elements use .wrap()**
- **To remove nodes from an element use .remove()**

Modifying Styles

- **.css() function is used to modify an object’s style**
  - $('div').css('color','red');
- **Multiple styles can be modified by passing a JSON Object**

```
        $('div').css({
                “color”:”red”,
                “font-weight”:”bold”
            });
```

Working with Classes

- The four methods for working with css class attributes are
  - .addClass()
  - .hasClass()
  - .removeClass()
  - .toggleClass()
- .addClass() adds one or more class names to the class attribute of each element.
  - $(‘p’).addClass(‘classOne’);
  - $(‘p’).addClass(‘classOne classTwo’);
- .hasClass() returns true if the selected element has a matching class that is specified
  - if($('p').hasClass('classOne')) { //perform operation}

Working with Classes (Contd)

- **.removeClass() can remove one or more classes**
  - **$(‘p’).removeClass(‘classOne classTwo’);**
- **To remove all class attributes for the matching selector**
  - **$(‘p’).removeClass();**
- **.toggleClass() alternates adding or removing a class based on the current presence or absence of the class.**
  - **$(‘#targetDiv’).toggleClass('highlight');**

Handling Events & Animations

Event attachment technique

- **Most Browsers**
  - **myButton.addEventListener(‘click’,function( ) { },false);**

- **Internet Explorer**
  - **myButton.attachEvent(‘onclick’,function( ) { });**

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 common across all browsers.**
- **jQuery event model is simple to use and provides a compact syntax**

jQuery Events

- click()
- blur()
- focus()
- dblclick()
- mousedown()
- mouseup()
- mouseover()
- keydown()
- keypress()

Handling Click Events

- **.click(handler(eventObject)) is used to listen for a click event or trigger a click event on an element**
  - **$(‘#submitButton’).click(function() { alert(‘Clicked’) });**
- **Raising a click event from with in another function**
  - **$(‘#otherID’).click(function() { $('#myID').click( ); }); - this would fire when the element otherID was clicked and raise the click event for myID**

bind() and unbind() Method

- **bind() method is used to bind events dynamically**
- **.bind(eventType,handler(eventObject)) attaches a handler to an event for the selected element(s).**
  - $(‘#submitButton’).bind('click', function() { //handle event });
  - .click() is the same as .bind('click')
- **.unbind(event) is used to remove handler previously bound to an element.**
  - $(‘#submitButton’).unbind();
  - $(‘#submitButton’).unbind('click'); - unbind specific event
- **bind() allows multiple events to be bound to one or more elements**
  - $(‘#targetDiv’).bind('mouseenter mouseleave', function() { //handle event });

live() and delegate() functions

- live() and delegate() allow new elements added into the DOM to automatically be attached to an event.
- live() method allows binding of event handlers to elements that match a selector, including future elements. Events bubble up to the document object
- delegate() method is a replacement for live() in jQuery 1.4. It attaches an event handler directly to the selector context

Using live() and die()

- Event handlers can be set using live()
- The document object handles the event by default
- It works even when new objects were added in to the DOM
  - $('.someclass').live('click',somefunction) – Any element added with .someclass will have the click event
- Stop live event handling using die()
  - $('.someclass').die('click',somefunction)

Using delegate() and undelegate()

- **Newer version of live() added in jQuery 1.4**
- **The context object handles the event by default rather than the document object.**
- **Works even when new objects are added into the DOM.**
  - **$('#targetDiv').delegate('div','click',somefunction);**
- **Stop delegate event handling using undelegate()**

Hover Events

- **Hover events can be handled using hover()**
  - **$(selector).hover(handlerIn,handlerOut)**
  - **handlerIn is equivalent to mouseenter and handlerOut is equivalent to mouseleave**
- **Another option is $(selector).hover(handlerInOut)**
- **Fires the same handler for mousenter and mouseleave events**

Showing and Hiding Elements

- **To set a duration and a callback function**
  - show(duration, callback)
  - duration is the amount of time taken (in milliseconds), and callback is a callback function jQuery will call when the transition is complete.
- **The corresponding version of hide( )**
  - hide(duration, callback)
- **To toggle an element from visible to invisible or the other way around with a specific speed and a callback function, use this form of toggle( )**
  - toggle(duration, callback)

jQuery Sliding Effects

- **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.**

jQuery Fading Effects

- **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.**

Creating Custom Animation

- **Custom animation can be created in jQuery with the animate() function**
  - **animate(params, duration, callback)**
  - **params contains the properties of the object you're animating, such as CSS properties, duration is the optional time in milliseconds that the animation should take and callback is an optional callback function.**

Working with Ajax & Plug-ins

jQuery Ajax features

- **Allows part of a page updated**
- **Cross-Browser support**
- **Simple API**
- **GET and POST supported**
- **Load JSON,XML, HTML …**

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 data**
  - **$.ajax() : Provides core functionality**
- **jQuery Ajax functions works with REST APIs, Webservices and more**

Loading HTML content from server

- **$(selector).load(url,data,callback) allows HTML content to be loaded from a server and added into DOM object.**
  - **$("#targetDiv").load('GetContents.html');**
- **A selector can be added after the URL to filter the content that is returned from the calling load().**
  - **$("#targetDiv").load('GetContents.html #Main');**
- **Data can be passed to the server using load(url,data)**
  - **$('#targetDiv').load('Add.aspx',{firstNumber:5,secondNumber:10})**
- **load () can be passed a callback function**

```
$(‘#targetDiv’).load(‘Notfound.html’, function (res,status,xhr) {
        If (status == “errror”) { alert(xhr.statusText); }
    });
```

Using get(), getJSON() & post()

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

```
$.get('GetContents.html',function(data){
                $('#targetDiv').html(data);
     },'html');
```

  - **datatype can be html, xml, json**

- **$.getJSON(url,data,callback) can retrieve data from a server.**

```
$.getJSON('GetContents.aspx,{id:5},function(data){
                $('#targetDiv').html(data);
     });
```

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

Using ajax() function

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

```
$.ajax({
 url: “employee.asmx/GetEmployees",
 data : null,
 contentType: "application/json; charset=utf-8",
 datatype: ‘json’,
 success: function(data,status,xhr){
  //Perform success operation
 },
error: function(xhr,status,error) {
    //show error details
}
});
```