CORE360 IT SERVICES
TRAINING. SOLUTIONS DEVELOPMENT. CONSULTING.
1A F. JACINTO STREET GALGUERRA COMPOUND DIAM BRGY. GEN. T. DELEON VALENZUELA CITY
CONTACT US 09228720135 | johnreygoh@gmail.com

WEB FUNDAMENTALS TRAINING COURSE

A Quick Guide To Establishing Your Foundation For Higher Level Web Technologies

Compiled by John Rey Goh
Core360 IT Services | johnreygoh@gmail.com

Module 1:The WEB and HTML

I. Web sites, web pages, web servers

II. Web technologies

III. Creating and designing your web pages --- online web page designers, content management systems, web authoring softwares, or manual coding

IV. HyperText Markup Language (HTML)

a. Tags

i. <!DOCTYPE html>
ii. <html>
iii. <head>
iv. <meta name=”keywords” content=”code,html,programming,rob” />
v. <meta name=”description” content =”rob faves web” />
vi. <meta http-equiv=”refresh” content=”5” />
→refreshes the page every 5 seconds
vii. <meta http-equiv=”refresh” content =”5; url=http://www.google.com” />
viii. <meta name=”author” content=”rob imperial” />
ix. <link> (ex: getting css external links)
x. <script> (ex:getting external javascript files)
xi. <title>
xii. <body>
xiii. <!-- -- >
xiv. <p>
xv. <br />
xvi. <hr />
xvii. <table><caption><tr><td or th>

Common attributes:

Cellpadding	(space between contetnt and border)
Cellspacing	(border thickness between cells)
Border	
Align	
Width	
Bgcolor	
Background	
Bordercolor	

<td rowspan=”2”…>
<td colspan=”3”…>

xviii. <pre>
xix. <font><u><b><i><strike><h1><h2><h3><h4><h5><sup><sub><q><code>

xx. <ul type=”square/disk/circle”><li></li>

xxi. <ol type=”1/i/I/a/A”><li></li>

xxii. <img src=? alt=? width=? height=? align=? />

xxiii. <a href=? Target=”_blank/_self”></a>

xxiv. <marquee direction=”?” width=”?” scrolldelay=”?” scrollamount=”?”

xxv. Core attributes on majority of the tags:

1.	Id	(unique identifier for the html element)
2.	Title	(tooltip)
3.	Class	(associate element with a css class)
4.	Style	(select a css style for the element)

xxvi. Other attributes

1. Align
2. Bgcolor
3. Background
4. Width
5. Height
6. Name
7. href

xxvii. <div><span>

xxviii. <form method=”post/get” action=”” enctype=”multipart/form-data” onSubmit=”?”>

xxix. <input type=”text” name=”?” pattern=”[0-9a-zA-Z]{3,8}” size=”?” maxlength=”?” required>

xxx. <input type=”password” name=”?”>

xxxi. <input type=”email” name=”?”>

xxxii. <input type=”radio” name=”?” value=”?” checked>

xxxiii. <input type=”checkbox” name=”?” value=”?” checked>

xxxiv. <input type=”button” name=”?” value=”?” onclick=”?”>

xxxv. <input type=”submit” name=”?” value=”?”>

xxxvi. <input type=”reset” name=”?” value=”?”>

xxxvii. <textarea name=”?” rows=”?” cols=”?”></textarea>

xxxviii. <select name=”?” *size=”4”>
<optgroup label=”?”>
<option value=”?” selected></option>
<option value=”?”></option>
<option value=”?”></option>
</optgroup>
</select>

xxxix. <input type=”file” name=”?” accept=”images/*”>

xl. <input type=”hidden” name=”?” value=”?”>

Module 2:Cascading Style-Sheets (CSS)

CSS is now the de-facto for implementing layout and design in elements and controls for web projects and applications.

I. Syntax

```
selector{
        property:value;
        }
```

II. Selctors

a. [type selector] `h1{color:red;}`

b. [universal selector] `*{color:blue;}`

c. [descendant selector] `ul li a{color:green;]`

d. [class selector] `.set1{color:yellow;}`

Ex. `<p class=”set1”>`

e. [id selector] `#set2{color:skyblue;}`

f. [child selector] `body>p{color:green;}`

g. [attribute selector] `input[“type=text”]{color:aqua;}`

h. [multiple style rules]

```
h1
{
color:red;
font-weight:normal;
text-transfer:lowercase;
}
```

i. [grouping selectors]

```
h1, h2, h3
{
color:red;
font-weight:normal;
text-transfer:lowercase;
}
```

III. Using css styles

a. Embedded (declaring a style rule on the <head> tag)

Ex.

```
<head>
        <style type=”text/css”>
        …
        </style>
</head>
```

b. Inline (using the style attribute inside the html tag)

Ex.

```
<div style=”background-color:yellow;border:2px solid black”>
```

c. External (create a separate .css file then include it on the page using the <link> in the <head>)

Ex.
```
<head>
	<link rel="stylesheet" type="text/css" href="rules_set1.css" />
</head>
```

d. Imported (importing an external .css file into the page)

Ex.
```
<head>
	<@import "mystyle.css">
</head>
```

e. Overriding and prioritization
   1. Inline
   2. Embedded
   3. External or imported

IV. Comments (/*…*/)

V. Measurement
   a. %
   b. cm
   c. em (height of font, ex. Font = 12pts so, 2em=24pts)
   d. ex (measurement relative to font's lowercase x-height)
   e. in
   f. mm
   g. pc (picas,1pc=12pts)
   h. pt(points, smaller than pica)
   i. px (pixels, based on screen resolution)

VI. Colors (see color sheet reference)

hex	#rrggbb	ex.	p{color:#ff0000;}
short hex	#rgb	ex.	p{color:#647;}
rgb%	rgb(rrr%,ggg%,bbb%)	ex.	p{color:rgb(50%,50%,50%);}
rgb absolute	rgb(rrr,ggg,bbb)	ex.	p{color:rgb(0,0,255);}
keyword	aqua,black,etc…	ex.	p{color:yellow;}

VII. Background
   a. background-color:black
   b. background-image:url(/images/pic1.jpg)
   c. background-repeat:norepeat or repeat
   d. background-position:100px →100px from left
      background-position:100px 200px →left and right position
   e. background-attachment:fixed or scroll

f. shorthand style
   ex.
   <p style=”background:url(/images/pic3.jpg) repeat fixed”>

VIII. Font
   a. font-family:georgia
   b. font-style:normal|italic|oblique
   c. font-variant:normal|smallcaps
   d. font-weight:bold|bolder|number?
   e. font-size:xx-small|x-small|small|medium|large|x-large|xx-large|smaller|larger|pixels?|in?|%?
   f. font-stretch:normal|wider|narrower|ultra-condensed|extra-condensed|condensed|semi-condensed|semi-expanded|expanded|extra-expanded|ultra-expanded

IX. Text
   a. color:red
   b. direction:ltr|rtl
   c. letter-spacing:normal|number?
   d. word-spacing:normal|number?
   e. text-indent:%|number?
   f. text-align:right|center|left
   g. text-decoration:underline|line-through|overline|blink
   h. text-transform:capitalize|lowercase|uppercase
   i. white-space:normal|pre|nowrap
   j. text-shadow:4px 4px 8px blue

X. Images
   a. border:1px dashed red
   b. height:100px
   c. width:100px

XI. Links
   a. [types}
      i. a:link
      ii. a:visited
      iii. a:hover
      iv. a:active
      ex:
         a:link{color:yellow;}

XII. Tables
   a. border-collapse:collapse|separate
   ex.
   <style type="text/css">

```
table.one {border-collapse:collapse;}
table.two {border-collapse:separate;}

td.a {
border-style:dotted;
border-width:3px;
border-color:#000000;
padding: 10px;
}

td.b {border-style:solid;
border-width:3px;
border-color:#333333;
padding:10px;
}
</style>

<table class="one">
<caption>Collapse Border Example</caption>
<tr><td class="a"> Cell A Collapse Example</td></tr>
<tr><td class="b"> Cell B Collapse Example</td></tr>
</table>
<br />
<table class="two">
<caption>Separate Border Example</caption>
<tr><td class="a"> Cell A Separate Example</td></tr>
<tr><td class="b"> Cell B Separate Example</td></tr>
</table>
```

b. empty-cells:show|hide

XIII. Border

a. border-color:green
   i. border-bottom-color:?
   ii. border-top-color:?
   iii. border-left-color:?
   iv. border-right-color:?

b. border-style:none|solid|dotted|dashed|double|groove|ridge|inset|outset|hidden
   i. border-bottom- style:?
   ii. border-top- style:?
   iii. border-left- style:?
   iv. border-right- style:?

c. border-width:number?
    i. border-bottom-width:?
    ii. border-top- width:?
    iii. border-left- width:?
    iv. border-right- width:?

XIV. Margins

Ex.

p{margin:15px;} →all sides

p{margin:15px 2%;} →top&bottom left&right

p{margin:15px 2% -10px auto;} →top left bottom right

XV. Lists

a. list-style-path
    i. <ul> →disc,circle,square
    ii. <ol>
        1. decimal
        2. decimal-leading-zero
        3. lower-alpha
        4. upper-alpha
        5. lower-roman
        6. upper-roman

b. list-style-image:url(/mybullets/b1.gif)

c. marker-offset:2em

XVI. Paddings (space between content and border, values:length,percent,inherit)

a. Ex.
   <p style="padding-bottom:5%;border:1px solid black">

b. Ex. (all 4 paddings)
   <p style="padding:5%;border:1px solid black">

c. Ex. (4 padding definition)
   <p style="padding:5% 2% 5% 2%;border:1px solid black">

Module 3: Client-Side Script (Javascript)

JavaScript is a client-side scripting language that is popularly and widely used as part of most web applications. It is now also being used as the script for creating and managing a variety of other well-known web application and scripting libraries and frameworks such as JQuery, Node.js, AngularJS and many others.

JavaScript offers cross-browser compatibility, can easily be integrated to new and existing web projects, and provide behavioral and event-programming functions along with object-oriented programming capabilities.

JavaScript can be called directly from a web page since the browser will be the one interpreting the code.

JavaScript functions are case-sensitive. Use a semi-colon to terminate each statement.

Example of a javascript in an html code:

```
<html>
<head><title>page1</title></head>
<body>

<script type="text/javascript">
document.write("iam created using javascript");
</script>

</body>
</html>
```

Example of a javascript in an html element event("onclick"):

```
<input type="button" name="btn1" id="btn01" value="click me" onclick="alert('hi');">
```

Example of a javascript in a function on the same page:

```
<html>
<head>
<title>page1</title>
<script type="text/javascript">
function showtext()
{
alert("iam created using javascript");
}
</script>
</head>

<body>
<input type="button" name="btn1" id="btn01" value="click me" onclick="showtext();">
</body>
</html>
```

Example of referencing an external javascript page to use its functions:

```
<head>
<script type="text/javascript" src="jscript1.js" />
</head>
```

- Displaying Alerts

```
//hi, iam a comment. I won't appear on the output
//the ff code is an alert box
```

```
alert("hi there!");

//the following is an example of a confirm box
var x = confirm("press a button:");
if(x==true){ alert("you pressed ok"); }
else{ alert("you pressed cancel"); }
```

- Declaring Variables

Variables hold data for processing. Can also hold control and event references.

Primitive Data Types:

1. Number
2. String
3. Boolean
4. Null
5. Object
6. Symbol
7. Undefined

Sample variable declaration:

*You don't have to specifically choose a data type upon variable creation. The data type will be automatically determined while the program is being processed.

```
var firstname = "marian";
alert(firstname);

var numberx = 45;
alert(numberx);
```

- Operators and Math Expressions

<u>Common arithmetic operators:</u>

Parenthesis	()
Addition	+
Subtraction	-
Multiplication	*
Division	/
Remainder	%

<u>Comparison operators:</u>

Equality	==
Non-equality	!=

<u>Relational operators:</u>

Greater than	>
Greater than or equal	>=
Less than	<
Less than or equal	<=

Logical operators:

And	&&
Or	||
Not	!

```
//sample math expression:
//note: follows PMDAS operator precedence
var ans=((var1+var2+var3)-var4)*var5;

//set decimal places
var num = 12.455678;
num=num.toFixed(2);//decimal places

//number type conversion
parseInt(float/string variable)
parseFloat(int/string variable)
```

sample:

```
<html>
<head>
<title>page1</title>
<script type="text/javascript">
function addme()
{

      var x=parseFloat(document.getElementById("text01").value);
      var y=parseFloat(document.getElementById("text02").value);
      var z=x+y;
      alert(z);
}
</script>
</head>
<body>
<input type="text" name="text1" id="text01"><br />
<input type="text" name="text2" id="text02"><br />
<input type="button" name="btn2" id="btn02" value="add" onclick="addme();">
</body>
</html>
```

- Prompts

Sample:

```
var name = prompt("enter name: ",defaultvalue);
alert("you entered " + name);
```

- Conditional Statements (IF and Switch)

Sample:

```
//if condition sample code
if(condition) { action; }
else if (condition) { action; }
else if (condition) { action; }
else {action; }

//switch statement sample code
switch(var to test)
{
```

```
case val1: action;break;
case val2: action;break;
case val3: action;break;
default:action;break;
}
```

- Arrays

```
//creating array
var  mysubjects = [“database”,”web”,”network”,”mobile”];

//to use array elements, use their array index number. The first item is 0.
alert(“the second subject is ” + mysubjects[1]);

//creating an empty array
var myfriends = [];

//then adding elements to the array
myfriends[0]=”angel”;
myfriends[1]=”castro”;
myfriends[2]=”delilah”;

//editing an array element value
myfriends[1]=”cassie”;

//adding more elements at the end of the array
myfriends.push(”bernard”,”anna”);

//adding elements at the beginning of the array
myfrineds.unshift(“kevin”,”sven”);

//removing the last element of the array
myfriends.pop();

//removing the first element of the array
myfriends.shift();

//splicing elements in the array
myfriends.splice(3,2,”elsa”,”hans”);      //1st  param=where to start, 2nd param=how many
to remove after the start position, succeeding params=array element to insert.

myfriends.splice(3,0,”troll”,”king”,”queen”);   //spliced the elements into the array
from the 3rd position without removing any other elements.

myfriends.splice(5,2);  //removing two elements from the third element on the array
without adding new elements.
```

- FOR loops

```
for(var counter=0;counter!=50;counter++)
{
action;
}

//looping though an array
var mypets=[“lion”,”tiger”,”panther”];
for(var x=0;x<=mypets.length;x++)
{
alert(mypets[x]);
}
```

```
//checking for a match in the array
var tosearch = document.getElementById("text03").value;
var mypets=["lion","tiger","panther"];
var matchfound="no";    //initializing a variable with a default value. This is called a
flag.
for(var x=0;x<mypets.length;x++)
{
     If(tosearch==mypets[x])
     {
     matchfound="yes";
     alert(mypets[x]);
     break;
     }
}
```

- while loops

```
var counter=0;
while(counter<=50)
{
action;
counter++;
}
```

- do...while loops

```
var counter=0;
do
{
action;
counter++;
}while(counter!=50)
```

- Using Pre-built String Functions

```
//create a string then use the following properties and functions to manipulate the
string.
var s = document.getElementById("text04").value;


s.length
s.charAt(int)            →gets character at specified position
s.indexOf(value)         →searches for an instance of a character or string,
                          returns -1 if not found
s.substr(start,#ofchars)
s.substring(start,end)
s.toLowerCase()
s.toUpperCase()
s.replace("old","new")
s.endswith(string)       →returns Boolean true if the string ends with the specified
string
s.trim()                 →removes leading and trailing whitespaces
```

- Rounding numbers

```
//round to the nearest whole number
var rn=Math.round(90.877654);
alert(rn);  //returns "90"
```

```
//round to 2 decimal places
var rn=Math.round(90.877654*100)/100;
alert(rn);
```

- Generating random numbers

```
//generates a random number from 1-10
Math.floor((Math.random() * 10) + 1);
```

- Converting strings to integers and float, vice-versa

```
//check the data type of a variable
typeof variable;

//string to integer
var i=parseInt("40");
var f=parseFloat("40"); //returns number with decimal places
var s=numbervar.toString();
```

- Getting the current date and time

```
//getting current system time and date
var mydate = new Date();
mydate.getDate();       //1-31
mydate.getMonth();      //0-11
mydate.getDay();        //0-6
mydate.getFullYear();
mydate.getHours();      //24-hr format
mydate.getMinutes();
mydate.getSeconds();

//date difference.
//Get the two dates
var startdate = new Date();                        //gets current date
var enddate = new Date("October 13, 2016");        //sets another date manually.

//you can also add time to the manually entered date.
//Ex: (var d=new Date("January 13, 2020 14:30:00"))

var millisecdiff = enddate - startdate;
var hoursdiff = millisecdiff / 1000;               //gets difference in hours
var daysdiff = millisecdiff / (1000*60*60*24);     //gets difference in days
```

- Functions:

```
function functionname(parameter1, parameter2)
{
var peso = (parameter 1* 45)+parameter2;
return peso;
}

//to use
var answer = functionname(78,20);
document.write(answer);

//automate a function
setTimeout(functionname(),10000);
setInterval(functionname(),3000);
```

```
//call a function from control event
onClick=functionname()
onMouseOver=functionname()
onSubmit=functionname()
```

- Placing scripts

<u>Example of a javascript in an html code:</u>
```
<html>
<head><title>page1</title></head>
<body>

<script type="text/javascript">
document.write("iam created using javascript");
</script>

</body>
</html>
```

<u>Example of a javascript in an html element event(“onclick”):</u>
```
<input type="button" name="btn1" id="btn01" value="click me" onclick="alert('hi');">
```

<u>Example of a javascript in a function on the same page:</u>
```
<html>
<head>
<title>page1</title>
<script type="text/javascript">
function showtext()
{
alert("iam created using javascript");
}
</script>
</head>

<body>
<input type="button" name="btn1" id="btn01" value="click me" onclick="showtext();">
</body>
</html>
```

<u>Example of referencing an external javascript page to use its functions:</u>
```
<head>
<script type=”text/javascript” src=”jscript1.js” />
</head>
```

- Events

popular control events

a. Form control or html element events
    i. onClick
    ii. onSubmit
    iii. onMouseOver
    iv. onMouseOut
    v. onMouseDown
    vi. onMouseUp
    vii. onBlur

viii. onFocus
ix. onChange
x. onSelect
xi. onReset
xii. onSubmit

b. body events
    i. onLoad
    ii. onPageHide
    iii. onPageShow
    iv. onResize
    v. onUnload

- Events: link

```
//showing the use of javascript:void(0) to negate the action of the link then using an
event "onClick" to trigger a javascript function

<a href="JavaScript:void(0)" onClick="alert('Hi');">Click</a>
```

- Events: button

```
<input type="button" name="btn1" id="btn01" value="click me"
onMouseOver="value='yes me'"
onMouseOut="value='click me'"
onClick="alert('singe click');"
onDblClick="alert('2x click');">
```

- Events: mouse

```
<img src="sample1.jpg" onMouseOver="src='sample2.jpg'" onMouseOut="src='sample1.jpg'" />
```

Or by using javascript to manipulate html element properties

```
<input type="button" onMouseOver="document.getElementById("img01").src='picture2.jpg';">
```

- Events: fields

Can be used for form input field validation and behavioral control.

```
<input type="text"
onFocus="this.style.backgroundcolor='yellow'"
onBlur="this.style.backgroundcolor='white'">
```

- Reading field values

```
//getting values from html input fields (input field not in a form)

function getinput(x)
{
var input1 = document.getElementById(x).value;

//this tests if field has data. Use if(!input){} to test it the other way around.
if(input1)
{
alert(input1);
}else{
document.getElementById(x).style.backgroundcolor="green";
alert("please fill up the green colored fields");
}

}

<input type="text" name="text1" id="text01">
<input type="button"  name="button1" id="button01" onClick="getinput('text01');">
```

```
//getting values from html input fields (using a form for submission)
function getinput(x)
{
var input1 = document.getElementById(x).value;

//this tests if field has data. Use if(!input){} to test it the other way around.
if(input1)
{
alert(input1);
}else{
document.getElementById(x).style.backgroundcolor="green";
alert("please fill up the green colored fields");
}

}

<form onSubmit="getinput('text01');">
<input type="text" name="text1" id="text01">
<input type="button"  name="button1" id="button01">
</form>

//creating a reusable function that dynamically checks and gets the ID of whoever calls
the function.
function get_id(mycontrol)
{
var id = mycontrol.id;
//…codes that utilizes the id…
}


//on control event
…onClick="get_id(this)";


//getting values from other html tag that act as container such as <p> or <div>
document.getElementById("p1").innerHTML
```

- Setting field values

Sample:

```
function fillfields(x)
{
      var i=document.getElementById(x).value;
      var newval="";
      switch(i)
      {
            case "101":newval="lobby";break;
            case "102":newval="reception";break;
            case "103":newval="classroom";break;
            default:newval="";
      }

      document.getElementById("text02").value=newval;
}

<input type="text" name="text1" id="text01" onBlur="fillfields('text01');"><br />
<input type="text" name="text2" id="text02"><br />
<input type="button" name="btn2" id="btn02" value="add" onClick="fillfields('text01');">
```

- Reading and setting paragraph text

*note: using *document.getElementById(?).innerHTML="value/text"* allows you to insert almost any content into other html containers such as a *<div>* or others.

```
//create a function to call your full paragraph
function fullparag1()
{

var fp1="What is JavaScript? JavaScript (often shortened to JS) is a lightweight,
interpreted, object-oriented language with first-class functions, and is best known as
the scripting language for Web pages, but it's used in many non-browser environments as
well. It is a prototype-based, multi-paradigm scripting language that is dynamic, and
supports object-oriented, imperative, and functional programming styles.<br>JavaScript
runs on the client side of the web, which can be used to design / program how the web
pages behave on the occurrence of an event. JavaScript is an easy to learn and also
powerful scripting language, widely used for controlling web page behaviour.<br>Contrary
to popular misconception, JavaScript is not \"Interpreted Java\". In a nutshell,
JavaScript is a dynamic scripting language supporting prototype based object
construction. The basic syntax is intentionally similar to both Java and C++ to reduce
the number of new concepts required to learn the language. Language constructs, such as
if statements, for and while loops, and switch and try... catch blocks function the same
as in these languages (or nearly so.) <br>JavaScript can function as both a procedural
and an object oriented language. Objects are created programmatically in JavaScript, by
attaching methods and properties to otherwise empty objects at run time, as opposed to
the syntactic class definitions common in compiled languages like C++ and Java. Once an
object has been constructed it can be used as a blueprint (or prototype) for creating
similar objects.<br>JavaScript's dynamic capabilities include runtime object
construction, variable parameter lists, function variables, dynamic script creation (via
eval), object introspection (via for ... in), and source code recovery (JavaScript
programs can decompile function bodies back into their source text).";

document.getElementById("parag1").innerHTML=fp1;

}

//create a shortened version of your paragraph and a link to trigger the paragraph
expansion.

<p id="parag1">
What is JavaScript?JavaScript® (often shortened to JS) is a lightweight, interpreted,
object-oriented language with first-class functions, and is best known as the scripting
language for Web pages, but it's used in <a href="javascript:void();"
onCLick="expandparag1();">read more…</a>
</p>
```

- Manipulating images and text

*note: the ff activity will show how to use a CSS class-rule through event-trigger function in javascript.

```
//create a css class-rule(sample: make an HTML element invisible)
<style type="text/css">
.hidden {display:none;}
.showme{display:block;}
</style>

//create a javascript function to invoke the css-rule
function makedisappear(x)
{
document.getElementById(x).className="hidden";
}

function makeappear(x)
{
document.getElementById(x).className="showme";
}


//add events to the control to call the javascript functions
<img src="boy1.gif" id="pic01" height="60" width="70"
onmouseout="makeappear('pic01');"
onmouseover="makedisappear('pic01');">

//if the html element already has a class for css styling and you don’t want to override
that class, you //can add additional class definition by editing the javascript code to
“add” another class.
Sample:
function makedisappear(x)
{
document.getElementById(x).className+=" hidden"; //a space before the classname is
needed.
}

//swapping pictures
function swappic(id,newpic)
{
document.getElementById(id).src=newpic;
}

//add an event on the control to trigger swapping the image
<img src=”pic1.jpg” id=”pic1” onMouseOver=”swappic(‘pic1’,’pic2.jpg’);”>
```

- Setting styles

```
//if used within a function
document.getElementById(?).style.backgroundColor=”yellow”;
document.getElementById(?).style.visibility=”hidden”;

//if used inline within the html tag.Sample:
<input type=”text” id=”text1” onFocus=”this.style.backgroundColor=’yellow’”
onBlur=”this.style.backgroundColor=’white’”>
```

- Target all elements by tag name

```
//select the html tag that you wish to target. Example: <p>
```

```
<p>paragraph1. this is an example of paragraph1</p>
<p>paragraph2. this is an example of paragraph2</p>
<p>paragraph3. this is an example of paragraph3</p>

//prepare javascript function to loop through all instances of that html tag to set
desired attribute
function setparagraphcolor()
{

//get an array of all the elements with the matching tag.example:<p>
//the first instance of the <p> will be assigned to p[0], then p[1]...
var pa=document.getElementsByTagName("p");

//loop to execute the action to all instances.
for(var i=0;i<pa.length;i++)
{
pa[i].style.color="yellow";
}

}

//prepare the html element to trigger the function
<input type="button" value="color it yellow" onclick="setparagraphcolor();">
```

- Target some elements by tag name

```
//as you loop through all the instances of the tag using
document.getElementsByTagName("?"), you can set condition(s) before executing the action.

function setparagraphcolor()
{

//get an array of all the elements with the matching tag.example:<p>
//the first instance of the <p> will be assigned to p[0], then p[1]...
var pa=document.getElementsByTagName("p");

//loop to execute the action to all instances.


for(var i=0;i<pa.length;i++)
{
//get attribute then set a condition
var ptext=pa[i].innerHTML;
if(ptext.endsWith("2"))
{
pa[i].style.color="yellow";
}
}

}
```

- Browser control: Filling the window with content

```
//open up a new window
function createwindow()
{
var newwindow=window.open();
}

//open up a new window then add content
```

```
function createwindow()
{
var newwindow=window.open();

var content1="<img src='boy1.gif' id='pic01' height='250' width='200'
onmouseout='makeappear('pic01');' onmouseover='makedisappear('pic01');'><img
src='dog.gif' id='pic01' height='250' width='200' ><img src='hula.gif' id='pic01'
height='250' width='250' >";

newwindow.document.write(content1);
}


//create and open a window then function to close it
var newwindow;
function createwindow()
{
      newwindow=window.open("http://www.facebook.com/");

}


function closedummy()
{
      newwindow.close();
}

<input type="button" id="btn50" value="new window" onClick="createwindow();">
<input type="button" id="btn51" value="close new window" onClick="closedummy();">
```

- Browser control: Controlling the window's size and location

```
var newwindow=window.open("url","name","width=200,height=200","left=200,top=100")
```

- Browser control: Testing for popup blockers

```
//for browsers except IE. For IE use the test (testPop===”undefined”)
function checkForPopBlocker() {
var testPop = window.open("", "","width=100,height=100");
if (testPop === null) {
alert("Please disable your popup blocker.");
}
testPop.close();
}

<body onLoad ="checkForPopBlocker();">
```

- Form validation: text fields

```
//a sample function to test if a textfield in blank upon page submission or on the
textfield’s own onBlur event

function checkForLastName() {
var targetField = document.getElementById("lastNameField");
if (targetField.value.length === 0) {
alert("Please enter your last name");
targetField.focus();
targetField.style.background = "yellow";
return false;
}
targetField.style.background = "white";
}
```

- Form validation: drop-downs

```
//sample markup for a combobox
<form onSubmit="return checkForSelection('states');">
<select id="states">
<option value="" selected="selected">
SELECT A STATE</option>
<option value="AL">Alabama</option>
<option value="AK">Alaska</option>
<option value="AZ">Arizona</option>
<option value="AR">Arkansas</option>
</select>&nbsp;&nbsp;
<input type="submit" value="Submit Form">
</form>

//sample function for validation
function checkForSelection(selecID) {
var target = document.getElementById(selecID);
if (target.selectedIndex === 0) {
alert("Please select a state.");
return false;
}
}
```

- Form validation: radio buttons

```
//actually easier if you simply set the property "checked" on one of the radio buttons.
//sample markup for radio buttons
<form onSubmit="return validateRadios();">
<input type="radio" name="r1" value="cat"> Cat<br>
<input type="radio" name="r1" value="bat"> Bat<br>
<input type="radio" name="r1" value="hat"> Hat<br>
<input type="submit" value="Submit Form">
</form>

//sample function to check if a user have selected a radio button
function validateRadios() {
 var radios = document.getElementsByName("r1");
 for (var i = 0; i < radios.length; i++) {
if (radios[i].checked) {
return true;
}
}
alert("Please check one.");
return false;
}
```

- Form validation: email

```
//easier by using html5 <input type="email">

//here is the javascript function that uses regex to check for the correct email format
function validateEmail() {
var eEntered = document.getElementById("address").value;
var regex = /^[\w\-\.\+]+\@[a-zA-Z0-9\. \-]+\.[a-zA-z0-9]{2,4}$/;
if (!(eEntered.match(emailCorrectPattern))) {
alert("Please correct email address");
return false;
}
}
```

- Exceptions: try and catch

```
//trying a risky code then displaying the error as generated by the browser

function greetWorld() {
try {
var greeting = "Hello world!";
aler(greeting);
}
catch(err) {
alert(err);
}
}
```

- Exceptions: throw

```
//assigning custom errors using throw instead of browser-generated errors

//this is a sample function that checks for password complexity
function checkPassword() {
try {

var pass = document.getElementById("f1").value;

if (pass.length < 8) {throw "Please enter at least 8 characters.";}

if (pass.indexOf(" ") !== -1) {throw "No spaces in the password, please.";}

var numberSomewhere = false;

for (var i = 0; i < pass.length; i++)
{
if (isNaN(pass(i, i+1)) === false) {numberSomewhere = true;break;}
}

if (numberSomewhere === false) {throw "Include at least 1 number.";}

}

catch(err) {
alert(err);
}
}
```

- Handling events within JavaScript

```
//you can trigger an HTML event programmatically in javascript
//the event (ex."onclick") should be lowercase

document.getElementById("button1").onclick = sayHello;
```