What is jQuery?

- ❑ jQuery is a fast, small, and feature-rich JavaScript library.
- ❑ Provides easy-to-use API that works across a multitude of browsers.
- ❑ Simplifies JavaScript programming
- ❑ It's tag line is - write less, do more
- ❑ Contains the following features:
    - ✓ HTML/DOM manipulation
    - ✓ CSS manipulation
    - ✓ HTML event methods
    - ✓ Effects and Animations
    - ✓ AJAX
    - ✓ Utilities
- ❑ jQuery has plugins for almost any task out there

Who uses jQuery?

- ❑ Microsoft
- ❑ IBM
- ❑ Netflix
- ❑ WordPress
- ❑ Many more….

More than 50% of websites use jQuery.

Approximately 88% of websites that use JavaScript library, opt for jQuery.

Why to use jQuery?

- ❑ It is lightweight
- ❑ It has great documentation
- ❑ Huge following – Microsoft not just uses, it contributes
- ❑ Shallow learning curve
- ❑ Easy to use – remember its tag line : write less, do more

Downloading jQuery

- There are two versions of jQuery available for downloading from jquery.com
- Production version - this is for your live website because it has been minified and compressed
- Development version - this is for testing and development (uncompressed and readable code)
- The jQuery library is a single JavaScript file, and you reference it with the HTML <script> tag.

```
<head>
        <script src="jquery-3.2.1.min.js"></script>
</head>
```

Using CDN

- If you don't want to download and host jQuery yourself, you can include it from a CDN (Content Delivery Network).
- Both Google and Microsoft host jQuery.

```
<head>
  <script src="http://ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js">
  </script>
</head>
```

```
<head>
  <script src="http://ajax.aspnetcdn.com/ajax/jQuery/jquery-3.2.1.min.js">
  </script>
</head>
```

jQuery Syntax

The jQuery syntax is tailor made for **selecting** HTML elements and performing some **action** on the element(s).

$(*selector*).*action*()

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

$("p").hide() - hides all <p> elements.
$(".test").hide() - hides all elements with class="test".
$("#test").hide() - hides the element with id="test".

Document Ready Event

```
$(document).ready(function() {
   // jQuery methods go here...
});

Or

$(function(){
  // jQuery methods go here...
});
```

- ❑ This is to prevent any jQuery code from running before the document is finished loading (is ready).
- ❑ It is good practice to wait for the document to be fully loaded and ready before working with it.

$ vs $()

- When a method is called on a jQuery object, it uses $()
- When a jQuery method is not acting on a selection, it is called using $

```
$("#p1").text("Welcome")       //  jQuery Object Method
$.get("list.aspx",{},show)     //  jQuery Method
```

jQuery Selectors

- jQuery selectors allow you to select and manipulate HTML element(s).
- jQuery selectors are used to "find" (or select) HTML elements based on their id, classes, types, attributes, values of attributes and much more.
- It's based on the existing CSS Selectors, and in addition, it has some own custom selectors.
- All selectors in jQuery start with the dollar sign and parentheses: $().

Element Selector

The jQuery element selector selects elements based on the element name.

```
$("div").hide()

$(“text[required]").val()
```

#Id Selector

- #id selector uses the id attribute of an HTML tag to find the specific element.
- An id should be unique within a page, so you should use the #id selector when you want to find a single, unique element.

```
$(“#output").hide()

$(“#password").val()
```

Class Selector

- Class selector finds elements with a specific CSS class.
- To find elements with a specific class, write a period character, followed by the name of the class.

```
$(“.header").hide()
```

Compound Selector

- It is possible to combine multiple selectors in one.

```
// selects span item with price class in an element with  id details

$("#details span.price")
```

Comma-separated Selector

- It is possible to select elements that match any of the given multiple selectors by separating selectors by comma

```
// selects span items with price class  and div items with discount class

$("span.price,  div.discount")
```

Pseudo-Selectors

- It is possible to select elements that a particular criteria based on the given pseudo selector like even, odd, visible, first etc.
- Pseudo selectors can be used to select different types of form elements

```
// selects all odd tr elements
$("tr :odd")

//  select first div element
$("div :first")

// select all submit buttons in div
$("div :submit")
```

Examples for Selectors

$("*")	Selects all elements
$(this)	Selects the current HTML element
$("p.intro")	Selects all <p> elements with class="intro"
$("p:first")	Selects the first <p> element
$("ul li:first")	Selects the first <li> element of the first <ul>
$("ul li:first-child")	Selects the first <li> element of every <ul>
$("[href]")	Selects all elements with an href attribute
$("a[target='_blank']")	Selects all <a> elements with a target attribute value equal to "_blank"
$(":button")	Selects all <button> elements and <input> elements of type="button"
$("tr:even")	Selects all even <tr> elements
$("h1,div,p")	All <h1>, <div> and <p> elements.

Method Chaining

- ❑ Technique called chaining allows us to run multiple jQuery commands, one after the other, on the same element(s).
- ❑ This way, browsers do not have to find the same element(s) more than once.
- ❑ To chain an action, you simply append the action to the previous action.
- ❑ The following example chains together the css(), slideUp(), and slideDown() methods.
- ❑ The "p1" element first changes to red, then it slides up, and then it slides down.

```
$("#p1").css("color","red").slideUp(2000).slideDown(2000);
```

jQuery Object

- In many cases what jQuery returns is a jQuery object.
- Especially when you are selecting elements using selectors, what you get is a jQuery object and not an array.
- It is array-like and supports usage of brackets [ ] with object
- It is possible to convert DOM object to jQuery object by enclosing DOM object in parentheses ().
- jQuery objects are not LIVE. It means once they are created they remain the same and do not reflect changes to document.

```
var firstHeadingElement = $( "h1" )[ 0 ];
or
var firstHeadingElement = $( "h1" ).get(0);
```

```
var price = document.getElementById( "price" );
$( price).html( "<h2>0</h2>");
```

Event Handling

- jQuery provides cross-browser event model
- jQuery event model is easy to use
- Need not use addEventListener (or attachEvent) of JavaScript
- The event object provides information about event.
- Every event handler function also get access to element that causes event using this reference.

jQuery Events

- All the different visitor's actions that a web page can respond to are called events.
- Moving a mouse over an element, selecting a radio button, clicking on a button are some of the examples for events.
- Most DOM events have an equivalent jQuery method.

Mouse Events	Keyboard Events	Form Events	Document/Window Events
click	keypress	submit	load
dblclick	keydown	change	resize
mouseenter	keyup	focus	scroll
mouseleave		blur	unload

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

Common Events

- click()
- dblclick()
- mouseenter()
- mouseleave()
- mousedown()
- hover()
- focus()
- blur()

on() and off()

- The **on**() method attaches one or more event handlers for the selected elements.
- The **off**() method removes an event handler.

```
.on( events [, selector ] [, data ], handler )
.off( events [, selector ] [, handler ] )
```

```
$( "#chk" ).on( “mouseenter  mouseleave",
            function() { console.log( event.type); }
);
```

```
$("p").on({
  mouseenter: function(){
    $(this).css("background-color", “blue");
  },
  mouseleave: function(){
    $(this).css("background-color", “red");
  }
});
```

Event Delegation

- Event delegation allows us to attach a single event listener, to a parent element, that will fire for all descendants matching a selector, whether those descendants exist now or are added in the future.
- Event delegation refers to the process of using event propagation (bubbling) to handle events at a higher level in the DOM than the element on which the event originated.
- Events bubble up in the DOM tree. So when an event occurs in the child element, it is bubbled (propagated) to parent and then to its parent and so on.
- We can use **trigger**() method to manually trigger an event without any human interaction.

```
<ul id="list">
  <li>Item #1</li>
  <li>Item #2</li>
  <li>Item #3</li>
</ul>

$( "#list" ).on( "click", "li", function( event ) {
        var elem = $( this );
        console.log( elem.text() );
});
```

Event Object

- ❑ jQuery's event system normalizes the event object according to W3C standards.
- ❑ The event object is guaranteed to be passed to the event handler.
- ❑ Most properties from the original event are copied over and normalized to the new event object.

Properties
✓ target
✓ relatedTarget
✓ pageX
✓ pageY
✓ which
✓ metaKey
✓ data
✓ currentTarget
✓ result
✓ type

Event Object - Methods

The following are methods of **event** object.

stopPropagation()
Prevents the event from bubbling up the DOM tree, preventing any parent handlers from being notified of the event.

preventDefault()
If this method is called, the default action of the event will not be triggered.

isPropagationStopped()
Returns whether event.stopPropagation() was ever called on this event object.

isDefaultPrevented()
Returns whether event.preventDefault() was ever called on this event object.

jQuery Effects

- $(*selector*).hide(*speed,callback*)
- $(*selector*).show(*speed,callback*)
- $(*selector*).toggle(*speed,callback*)
- $(*selector*).fadeIn(*speed,callback*)
- $(*selector*).fadeOut(*speed,callback*)
- $(*selector*).fadeToggle(*speed,callback*)
- $(*selector*).fadeTo(*speed,opacity,callback*)
- $(*selector*).slideDown(*speed,callback*)
- $(*selector*).slideUp(*speed,callback*)
- $(*selector*).slideToggle(*speed,callback*)
- $(*selector*).animate({*params*},*speed,callback*)
- $(*selector*).stop(*stopAll,goToEnd*);

DOM Manipulation

text()
Sets or returns the text content of selected elements

html()
Sets or returns the content of selected elements (including HTML markup)

val()
Sets or returns the value of form fields

attr()
Is used to get or set attribute values.

```
$("#test2").html("<b>Hello world!</b>");

$(“#link1").attr(
   {"href" : "http://www.srikanthtechnologies.com",
   "title" : “Srikanth Technologies” }
);
```

DOM Manipulation

append(), appendTo()
Inserts content at the end of the selected elements
prepend(), prependTo()
Inserts content at the beginning of the selected elements
after()
Inserts content after the selected elements
before()
Inserts content before the selected elements
remove()
Removes the selected element (and its child elements)
empty()
Removes the child elements from the selected element

```
$("p").append(“More…“)

$("img").before(“Photo“)

$("#div1").remove()

$("p").remove(“.red ");
```

Manipulating CSS

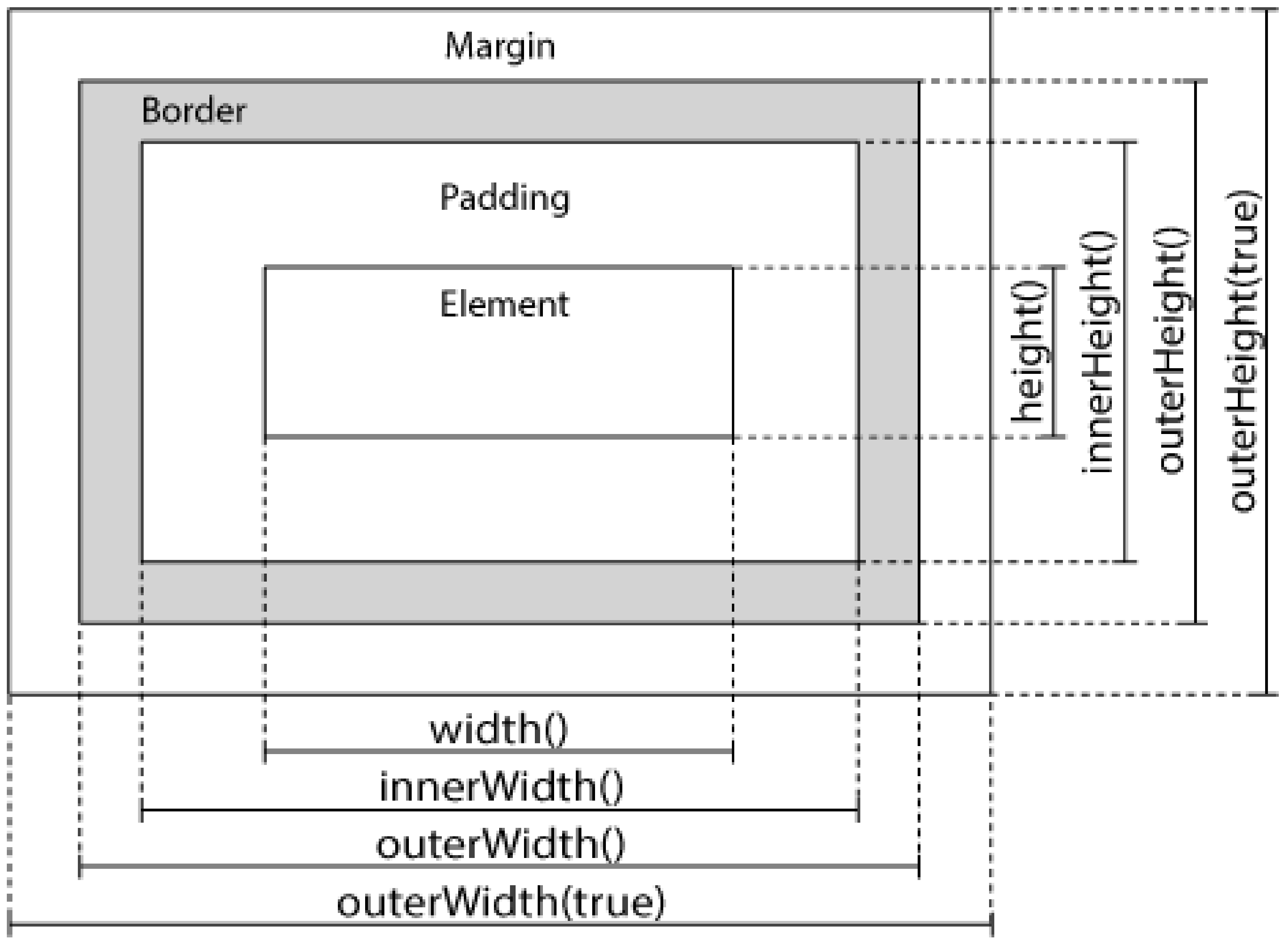
addClass()
Adds one or more classes to the selected elements
removeClass()
Removes one or more classes from the selected elements
toggleClass()
Toggles between adding/removing classes from the selected elements
css()
Sets or returns the style attribute
hasClass()
Checks whether the given class is already set.

```
$("h1,h2,p").addClass("blue white“);  // adds two classes  blue and white

$(“#output").removeClass("blue“)

$("p").css({"background-color":"yellow","font-size":”20pt”});
```

Examples

```
$("divResult").attr (  { colors : 'red',  title : 'Sample'} )

$("Value for Span").appendTo("#span1")
Or
$("#span1").append("Value for Span");

$("div1").warp("<div> Parent </div>");  // wraps  div1 with new parent div

$(".red, .green").remove();

$("div").css( { "color":"red", "font-size" : "20pt"} );
```

Dimension Methods



Dimension Methods

width()
Sets or returns the width of an element (includes NO padding, border, or margin)

height()
Sets or returns the height of an element (includes NO padding, border, or margin)

innerWidth()
Returns the width of an element (includes padding)

innerHeight()
Returns the height of an element (includes padding)

outerWidth()
Returns the width of an element (includes padding and border)

outerHeight()
Returns the height of an element (includes padding and border)

Traversing Methods

parent()
Returns the direct parent element of the selected element

parents()
Returns all ancestor elements of the selected element, all the way up to the document's root element (<html>)

parentsUntil()
Returns all ancestor elements between two given arguments

children()
Returns all direct children of the selected element

find()
Returns descendant elements of the selected element, all the way down to the last descendant.

siblings()
Returns all sibling elements of the selected element

Traversing Methods

next()
Returns the next sibling element of the selected element

nextAll()
Returns all next sibling elements of the selected element

nextUntil()
Returns all next sibling elements between two given arguments

prev(), prevAll() and prevUntil()
Work just like next(), nextAll(), nextUntil() but with reverse functionality: they return previous sibling elements (traverse backwards along sibling elements in the DOM tree, instead of forward)

first()
Returns the first element of the selected elements

last()
Returns the last element of the selected elements

Traversing Methods

eq()

Returns an element with a specific index number of the selected elements

filter()

Lets you specify a criteria. Elements that do not match the criteria are removed from the selection, and those that match will be returned

not()

Returns all elements that do not match the criteria

Traversing Examples

```
$(“#output").parent()
$(“#message").parents("ul“)
$("div").children()
$("div").find("span“)
$("div p").last()
$("h2").siblings("p“)
$("h2").nextAll()
$("p").eq(1)
$("p").filter(".intro“)
```

.each

.each(function)

Function (Integer index, Element element)

A function to execute for each matched element.

Inside the function **this** can be used to refer to element.

```
$('div').each(
    function (index, elem) {
        document.write( index + " =" +  $(elem).text());
    }
);
```

Ajax – load()

$(*selector*).load(*URL,data,callback*)
Loads data from a server and puts the returned data into the selected element.

URL parameter specifies the URL you wish to load.
Data parameter specifies a set of querystring key/value pairs to send along with the request.
callback parameter specifies a callback function to run when the load() method is completed. The callback function can have different parameters:

- ❑ responseTxt - contains the resulting content if the call succeed
- ❑ statusTxt - contains the status of the call
- ❑ xhr - contains the XMLHttpRequest object

```
$(document).ready(function () {
     $("#load").click(function () {
            // $("#quote").load("quote.txt", process);
            $("#quote").load("quote.txt");
         });
      }
);
function process(r, s, x) {
    if (s == “error")
         alert("Sorry! Could not load quote!");
}
```

Ajax – get()

$.get(*URL, callback*)

- **The required URL parameter specifies the URL you wish to request.**
- **The optional callback parameter is the name of a function to be executed if the request succeeds.**
- **First parameter of callback function holds the content of the page requested, and the second holds the status of the request.**

Ajax – post()

$.post(*URL,data,callback*)

- URL parameter specifies the URL you wish to request.
- Data parameter specifies some data to send along with the request.
- Callback parameter is the name of a function to be executed if the request succeeds.

Ajax – getJSON()

$.getJSON(*url,data,success(data,status,xhr))*

- URL parameter specifies the URL you wish to request.
- Data parameter specifies some data to send along with the request.
- Success parameter is the name of a function to be executed if the request succeeds.

Ajax – ajax()

.ajax(url [, settings])

- URL parameter specifies the URL you wish to request.
- Settings is a set of key/value pairs that configure the Ajax request. All settings are optional.
- The following are important keys.

complete	-	Function(jqXHR jqXHR, String textStatus)
data	-	Plain object or string
dataType	-	Default: Intelligent Guess (xml, json, script, or html)
error	-	Function(jqXHR jqXHR, String textStatus, String errorThrown)
method	-	String – default is GET
success	-	Function(Anything data, String textStatus, jqXHR jqXHR)

JQuery UI

- UI is a set of user interface interactions, effects, widgets, and themes built on top of the jQuery JavaScript Library.
- Download jQuery UI from http://jqueryui.com/

DatePicker

- Select a date from a popup or inline calendar.
- The datepicker is tied to a standard form input field.
- Focus on the input (click, or use the tab key) to open an interactive calendar in a small overlay

OPTIONS

- changeMonth
- changeYear
- dateFormat
- defaultDate
- maxDate
- minDate
- onChangeMonthYear
- onClose
- onSelect
- showAnim
- showButtonPanel
- yearRange

METHODS

- destroy
- dialog
- getDate
- hide
- isDisabled
- option
- refresh
- setDate
- show
- widget

Menu

- ❑ **A menu with the default configuration, disabled items and nested menus.**
- ❑ **A list is transformed, adding theming, mouse and keyboard navigation support.**
- ❑ **Try to tab to the menu then use the cursor keys to navigate.**

OPTIONS
disabled
icons
menus
position
role

METHODS
blur
collapse
collapseAll
expand
focus
isFirstItem
isLastItem
next
nextPage
option
previous
previousPage
refresh
select
widget

EVENTS
blur
create
focus
select

Tabs

A single content area with multiple panels, each associated with a header in a list.

OPTIONS	METHODS	EVENTS
active	destroy	activate
collapsible	disable	beforeActivate
disabled	enable	beforeLoad
event	load	create
heightStyle	option	load
hide	refresh	
show	widget	

AutoComplete

- Enables users to quickly find and select from a pre-populated list of values as they type, leveraging searching and filtering.
- Provides suggestions while you type into the field

OPTIONS	METHODS	EVENTS
appendTo	close	change
autoFocus	destroy	close
delay	disable	create
disabled	enable	focus
minLength	option	open
position	search	response
source	widget	search
		select

Accordion

- Displays collapsible content panels for presenting information in a limited amount of space.
- Click headers to expand/collapse content that is broken into logical sections, much like tabs.
- The underlying HTML markup is a series of headers (H3 tags) and content divs so the content is usable without JavaScript

OPTIONS
active
animate
collapsible
disabled
event
header
heightStyle
icons

METHODS
destroy
disable
enable
option
refresh
widget

EVENTS
activate
beforeActivate
create

ProgressBar

Display status of a determinate or indeterminate process.

OPTIONS	METHODS	EVENTS
disabled	destroy	change
max	disable	complete
value	enable	create
	option	
	value	
	widget	