SCS1620 User Interface Technologies

Unit III - JavaScript and jQuery

Introduction - Core features - Data types and Variables - Operators, Expressions and Statements - Functions & Scope - Objects - Array, Date and Math related Objects - Document Object Model - Event Handling – Browser Object Model - Windows and Documents - Form handling and validations.Object-Oriented Techniques in JavaScript - Classes – Constructors and Prototyping (Sub classes and Super classes) – JSON – Introduction to AJAX.Introduction – jQuery Selectors – jQuery HTML - Animations – Effects – Event Handling – DOM – jQuery DOM Traversing, DOM Manipulation – jQuery AJAX.

Introduction

JavaScript is the programming language of HTML and the Web.

Why Javascript?

1. HTML to define the content of web pages
2. CSS to specify the layout of web pages
3. JavaScript to program the behavior of web pages

```
<!DOCTYPE html>
<html>
<body>
<h2>My First JavaScript</h2>
<button type="button"
onclick="document.getElementById('demo').innerHTML = Date()">
Click me to display Date and Time.</button>
<p id="demo"></p>
</body>
</html>
```

```
<!DOCTYPE html>
<html>
<body>
<h2>What Can JavaScript Do?</h2>
<p>JavaScript can change HTML attributes.</p>
<p>In this case JavaScript changes the src (source) attribute of an image.</p>
<button onclick="document.getElementById('myImage').src='pic_bulbon.gif'">Turn on the
light</button>
<img id="myImage" src="pic_bulboff.gif" style="width:100px">
<button onclick="document.getElementById('myImage').src='pic_bulboff.gif'">Turn off the
light</button>
</body>
</html>
```

```
<!DOCTYPE html>
<html>
<head>
<script>
function myFunction() {
   document.getElementById("demo2").innerHTML = "Paragraph changed using
myFunction.";
}
</script>
</head>
<body>
<h2>What Can JavaScript Do?</h2>
<p id="demo">JavaScript can change the style of an HTML element.</p>
<p id="demo1">Will be blocked by javascript</p>
<h1>A Web Page</h1>
<p id="demo2">A Paragraph</p>
<h2>External JavaScript</h2>
<p id="demo3">A Paragraph.</p>
<button type="button" onclick="myFunction()">External JavaScript</button>
<p>(myFunction is stored in an external file called "myScript.js")</p>
<button type="button"
onclick="document.getElementById('demo').style.fontSize='35px'">Click Me!</button>
<button type="button"
onclick="document.getElementById('demo1').style.display='none'">Click Me!</button>
<button type="button" onclick="myFunction()">Click Me!</button>
<script src="myScript.js"></script>
</body>
</html>
```

Placing scripts in external files has some advantages:

- It separates HTML and code
- It makes HTML and JavaScript easier to read and maintain
- Cached JavaScript files can speed up page loads
- To add several script files to one page - use several script tags:

Example

```
<script src="myScript1.js"></script>
<script src="myScript2.js"></script>
```

Display Example

```
<!DOCTYPE html>
<html>
<head>
<script>
function myFunction() {
        document.getElementById("demo1").innerHTML = 5 + 6;
        document.write(50 + 60);
```

```
        window.alert(5 + 6);
        console.log(5 + 6);
}
</script>

</head>
<body onload="myFunction()">
<p id="demo1"></p>
<p id="demo2"></p>
<button type="button" onclick="window.alert(5 + 6)">Add 5+6</button>
</body>
</html>
```

Comments

Single Line

```
// This is single line comment
/* This is
Multiple line comment */
```

Variables

Variables are declared using the keyword “var”.

Example

```
var x = 10;
var name = ‘test’;
var dob = ’02-10-2005’;
var y = 10.15;
```

Operators

Arithmetic Operators		**Assignment Operators**		
Operator	**Description**	**Operator**	**Example**	**Same As**
+	Addition	=	x = y	x = y
-	Subtraction	+=	x += y	x = x + y
*	Multiplication	-=	x -= y	x = x - y
/	Division	*=	x *= y	x = x * y
%	Modulus (Remainder)	/=	x /= y	x = x / y
++	Increment	%=	x %= y	x = x % y
--	Decrement			

JavaScript Data Types

JavaScript variables can hold many data types: numbers, strings, objects and more.

```
var length = 11;                                  // Number
var dept = "CSE";                                 // String
var x = {school:"computing", dept:"CSE"};         // Object
var numStr = 11+“CSE”;                            // 11CSE
var strNum = “CSE”+11;                            // CSE11
var numNumStr = 11 + 11 + ”CSE”                   // 22CSE
var strNumNum = “CSE” + 11 + 11                   // CSE1111
// Variables are dynamic
var x;
var x=11;
var x=”CSE”;
var x=’CSE’;
var x=”Single ‘quotes’”;
var x=’Double “Quotes”’;
var x=123e4                                       // 1230000
var y=123e-4                                      // 0.0123
var x=5;
var y=5;
var z=6;
(x==y)                                            // returns true
(x==z)                                            // returns false
var array = [“CSE”, “IT”];
typeof “”                                         // returns String
typeof “CSE”                                      // returns String
typeof “School of Computing”                      // returns String
typeof 0                                          // returns number
typeof 314                                        // returns number
typeof 3.14                                       // returns number
typeof (3+4)                                      // returns number
typeof true                                       // returns Boolean
typeof false                                      // returns Boolean
typeof function myFunction() {}                   // returns Function
```

JavaScript Functions

Example 1:

```
<!DOCTYPE html>
<html>
<body>
<h2>JavaScript Functions</h2>
<p>This example calls a function which performs a calculation, and returns the result to the
p tag of ids result and ctof:</p>
<p id="result"></p>
<p id="ctof"></p>
<p id="objectDemo"></p>
<script>
```

```
document.getElementById("result").innerHTML = calcMultiply(4, 3);
document.getElementById("ctof").innerHTML = "The temperature is " + toCelsius(100) + "
Celsius";
function calcMultiply(n1, n2) {
   return n1 * n2;
}
function toCelsius(fahrenheit) {
   return (5/9) * (fahrenheit-32);
}
var course = {
   school        : "SoC",
   dept          : "CSE",
   id      : 11
};
document.getElementById("objectDemo").innerHTML =course.school + " " + course.dept;
</script>
</body>
</html>
```

HTML Events

HTML events are "actions" that are performed on HTML elements.
When JavaScript is used in HTML pages, JavaScript can "react" on these events.

Events Example

- An HTML web page has finished loading
- An HTML input field was changed
- An HTML button was clicked

Example

```
<!DOCTYPE html>
<html>
<script>
function displayDate() {
   document.getElementById("eventDemo2").innerHTML = Date();
}
</script>
<body>
<button onclick="document.getElementById('eventDemo1').innerHTML=Date()">Current
Time?</button>
<p id="eventDemo1"></p>
<button onclick="this.innerHTML=Date()">Click Here to get Current Time</button>
<br><br>
<button onclick="displayDate()">Current Time using Javascript Function</button>
<p id="eventDemo2"></p>
</body>
</html>
```

Common HTML Events

Event	Description
onchange	An HTML element has been changed
onclick	User clicks an HTML element
onmouseover	Moves the mouse over an HTML element
onmouseout	Moves the mouse away from an HTML element
onkeydown	Push a keyboard key
onload	Browser has finished loading the page

Strings

```
<!DOCTYPE html>
<html>
<body>
<h2>JavaScript String Properties</h2>
<p>The length property returns the length of a string:</p>
<p id="stringLength"></p>
<p id="stringObject"></p>
<p id="stringFind1"></p>
<p id="stringFind2"></p>
<p id="stringFind3"></p>
<p id="stringSearch"></p>
<pre> Difference between indexOf() and search()
Both Functions accept the same arguments and return the same value.
The search() method cannot take a second start position argument.
The indexOf() method cannot take powerful search values (regular expressions).
</pre>
<p>String extracted using Slice function</p>
<p> With positive values: </p><p id="stringSlice1"></p>
<p> with negative values: </p><p id="stringSlice2"></p>
<p> with positive value: </p><p id="stringSlice3"></p>
<p> with negative value: </p><p id="stringSlice4"></p>

<p> Using replace function</p>
<p> Standard: <p id="stringReplace1"></p>
<p> Case Sensitive: <p id="stringReplace2"></p>
<p> Case Ignore: <p id="stringReplace3"></p>
<p> Replace All Occurrence: <p id="stringReplace4"></p>

<p> Case Conversion </p>
<p> To Upper Case: </p><p id="stringCase1"></p>
<p> To Lower Case: </p><p id="stringCase2"></p>

<p> Concatenation </p>
<p> Using Variables: </p><p id="stringConcat1"></p>
<p> Direct String: </p><p id="stringConcat2"></p>

<p> Extract String Characters </p>
<p> Using charAt: </p><p id="stringExtract1"></p>
```

```
<p> Using charCodeAt: </p><p id="stringExtract2"></p>

<p> Convert String to Array </p>
<p> Using comma separator: </p><p id="stringArray1">
<p> Using space separator: </p><p id="stringArray2">
<p> Using semicolon separator: </p><p id="stringArray3">
<script>
var txt = "Department of Computer Science and Engineering";
document.getElementById("stringLength").innerHTML = txt.length;

var x = "CSE";             // x is a string
var y = new String("CSE");  // y is an object
document.getElementById("stringObject").innerHTML = typeof x + "<br>" + typeof y;
//alert(x==y); // returns true. Sine CSE == CSE
//alert(x===y); // returns false. Since string != object

var str = "Computer Science and Engineering";
var pos = str.indexOf("Science");
var comm="Science found at : ";
document.getElementById("stringFind1").innerHTML =  comm + pos;

var str = "Computer Science and Engineering Science";
var pos = str.lastIndexOf("Science");
document.getElementById("stringFind2").innerHTML = "Last Science found at : " + pos;

var pos = str.lastIndexOf("Science1");
document.getElementById("stringFind3").innerHTML = "String not found : " + pos;

var pos = str.search("Science");
document.getElementById("stringSearch").innerHTML = "Science found at : " + pos;

var str = "IoT, Python, UIT";
var res = str.slice(5,11);
document.getElementById("stringSlice1").innerHTML = res;

var res = str.slice(-11,-5);
document.getElementById("stringSlice2").innerHTML = res;

var res = str.slice(5);
document.getElementById("stringSlice3").innerHTML = res;

var res = str.slice(-11);
document.getElementById("stringSlice4").innerHTML = res;

var str = "Please visit ECE!";
var n = str.replace("ECE", "CSE");
document.getElementById("stringReplace1").innerHTML=n;
```

```
var str = "Please visit ece!";
var n = str.replace("ECE", "CSE");
document.getElementById("stringReplace2").innerHTML=n;

var str = "Please visit ece!";
var n = str.replace(/ECE/i, "CSE");
document.getElementById("stringReplace3").innerHTML=n;

var str = "Please visit ece and ece!";
var n = str.replace(/ece/g, "CSE");
document.getElementById("stringReplace4").innerHTML=n;

var str1 = "Please visit CSE!";
var str2 = str1.toUpperCase();
document.getElementById("stringCase1").innerHTML=str2;

var str2 = str1.toLowerCase();
document.getElementById("stringCase2").innerHTML=str2;

var str1 = "School";
var str2 = "of Computing";
var str3 = str1.concat(" ", str2);
document.getElementById("stringConcat1").innerHTML=str3;

var str3="School".concat(" ", "of Computing");
document.getElementById("stringConcat2").innerHTML=str3;

var str = "CSE";
var extr = str.charAt(0);
document.getElementById("stringExtract1").innerHTML=extr;

var extr = str.charCodeAt(0);
document.getElementById("stringExtract2").innerHTML=extr;

var str = "a,b,c,d,e,f";
var arr = str.split(",");
document.getElementById("stringArray1").innerHTML = arr[0];

var str = "a b c d e f";
var arr = str.split(" ");
document.getElementById("stringArray2").innerHTML = arr[0];

var str = "a;b;c;d;e;f";
var arr = str.split(";");
document.getElementById("stringArray3").innerHTML = arr[0];
</script>
</body>
</html>
```

Numbers

```
<!DOCTYPE html>
<html>
<body>
<p>The toString() method converts a number to a string.</p>
<p id="numberString"></p>

<p>The toExponential() method returns a string, with the number rounded and written using exponential notation.</p>
<p id="numberExponential"></p>

<p>The toFixed() method rounds a number to a given number of digits.</p>
<p id="numberFixed"></p>

<p>The toPrecision() method returns a string, with a number written with a specified length:</p>
<p id="numberPrecision"></p>

<p>valueOf() returns a number as a number. The valueOf() method is used internally in JavaScript to convert Number objects to primitive values.</p>
<p id="numberValue"></p>

<p>The JavaScript function Number() converts variables to numbers and can also convert a date to a number</p>
<p id="numberNumber"></p>

<p>parseInt() parses a string and returns a whole number. Spaces are allowed. Only the first number is returned</p>
<p id="numberParseint"></p>

<p>parseFloat() parses a string and returns a number. Spaces are allowed. Only the first number is returned</p>
<p id="numberParsefloat"></p>
<p>Number Properties</p>
<p id="numberProperties"></p>
<script>
var x = 123;
document.getElementById("numberString").innerHTML =
    x.toString() + "<br>" + (123).toString() + "<br>" +
   (100 + 23).toString();

var x = 9.656;
document.getElementById("numberExponential").innerHTML =
    x.toExponential() + "<br>" + x.toExponential(2) + "<br>" +
    x.toExponential(4) + "<br>" + x.toExponential(6);
//Output: 9.656e+0, 9.66e+0, 9.6560e+0, 9.656000e+0

document.getElementById("numberFixed").innerHTML =
```

```
    x.toFixed(0) + "<br>" + x.toFixed(2) + "<br>" +
    x.toFixed(4) + "<br>" + x.toFixed(6);
// Output: 10, 9.66, 9.6560, 9.656000

document.getElementById("numberPrecision").innerHTML =
    x.toPrecision() + "<br>" + x.toPrecision(2) + "<br>" +
    x.toPrecision(4) + "<br>" + x.toPrecision(6);
// Output: 9.656, 9.7, 9.656, 9.65600

var x="123";
document.getElementById("numberValue").innerHTML = x.valueOf();

document.getElementById("numberNumber").innerHTML =
    Number(true) + "<br>" + Number(false) + "<br>" +
    Number("  10") + "<br>" + Number("10  ") + "<br>" + Number("10 6") + "<br>" +
Number("CSE") + "<br>" + Number(new Date("2018-03-01"));
// Output: 1, 0, 10, 10, NaN, NaN, 1519862400000

document.getElementById("numberParseint").innerHTML =
    parseInt("10") + "<br>" + parseInt("10.33") + "<br>" +
    parseInt("10 6") + "<br>" + parseInt("10 years") + "<br>" +
    parseInt("years 10");
//Output: 10, 10, 10, 10, NaN

document.getElementById("numberParsefloat").innerHTML =
    parseFloat("10") + "<br>" + parseFloat("10.33") + "<br>" +
    parseFloat("10 6") + "<br>" + parseFloat("10 years") + "<br>" +
    parseFloat("years 10");
//Output: 10, 10.33, 10, 10, NaN

document.getElementById("numberProperties").innerHTML = Number.MAX_VALUE +
"<br>" + Number.MIN_VALUE + "<br>" + Number.NEGATIVE_INFINITY + "<br>" +
Number.POSITIVE_INFINITY + "<br>" + Number.NaN;
//Output: 1.7976931348623157e+308, 5e-324, -Infinity, Infinity, NaN
</script>
</body>
</html>
```

Math Object

Math object is to perform mathematical tasks on numbers.

```
<!DOCTYPE html>
<html>
<body>
<h1> Math Object </h1>
<p>Math.PI returns the ratio of a circle's circumference to its diameter:</p>
<p id="mathpi"></p>
<p>Math.round(x) returns the value of x rounded to its nearest integer</p>
```

```
<p id="mathround"></p>
<p>Math.pow(x, y) returns the value of x to the power of y</p>
<p id="mathpow"></p>
<p>Math.sqrt(x) returns the square root of x</p>
<p id="mathsqrt"></p>
<p>Math.abs(x) returns the absolute (positive) value of x</p>
<p id="mathabsolute"></p>
<p>Math.ceil(x) returns the value of x rounded up to its nearest integer. Math.floor(x) returns the value of x rounded down to its nearest integer</p>
<p id="mathceilfloor"></p>
<p>Math.sin(x) returns the sine (a value between -1 and 1) of the angle x (given in radians). To convert degrees to radians: <br>
Angle in radians = Angle in degrees x PI / 180. <br> Similarly Math.cos(x), Math.tan(x), Math.asin(x), Math.acos(x), Math.atan(x) math objects are available in java script.</p>
<p id="mathtrig"></p>
<p>Math.min() and Math.max() can be used to find the lowest or highest value in a list of arguments</p>
<p id="mathminmax"></p>
<p>Math.random() returns a random number between [0,1)</p>
<p id="mathrandom"></p>
<p>8  mathematical constants that can be accessed with Math object</p>
<p id="mathconstants"></p>
<script>
document.getElementById("mathpi").innerHTML = Math.PI;
document.getElementById("mathround").innerHTML = Math.round(4.7) + "<br>" +
Math.round(4.4); // Output: 5, 4
document.getElementById("mathpow").innerHTML = Math.pow(6,2);
document.getElementById("mathsqrt").innerHTML = Math.sqrt(36);
document.getElementById("mathabsolute").innerHTML = Math.abs(-6.6);
document.getElementById("mathceilfloor").innerHTML = Math.ceil(4.4) + "<br>" +
Math.floor(4.7); // Output: 5, 4
document.getElementById("mathtrig").innerHTML = Math.sin(45 * Math.PI / 180) + "<br>" +
Math.cos(45 * Math.PI / 180) + "<br>" + Math.tan(45 * Math.PI / 180)
document.getElementById("mathminmax").innerHTML = Math.max(0, 150, 30, 20, -8, -200)
+ "<br>" + Math.min(0, 150, 30, 20, -8, -200);
document.getElementById("mathrandom").innerHTML = Math.random();
document.getElementById("mathconstants").innerHTML =
"<p><b>Math.E:</b> " + Math.E + "</p>" +
"<p><b>Math.PI:</b> " + Math.PI + "</p>" +
"<p><b>Math.SQRT2:</b> " + Math.SQRT2 + "</p>" +
"<p><b>Math.SQRT1/2:</b> " + Math.SQRT1_2 + "</p>" +
"<p><b>Math.LN2:</b> " + Math.LN2 + "</p>" +
"<p><b>Math.LN10:</b> " + Math.LN10 + "</p>" +
"<p><b>Math.LOG2E:</b> " + Math.LOG2E + "</p>" +
"<p><b>Math.Log10E:</b> " + Math.LOG10E + "</p>";
</script>
</body>
</html>
```

Date Objects

The Date object is available to work with dates (years, months, days, hours, minutes, seconds, and milliseconds).

A JavaScript date can be written as a string:

Tue Mar 06 2018 05:05:49 GMT+0530 (India Standard Time)

or as a number:

1520292949017

Dates written as numbers, specifiy the number of milliseconds since January 1, 1970, 00:00:00.

```
<!DOCTYPE html>
<html>
<body>
<pre>new Date() constructor, creates a new date object with the current date and time. There are 4 ways to initiate a date.
new Date()
new Date(milliseconds)
new Date(dateString)
new Date(year, month, day, hours, minutes, seconds, milliseconds)</pre>
<p id="date1"></p>
<p id="date2"></p>
<p id="date3"></p>
<p id="date4"></p>
<p> toString, toUTCString, toDateString </p>
<p id="date5"></p>
<p id="date6"></p>
<p id="date7"></p>
<p id="date8"></p>
<p>Date.parse(string) returns milliseconds.</p>
<p>The return value of Date.parse is used to convert the string to a date object</p>
<p id="date9"></p>
<p id="date10"></p>
<script>
var d = new Date();
document.getElementById("date1").innerHTML = d;
var d = new Date(100000000000);
document.getElementById("date2").innerHTML = d;
var d = new Date("March 06, 2018 11:13:00");
document.getElementById("date3").innerHTML = d;
var d = new Date(2018, 2, 6, 11, 33, 30, 0);
document.getElementById("date4").innerHTML = d;

document.getElementById("date5").innerHTML = d.toString() + "<br>" + d.toUTCString() + "<br>" + d.toDateString();
```

```
document.getElementById("date6").innerHTML = d.getDate() + "<br>" + d.getDay() + "<br>"
+ d.getFullYear() + "<br>" + d.getMonth() + "<br>" + d.getTime() + "<br>" + d.getHours() +
"<br>" + d.getMinutes() + "<br>" + d.getSeconds() + "<br>" + d.getMilliseconds();

d.setDate(2); //Set the day as a number (1-31)
d.setFullYear(2018); //Set the year (optionally month and day)
d.setMonth(4); //Set the month (0-11)
d.setTime(1520316210000); //Set the time (milliseconds since January 1, 1970)
d.setHours(3); //Set the hour (0-23)
d.setMinutes(30);  //Set the minutes (0-59)
d.setSeconds(55); //Set the milliseconds (0-59)
d.setMilliseconds(989); //Set the milliseconds (0-999)

document.getElementById("date8").innerHTML = d.getDate() + "<br>" + d.getDay() + "<br>"
+ d.getFullYear() + "<br>" + d.getMonth() + "<br>" + d.getTime() + "<br>" + d.getHours() +
"<br>" + d.getMinutes() + "<br>" + d.getSeconds() + "<br>" + d.getMilliseconds();

var msec = Date.parse("March 6, 2018");
var d = new Date(msec);
document.getElementById("date9").innerHTML = d;

var today, someday, text;
today = new Date();
someday = new Date();
someday.setFullYear(2018, 31, 12);

if (someday > today) {
   text = "Today is before December 31, 2018.";
} else {
   text = "Today is after December 31, 2018.";
}
document.getElementById("date10").innerHTML = text;
</script>
</body>
</html>
```

Arrays

```
<!DOCTYPE html>
<html>
<body>

<h2>JavaScript Arrays</h2>

<p id="array1"></p>
<p id="array2"></p>
<p id="array3"></p>
<h2> Arrays as objects </h2>
<p id="array4"></p>
```

```
<p id="array5"></p>
<p id="array6"></p>
<p id="array7"></p>
<p id="array8"></p>
<p id="array9"></p>
<p id="array10"></p>
<p id="array11"></p>
<p id="array12"></p>
<p id="array13"></p>
<p id="array14"></p>
<p id="array15"></p>
<p id="array16"></p>
<p id="array17"></p>
<p id="array18"></p>
<script>
var soc1 = ["CSE", "IT", "CS"]; // Advisable
var soc2 = new Array("CSE", "IT", "CS"); // Avoid. Complicates code. Produces unexpected results.
document.getElementById("array1").innerHTML = soc1;
document.getElementById("array2").innerHTML = soc2;
document.getElementById("array3").innerHTML = soc2[2];
var soc3 = {dept1: "CSE", dept2: "IT", dept3: "CS"};
document.getElementById("array4").innerHTML = soc3.dept1 + " " + soc3.dept2 + " " +soc3.dept3;
document.getElementById("array5").innerHTML = soc2.length;
document.getElementById("array6").innerHTML = soc2.sort();

soc2.push("M.E. CSE")
var alen = soc2.length;
var text = "<ul>";
for (i = 0; i<alen; i++) {
        text += "<li>" + soc2[i] + "</li>";
}
text += "</ul>";
document.getElementById("array7").innerHTML = text;
document.getElementById("array8").innerHTML = soc1 instanceof Array;
document.getElementById("array9").innerHTML = typeof soc2 + " " + Array.isArray(soc1);

document.getElementById("array10").innerHTML = soc2.toString();
document.getElementById("array11").innerHTML = soc2.join(" + ");
document.getElementById("array12").innerHTML = soc2.pop();
document.getElementById("array13").innerHTML = soc2.shift();
document.getElementById("array14").innerHTML = soc2.unshift("IoT");
delete soc1[4];
document.getElementById("array15").innerHTML = soc1[4] ;

ele1 = ["UIT", "IoT"];
ele2 = ["Python", "UNIX"];
```

```
ele3 = ["ST", "SA"];
var eles = ele1.concat(ele2);
var eles2 = ele1.concat(ele2, ele3);
document.getElementById("array16").innerHTML = eles;
document.getElementById("array17").innerHTML = eles2;
document.getElementById("array18").innerHTML = eles2.reverse();

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

Form Validation

```
<!DOCTYPE html>
<html lang="en"><head>
<meta charset="utf-8">
<title>Copper UI - Registration form</title>
<meta name="keywords" content="Copper UI, JavaScript Form Validation, registration form" />
<meta name="description" content="This document is an example of JavaScript Form Validation using a registration form. " />
<link rel='stylesheet' href='js-validation.css' type='text/css' />
<script src="registration-form-validation.js"></script>
</head>
<body onload="document.registration.userid.focus();">
<h1>Registration Form </br> COPPER UI GROUP - I</h1>
<p>Use tab keys to move from one input field to the next</p>
<form name='registration' onSubmit="return formValidation();">
<ul>
<li><label for="userid">User id:</label></li>
<li><input type="text" name="userid" size="12" /></li>
<li><label for="passid">Password:</label></li>
<li><input type="password" name="passid" size="12" /></li>
<li><label for="username">Name:</label></li>
<li><input type="text" name="username" size="50" /></li>
<li><label for="address">Address:</label></li>
<li><input type="text" name="address" size="50" /></li>
<li><label for="country">Country:</label></li>
<li><select name="country">
<option selected="" value="Default">(Please select a country)</option>
<option value="AF">Australia</option>
<option value="AL">Canada</option>
<option value="DZ">India</option>
<option value="AS">Russia</option>
<option value="AD">USA</option>
</select></li>
<li><label for="zip">ZIP Code:</label></li>
<li><input type="text" name="zip" /></li>
<li><label for="email">Email:</label></li>
```

```
<li><input type="text" name="email" size="50" /></li>
<li><label id="gender">Sex:</label></li>
<li><input type="radio" name="usex" value="Male"/><span>Male</span></li>
<li><input type="radio" name="usex" value="Female" /><span>Female</span></li>
<li><label>Language:</label></li>
<li><input type="checkbox" name="en" value="en" checked /><span>English</span></li>
<li><input type="checkbox" name="en" value="noen" /><span>Non English</span></li>
<li><label for="desc">About:</label></li>
<li><textarea name="desc" id="desc"></textarea></li>
<li><input type="submit" name="submit" value="Submit" /></li>
</ul>
</form>
</body>
</html>

function formValidation() {
	var usex = document.registration.usex;
	var uid = document.registration.userid;
	var passid = document.registration.passid;
	var uname = document.registration.username;
	var uadd = document.registration.address;
	var ucountry = document.registration.country;
	var uzip = document.registration.zip;
	var uemail = document.registration.email;
	if(userValidation(uid,5,12) && passwordValidation(passid,7,12) && allLetter(uname)
&& alphanumeric(uadd) && countrySelect(ucountry) && allNumeric(uzip) &&
validateEmail(uemail) && validateSex(usex)) {
		alert("Form Successfully Validated!!!");
		return true;
	}
	else
		return false;
}
function userValidation(uid,mx,my) {
	var uid_len = uid.value.length;
	console.log(uid_len);
	if (uid_len == 0 || uid_len >= my || uid_len < mx) {
		alert("User Id should not be empty / length be between "+mx+" to "+my);
		uid.focus();
		return false;
	}
	console.log("uv - true");
	return true;
}
function passwordValidation(passid,mx,my) {
	var passid_len = passid.value.length;
	if (passid_len == 0 ||passid_len >= my || passid_len < mx)  {
		alert("Password should not be empty / length be between "+mx+" to "+my);
```

```
                passid.focus();
                return false;
        }
                return true;
}
function allLetter(uname) {
        var letters = /^[A-Za-z]+$/;
        if(uname.value.match(letters))
                return true;
        else {
                alert('Username must have alphabet characters only');
                uname.focus();
                return false;
        }
}
function alphanumeric(uadd) {
        if(uadd.value != null)
                return true;
        else {
                alert('User address should not be null');
                uadd.focus();
                return false;
                }
}
function countrySelect(ucountry) {
        if(ucountry.value == "Default") {
                alert('Select your country from the list');
                ucountry.focus();
                return false;
        }
        else
                return true;
}
function allNumeric(uzip) {
var numbers = /^[0-9]+$/;
if(uzip.value.match(numbers))
        return true;
else {
        alert('ZIP code must have numeric characters only');
        uzip.focus();
        return false;
        }
}
function validateEmail(uemail) {
        var mailformat = /^\w+([\.-]?\w+)*@\w+([\.-]?\w+)*(\.\w{2,3})+$/;
        if(uemail.value.match(mailformat))
                return true;
        else    {
```

```
                alert("You have entered an invalid email address!");
                uemail.focus();
                return false;
        }
}
function validateSex(usex) {
        if(usex.value != "")
                return true;
        else {
                alert("Select Male/Female");
                document.getElementsByName('usex')[0].focus();
                return false;
        }}
h1 {
margin-left: 70px;
color: green;
}
form li {
list-style: none;
margin-bottom: 5px;
}
form ul li label{
float: left;
clear: left;
width: 100px;
text-align: right;
margin-right: 10px;
font-family: Verdana, Arial, Helvetica, sans-serif;
font-size: 14px;
}
form ul li input, select, span {
float: left;
margin-bottom: 10px;
}
form textarea {
float: left;
width: 350px;
height: 150px;
}
[type="submit"] {
clear: left;
margin: 20px 0 0 230px;
font-size:18px
}
p {
margin-left: 70px;
font-weight: bold;
}
```

jQuery

- jQuery is a lightweight, "write less, do more", JavaScript library.
- The purpose of jQuery is to make it much easier to use JavaScript in website.
- jQuery takes a lot of common tasks that require many lines of JavaScript code to accomplish, and wraps them into methods that can be called with a single line of code.
- jQuery also simplifies a lot of the complicated things from JavaScript, like AJAX calls and DOM manipulation.
- jQuery has plugins for almost any task out there.

The jQuery library contains the following features:

- HTML/DOM manipulation
- CSS manipulation
- HTML event methods
- Effects and animations
- AJAX
- Utilities

There are two versions of jQuery available for downloading:

- **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)

Both versions can be downloaded from jQuery.com.

The jQuery library is a single JavaScript file, and you reference it with the HTML <script> tag (notice that the <script> tag should be inside the <head> section):

```
<head>
    <script src="jquery-3.3.1.min.js"></script>
    <!-- Google CDN -->
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
    <!-- Microsoft CDN -->
    <script src="https://ajax.aspnetcdn.com/ajax/jQuery/jquery-3.3.1.min.js"></script>
</head>
```

With jQuery you select (query) HTML elements and perform "actions" on them.
The jQuery syntax is tailor-made for selecting HTML elements and performing some action on the element(s).

Basic syntax is: $(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)

Examples:
$(this).hide() - hides the current element.

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

All jQuery methods are written inside a document ready event

```
$(document).ready(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.
- Here are some examples of actions that can fail if methods are run before the document is fully loaded:
  - Trying to hide an element that is not created yet
  - Trying to get the size of an image that is not loaded yet

Shorter method for the document ready event

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

jQuery Selectors

- jQuery selectors allow to select and manipulate HTML element(s).
- jQuery selectors are used to "find" (or select) HTML elements based on their name, 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.
To select all <p> elements on a page

```
$(“p”)
```

ID Selector

The jQuery #id selector uses the id attribute of an HTML tag to find the specific element.
An id should be unique within a page, use #id selector for finding a single, unique element.
To find an element with a specific id, write a hash character, followed by the id of the HTML element:

```
$(“#txtName”)
```

Class Selector

The jQuery class selector finds elements with a specific class.
To find elements with a specific class, write a period character, followed by the class name

```
$(“.button”)
```

Syntax Description	Example
$("*")	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 element of the first
$("ul li:first-child")	Selects the first element of every
$("[href]")	Selects all elements with an href attribute
$("a[target='_blank']")	Selects all <a> elements with a target attribute value equal to "_blank"
$("a[target!='_blank']")	Selects all <a> elements with a target attribute value NOT equal to "_blank"
$(":button")	Selects all <button> elements and <input> elements of type="button"
$("tr:even")	Selects all even <tr> elements
$("tr:odd")	Selects all odd <tr> elements

Example

```
<!DOCTYPE html>
<html>
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
<script>
$(document).ready(function(){
   $("button").click(function(){
      $("p").hide();
                  $("#test").css("background-color", "lightyellow");
                  $(".test").css("background-color", "lightblue");
   });
});
</script>
</head>
<body>
<h2>This is a heading</h2>
<p>This is a paragraph.</p>
<p>This is another paragraph.</p>
<h2 id="test">This is another paragraph.</h2>
<h3 class="test">This is another paragraph.</h2>
<button>Click me to hide paragraphs</button>
</body>
</html>
```

Events

- jQuery is tailor-made to respond to events in an HTML page.
- All the different visitor's actions that a web page can respond to are called events.
- An event represents the precise moment when something happens.
- For example: moving a mouse over an element, selecting a radio button, clicking on an element

DOM Events

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

To assign a click event to all paragraphs on a page

```
$(“p”).click(function() {
        // code goes here;
});
```

click(): is executed when the user clicks on the HTML element.
dblclick():is executed when the user double clicks on the HTML element.
mouseenter():is executed when the mouse pointer enters the HTML element
mouseleave():is executed when the mouse pointer leaves the HTML element
mousedown():is executed when the left, middle or right mouse button is pressed down, while the mouse is over the HTML element
mouseup():is executed when the left, middle or right mouse button is released, while the mouse is over the HTML element
hover(): method takes two functions and is a combination of the mouseenter() and mouseleave() methods.
focus():is executed when the form field gets focus
blur():is executed when the form field loses focus
on(): method attaches one or more event handlers for the selected elements

Example

```
<!DOCTYPE html>
<html>
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
<script>
$(document).ready(function(){
   $("p").click(function(){
      $(this).hide();
   });
        $("h1").dblclick(function(){
                $(this).hide();
        });
        $("h2").mouseenter(function(){
                alert("Mouse Enter Function Triggered!");
        });
        $("h2").mouseleave(function(){
                alert("Mouse Leave Function Triggered!");
        });
        $("h3").mousedown(function(){
                alert("Mouse Down Function Triggered!");
        });
```

```
        $("h4").mouseup(function(){
                alert("Mouse up Function Triggered!");
        });
        $("h5").hover(function(){
                alert("Hover in!");
        },
                function(){
                        alert("Hover Out");
        });
        $("input").focus(function(){
      $(this).css("background-color", "yellow");
                $(this).css("color", "black");
   });
   $("input").blur(function(){
      $(this).css("background-color", "blue");
                $(this).css("color", "white");
   });
        $("#pid").on({
                mouseenter: function(){
                        $(this).css("background-color", "lightgray");
                },
                mouseleave: function(){
                        $(this).css("background-color", "lightblue");
                },
                click: function(){
                        $(this).css("background-color", "yellow");
                }
        });
});
</script>
</head>
<body>
<p>Click to Hide.</p>
<h1>Double Click to Hide</h1>
<h2>Mouse Enter Leave Event</h2>
<h3>Mouse Down Event</h3>
<h4>Mouse Up Event</h4>
<h5>Hover Event</h5>
Register Number: <input type="text" name="registernumber"><br>
Date of Birth: <input type="text" name="dob">
<h6 id="pid">On Event</h6>
</body>
</html>
```

jQuery Effect Methods for creating animation

Method	Description
animate()	Runs a custom animation on the selected elements
clearQueue()	Removes all remaining queued functions from the selected elements
delay()	Sets a delay for all queued functions on the selected elements
dequeue()	Removes the next function from the queue, and then executes the function
fadeIn()	Fades in the selected elements
fadeOut()	Fades out the selected elements
fadeTo()	Fades in/out the selected elements to a given opacity
fadeToggle()	Toggles between the fadeIn() and fadeOut() methods
finish()	Stops, removes and completes all queued animations for the selected elements
hide()	Hides the selected elements
queue()	Shows the queued functions on the selected elements
show()	Shows the selected elements
slideDown()	Slides-down (shows) the selected elements
slideToggle()	Toggles between the slideUp() and slideDown() methods
slideUp()	Slides-up (hides) the selected elements
stop()	Stops the currently running animation for the selected elements
toggle()	Toggles between the hide() and show() methods

Example 1 (Effects):

```
<!DOCTYPE html>
<html>
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
<script>
$(document).ready(function(){
  $("#flip1").click(function(){
    $("#panel1").slideDown("slow");
  });
      $("#flip2").click(function(){
    $("#panel2").slideUp("fast");
  });
      $("#flip3").click(function(){
    $("#panel3").slideToggle(2000);
  });
      $("#btnChaining").click(function(){
    $("#p1").css("color", "blue").slideUp(2000).slideDown(2000);
  });
      $("#btnCallback").click(function(){
    $("#p2").hide("slow", function(){
      alert("The paragraph is now hidden");
    });
  });
});
</script>
```

```
<style>
#panel1, #flip1, #panel2, #flip2, #panel3, #flip3 {
   padding: 5px;
   text-align: center;
   background-color: blue;
         color: white;
   border: solid 1px #c3c3c3;
}

#panel1, #panel3 {
   padding: 50px;
   display: none;
}
#panel2 {
         padding: 50px;
}
</style>
</head>
<body>
<div id="flip1">Click to slide down panel</div>
<div id="panel1">Department of Computer Science and Engineering</div>
<div id="flip2">Click to slide up panel</div>
<div id="panel2">Department of Computer Science and Engineering</div>
<div id="flip3">Click to slide Toggle panel</div>
<div id="panel3">Department of Computer Science and Engineering</div>
<p id="p1">jQuery Chaining</p>
<button id="btnChaining">jQuery Chaining</button>
<button id="btnCallback">Callback</button>
<p id="p2">School of Computing</p>
</body>
</html>
```

Example 2 (Animation):

```
<!DOCTYPE html>
<html>
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
<script>
$(document).ready(function(){
   $("button").click(function(){
      $("#div1").animate({left: '250px'});
                  $("#div2").animate({
         left: '250px',
         opacity: '0.5',
         height: '150px',
         width: '150px'
      });
                  $("#div3").animate({
```

```
        left: '250px',
        opacity: '0.5',
        height: '+=150px',
        width: '+=150px'
    });
                $("#div4").animate({
        height: 'toggle'
    });
                var div1 = $("#div5");
    div1.animate({height: '300px', opacity: '0.4'}, "slow");
    div2.animate({width: '300px', opacity: '0.8'}, "slow");
    div3.animate({height: '100px', opacity: '0.4'}, "slow");
    div4.animate({width: '100px', opacity: '0.8'}, "slow");
                var div2 = $("#div6");
    div2.animate({left: '100px'}, "slow");
    div2.animate({fontSize: '3em'}, "slow");
        });
});
</script>
</head>
<body>

<button>Start Animation</button>
<p>By default, all HTML elements have a static position, and cannot be moved. To
manipulate the position, remember to first set the CSS position property of the element to
relative, fixed, or absolute!</p>
<div id="div1"
style="background:#98bf21;height:100px;width:100px;position:absolute;"></div>
<br><br><br><br><br><br><br><br><br><br>
<div id="div2"
style="background:#98bf21;height:100px;width:100px;position:absolute;"></div>
<br><br><br><br><br><br><br><br><br><br>
<div id="div3"
style="background:#98bf21;height:100px;width:100px;position:absolute;"></div>
<br><br><br><br><br><br><br><br><br><br>
<div id="div4"
style="background:#98bf21;height:100px;width:100px;position:absolute;"></div>
<br><br><br><br><br><br><br><br><br><br>
<div id="div5" style="background:#98bf21;height:100px;width:100px;position:absolute;">
</div>
<br><br><br><br><br><br><br><br><br><br>
<div id="div6"
style="background:#98bf21;height:100px;width:100px;position:absolute;color:white">
CSE</div>
</body>
</html>
```

jQuery HTML

jQuery methods for DOM manipulation are:

- 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() - to get attribute values

Example 1:

```
<!DOCTYPE html>
<html>
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
<script>
$(document).ready(function(){
        // Get Functions
        $("#btn1").click(function(){
    alert("Text: " + $("#p1").text());
  });
  $("#btn2").click(function(){
    alert("HTML: " + $("#p1").html());
  });
  $("#btn3").click(function(){
    alert("Value: " + $("#test").val());
  });
        $("#btn4").click(function(){
    alert($("#sist").attr("href"));
  });
        // Set Functions
        $("#sbtn1").click(function(){
    $("#set1").text("SoC");
  });
  $("#sbtn2").click(function(){
    $("#set2").html("<b>CSE</b>");
  });
  $("#sbtn3").click(function(){
    $("#set3").val("IoT");
  });
        $("#sbtn4").click(function(){
                $("#set4").attr("href", "https://www.facebook.com/SathyabamaOfficial");
        });
        $("#sbtn5").click(function(){
    $("#set5").attr("href", function(i, origValue){
        return origValue + "/SathyabamaOfficial";
    });
  });

});
```

```
</script>
</head>
<body>
<h1> HTML Get </h1>
<p id="p1">This is some <b>bold</b> text in a paragraph.</p>
<button id="btn1">Show Text</button>
<button id="btn2">Show HTML</button>
<p>Name: <input type="text" id="test" value="SoC-CSE"></p>
<button id="btn3">Show Value of Text Box</button>
<p><a href="http://www.sathyabama.ac.in" id="sist">SATHYABAMA</a></p>
<button id="btn4">Show href Value</button>

<h1> HTML Set </h1>
<p id="set1">School of Computing</p>
<p id="set2">Department of Computer Science and Engineering</p>
<p>Input field: <input type="text" id="set3" value="UIT"></p>
<p><a href="http://www.sathyabama.ac.in" id="set4">SATHYABAMA</a></p>
<button id="sbtn1">Set Text</button>
<button id="sbtn2">Set HTML</button>
<button id="sbtn3">Set Value</button>
<button id="sbtn4">Set Attribute</button>

<h1> Callback Function </h1>
<p><a href="https://www.facebook.com" id="set5">SATHYABAMA</a></p>
<button id="sbtn5">Callback</button>
</body>
</html>
```

Example 2:

With jQuery, it is easy to add new elements/content. The methods are:

- append() - Inserts content at the end of the selected elements
- prepend() - Inserts content at the beginning of the selected elements
- after() - Inserts content after the selected elements
- before() - Inserts content before the selected elements

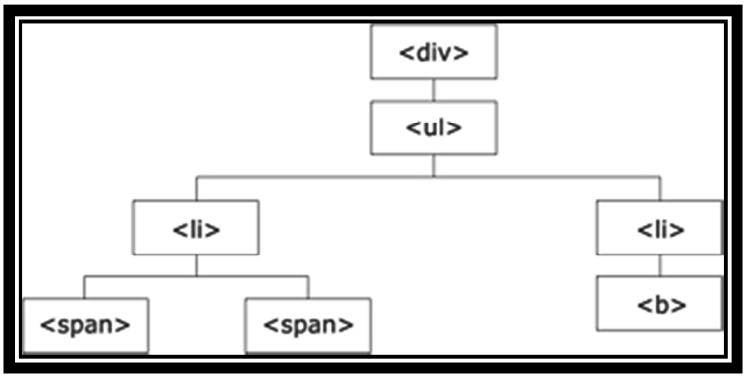
```
<!DOCTYPE html>
<html>
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
<script>
$(document).ready(function(){
   $("#btn1").click(function(){
                    $("#dom1").prepend(" <b><br>SATHYABAMA<br></b>");
      $("#dom1").append(" <b><br>Department of CSE</b>");
   });

   $("#btn2").click(function(){
```

```
        $("ol").append("<li>New Item 1</li>");
    });
        $("#btn4").click(function(){
            $("#dom1").remove();
        });
        $("#btn5").click(function(){
            $("#div1").empty();
        });
});
function appendText() {
    var txt1 = "<p>Text.</p>";              // Create text with HTML
    var txt2 = $("<p></p>").text("Text.");  // Create text with jQuery
    var txt3 = document.createElement("p");
    txt3.innerHTML = "Text.";               // Create text with DOM
    $("body").append(txt1, txt2, txt3);     // Append new elements
}
</script>
</head>
<body>
<p id="dom1">School of Computing</p>
<ol>
  <li>List item 1</li>
  <li>List item 2</li>
  <li>List item 3</li>
</ol>
<div id="div1">
        <p>Line 1</p>
        <p>Line 2</p>
        <button>Dummy Button</button>
</div>
<button id="btn1">Append text</button>
<button id="btn2">Append list items</button>
<button id="btn3" onclick="appendText()">Append Multiple text</button>
<button id="btn4">Remove p Tag with id dom1</button>
<button id="btn5">Empty Div Tag with id div1</button>
</body>
</html>
```

jQuery Traversing

jQuery traversing means "move through", used to "find" (or select) HTML elements based on their relation to other elements. Start with one selection and move through that selection until the elements of interest are found.

The image below illustrates an HTML page as a tree (DOM tree). With jQuery traversing, it is possible to move up (ancestors), down (descendants) and sideways (siblings) in the tree, starting from the selected (current) element. This movement is called traversing - or moving through - the DOM tree.



In the above figure

- The element is the parent of <ul>, and an ancestor of everything inside of it
- The <ul> element is the parent of both <li> elements, and a child of
- The left <li> element is the parent of , child of <ul> and a descendant of
- The element is a child of the left <li> and a descendant of <ul> and
- The two <li> elements are siblings (they share the same parent)
- The right <li> element is the parent of <b>, child of <ul> and a descendant of
- The <b> element is a child of the right <li> and a descendant of <ul> and

Example 1: - Ancestors and Descendants

```
<!DOCTYPE html>
<html>
<head>
<style>
.ancestors * {
   display: block;
   border: 2px solid green;
   color: green;
   padding: 5px;
   margin: 15px;
}
</style>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
<script>
$(document).ready(function(){
        $("#btn1").click(function() {
                $("span").parents().css({"color": "blue", "border": "2px solid blue"});
        });
        $("#btn2").click(function() {
                $("span").parents("ul").css({"color": "red", "border": "2px solid red"});

        });
        $("#btn3").click(function() {
                $("span").parentsUntil("div").css({"color": "orange", "border": "2px solid
orange"});
        });
        $("#btn4").click(function() {
                $("div").children().css({"color": "brown", "border": "2px solid brown"});
```

```
        });
        $("#btn5").click(function() {
                $("div").find("span").css({"color": "brown", "border": "2px solid brown"});
        });
        $("#btn6").click(function() {
                $("div").find("*").css({"color": "brown", "border": "2px solid brown"});
        });
});
</script>
</head>
<body class="ancestors">body (great-great-grandparent)
  <div style="width:500px;">div (great-grandparent)
    <ul>ul (grandparent)
      <li>li (direct parent)
        <span>span</span>
      </li>
    </ul>
  </div>
  <button id="btn1">Identify Parents</button>
  <button id="btn2">Parents of ul</button>
  <button id="btn3">Parents until div</button>
  <button id="btn4">Find Children</button>
  <button id="btn5">Find Children by name</button>
  <button id="btn6">Find all Children</button>
</body>
</html>
```

Example 2: Siblings

```
<!DOCTYPE html>
<html>
<head>
<style>
.siblings * {
    display: block;
    border: 2px solid blue;
    color: blue;
    padding: 5px;
    margin: 15px;
}
.disp {
        display: inline-block;
}
</style>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
<script>
$(document).ready(function(){
        $("#btn1").click(function() {
                $("h2").siblings().css({"color": "green", "border": "2px solid green"});
```

```
                $("p").text("Sibling of h2");
                $("h2").text("Sibling of h3, p, p, span");
                $("h3").text("Sibling of h2");
                $("span").text("Sibling of h2");
        });
        $("#btn2").click(function() {
                $("h2").siblings("p").css({"color": "green", "border": "2px solid green"});
                $("p").text("Sibling of h2");
        });
        $("#btn3").click(function() {
                $("h2").next().css({"color": "green", "border": "2px solid green"});
        });
        $("#btn4").click(function() {
                $("h2").nextAll().css({"color": "green", "border": "2px solid green"});
        });
        $("#btn5").click(function() {
                $("h2").nextUntil("h6").css({"color": "green", "border": "2px solid green"});
        });
});
</script>
</head>
<body class="siblings">
<div>div (parent)
  <p>p</p>
  <span>span</span>
  <h2>h2</h2>
  <h3>h3</h3>
  <h4>h4</h4>
  <h5>h5</h5>
  <h6>h6</h6>
  <p>p</p>
  <p>Similarly the methods prev(), prevAll(), prevUntil() - performs the reverse operations of next, nextAll() and nextUntil()</p>
</div>
<button id="btn1" class="disp">Find All Siblings</button>
<button id="btn2" class="disp">Find Sibling</button>
<button id="btn3" class="disp">Next Sibling</button>
<button id="btn4" class="disp">Next All Sibling</button>
<button id="btn5" class="disp">Next Until Sibling</button>
</body>
</html>
```

jQuery AJAX

- AJAX = Asynchronous JavaScript and XML.
- AJAX is the art of exchanging data with a server, and updating parts of a web page - without reloading the whole page.
- AJAX is about loading data in the background and display it on the webpage, without reloading the whole page.
- Examples of applications using AJAX: Gmail, Google Maps, Youtube, and Facebook tabs.
- With the jQuery AJAX methods, it is possible to request the text, HTML, XML, or JSON from a remote server using both HTTP Get and HTTP Post – can load the external data directly into the selected HTML elements of the web page!

load(): The load() method loads data from a server and puts the returned data into the selected element.

$(selector).load(URL,data,callback);

The optional 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 succeeds
- statusTxt - contains the status of the call
- xhr - contains the XMLHttpRequest object

Example:

```
<!DOCTYPE html>
<html>
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
<script>
$(document).ready(function(){
  $("#btn1").click(function(){
    $("#div1").load("sample.txt");
  });
      $("#btn2").click(function(){
    $("#div1").load("sample.txt", function(responseTxt, statusTxt, xhr){
            // For Error Message, rename the file to sample1.txt
                  alert("Response Text: " + responseTxt);
      if(statusTxt == "success")
        alert("External content loaded successfully!");
      if(statusTxt == "error")
        alert("Error: " + xhr.status + ": " + xhr.statusText);
    });
  });
});
</script>
</head>
<body>
<div id="div1"><h2>Let jQuery AJAX Change This Text</h2></div>
```

```
<button id="btn1">load()</button>
<button id="btn2">callback()</button>
</body>
</html>
```

get() and post(): Used to request data from the server with an HTTP GET or POST request.

- GET - Requests data from a specified resource
- POST - Submits data to be processed to a specified resource

GET is for retrieving data from the server. The GET method may return cached data.

POST is also used to get the data from the server. However, the POST method NEVER caches data, and is often used to send data along with the request.

get()

```
$.get(URL,callback);
```

- The required URL parameter specifies the requested URL.
- The optional callback parameter is the name of a function to be executed if the request succeeds.

Example

ajax1.html

```
<!DOCTYPE html>
<html>
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
<script>
$(document).ready(function(){
   $("button").click(function(){
      $.get("http://localhost/uitdemo/example1.php", function(data, status){
                              // example1.php is sufficient instead of providing the complete URL.
         alert("Data: " + data + "\nStatus: " + status);
      });
   });
});
</script>
</head>
<body>
<button>HTTP Request - PHP File and Callback for Result</button>
</body>
</html>
```

example1.php

```
<?php
        echo "Data from a PHP file";
?>
```

post()

```
$.post(URL,data,callback);
```

- The required URL parameter specifies the request URL.

- The optional data parameter specifies some data to send along with the request.
- The optional callback parameter is the name of a function to be executed if the request succeeds.

Example:

ajax.html

```
<!DOCTYPE html>
<html>
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
<script>
$(document).ready(function(){
   $("button").click(function(){
      $.post("example2.php",
      {
        school: "Computing",
        deptt: "Computer Science and Engineering"
      },
      function(data,status){
          alert("Data: " + data + "\nStatus: " + status);
      });
   });
});
</script>
</head>
<body>
<button>HTTP POST Request - Callback Result</button>
</body>
</html>
```

example2.php

```
<?php
        $school = $_POST['school'];
        $deptt = $_POST['deptt'];
        echo "School of " . $school;
        echo "Department of " . $deptt;
?>
```