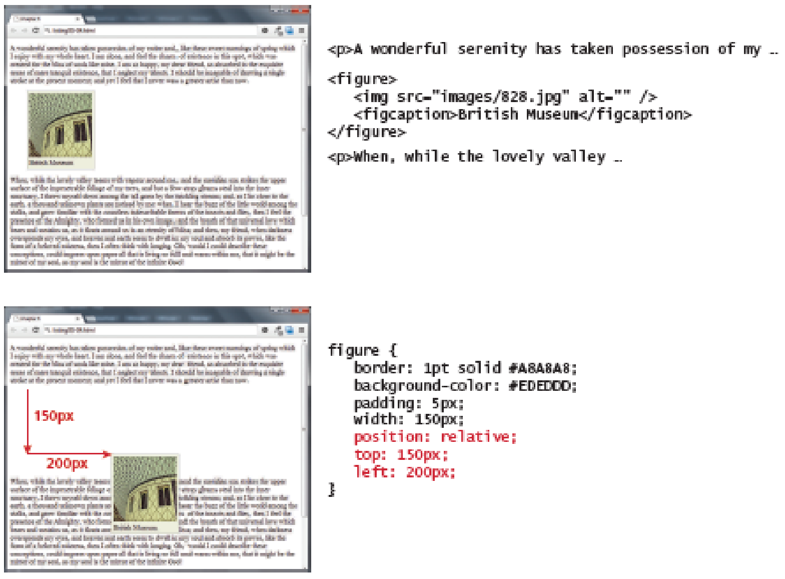
Internal Assessment Test 1 – Sept. 2019

Sub:	Web Technologies & its Applications			Sub Code:	15CS71	Branch :	CSE
Date:	12-10-2019	Duration:	90 min’s	Max Marks: 50	Sem / Sec:	7 – A, B & C	OBE

	Answer any FIVE FULL Questions	MARKS	CO	RBT
1 (a)	Discuss positioning elements with examples.	[10]	CO3	L2

Relative Positioning:

- In relative positioning an element is displaced out of its normal flow position and moved relative to where it would have been placed.
- When an element is positioned relatively, it is displaced out of its normal flow position and moved relative to where it would have been placed.
- The other content around the relatively positioned element “remembers” the element’s old position in the flow; thus the space the element would have occupied is preserved as shown in Figure 5.4.



```
<p>A wonderful serenity has taken possession of my …
<figure>
   <img src="images/828.jpg" alt="" />
   <figcaption>British Museum</figcaption>
</figure>
<p>When, while the lovely valley …
```

```
figure {
   border: 1pt solid #A8A8A8;
   background-color: #EDEDDD;
   padding: 5px;
   width: 150px;
   position: relative;
   top: 150px;
   left: 200px;
}
```

FIGURE 5.4 Relative positioning

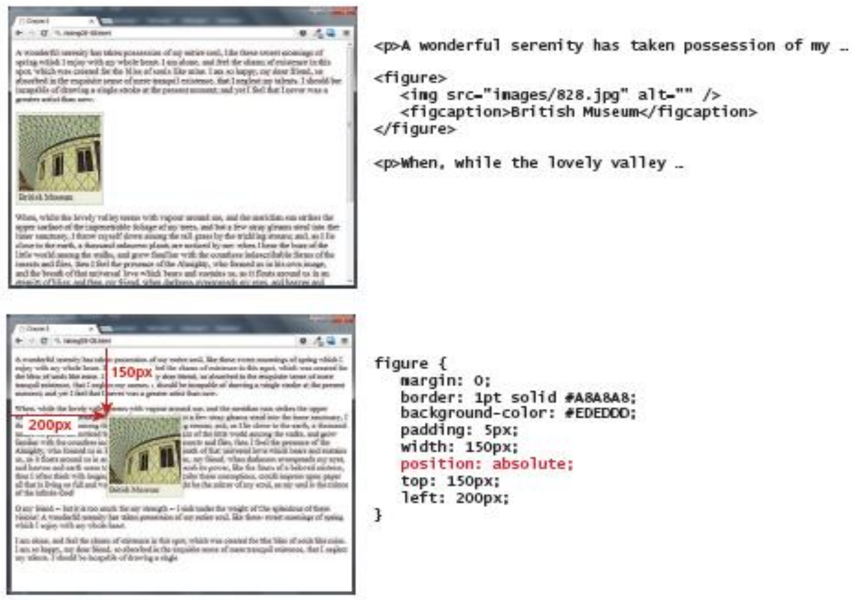
- As you can see in Figure 5.4, the original space for the positioned

<figure> element is preserved, as is the rest of the document's flow.

- As a consequence, the repositioned element now overlaps other content: that is, the <p> element following the <figure> element does not change to accommodate the moved<figure>.

Absolute Positioning:

- When an element is positioned absolutely, it is removed completely from normal flow. Thus, unlike with relative positioning, space is not left for the moved element, as it is no longer in the normal flow.
- Its position is moved in relation to its container block.
- In the example shown in Figure 5.5, the container block is the <body> element. Like with the relative positioning example, the moved block can now overlap content in the underlying normal flow.



```
<p>A wonderful serenity has taken possession of my …

<figure>
   <img src="images/828.jpg" alt="" />
   <figcaption>British Museum</figcaption>
</figure>

<p>When, while the lovely valley …
```

```
figure {
   margin: 0;
   border: 1pt solid #A8A8A8;
   background-color: #EDEDDD;
   padding: 5px;
   width: 150px;
   position: absolute;
   top: 150px;
   left: 200px;
}
```

FIGURE 5.5 Absolute positioning

- A moved element via absolute position is actually positioned relative to its nearest positioned ancestor container (that is, a block-level element whose position is fixed, relative, or absolute).
- In the example shown in Figure 5.6, the <figcaption> is absolutely positioned; it is moved 150 px down and 200 px to the left of its nearest positioned ancestor, which happens to be its parent (the <figure> element).

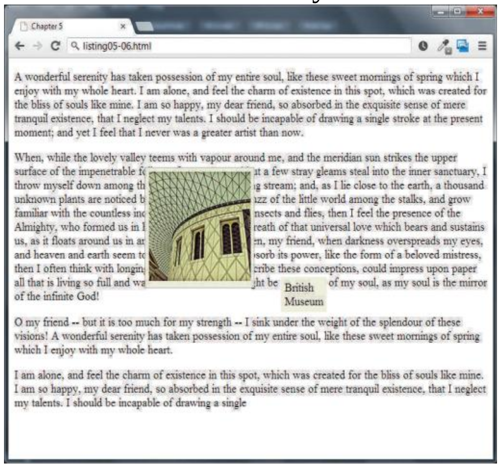
Z-Index

Looking at Figure 5.6, you may wonder what would have happened if the <figcaption> had been moved so that it overlapped the <figure>. Each positioned element has a stacking order defined by the z-index property (named for the z-axis). Items closest to the viewer (and thus on the top) have a larger z-index value, which can be seen in the first Unfortunately, working with z-index can be tricky and seemingly counterintuitive.

First, only positioned elements will make use of their z-index. Second, as can be seen in Figure , simply setting the z-index value of elements will not necessarily move them on top or behind other items.

Fixed Position

The fixed position value is used relatively infrequently. It is a type of absolute positioning, except that the positioning values are in relation to the viewport (i.e., to the browser window). Elements with fixed positioning do not move when the user scrolls up or down the page, as can be seen in Figure. The fixed position is most commonly used to ensure that navigation elements or advertisements are always visible.

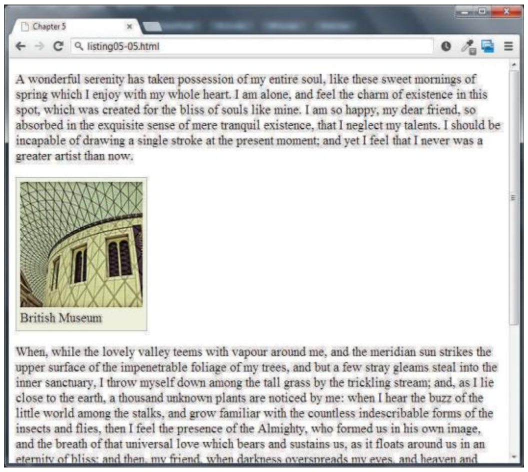


A wonderful serenity has taken possession of my entire soul, like these sweet mornings of spring which I enjoy with my whole heart. I am alone, and feel the charm of existence in this spot, which was created for the bliss of souls like mine. I am so happy, my dear friend, so absorbed in the exquisite sense of mere tranquil existence, that I neglect my talents. I should be incapable of drawing a single stroke at the present moment; and yet I feel that I never was a greater artist than now.

When, while the lovely valley teems with vapour around me, and the meridian sun strikes the upper surface of the impenetrable fo at a few stray gleams steal into the inner sanctuary, I throw myself down among th g stream; and, as I lie close to the earth, a thousand unknown plants are noticed b uzz of the little world among the stalks, and grow familiar with the countless inc nsects and flies, then I feel the presence of the Almighty, who formed us in h reath of that universal love which bears and sustains us, as it floats around us in ar en, my friend, when darkness overspreads my eyes, and heaven and earth seem to sorb its power, like the form of a beloved mistress, then I often think with longin cribe these conceptions, could impress upon paper all that is living so full and wa ght be British Museum of my soul, as my soul is the mirror of the infinite God!

O my friend -- but it is too much for my strength -- I sink under the weight of the splendour of these visions! A wonderful serenity has taken possession of my entire soul, like these sweet mornings of spring which I enjoy with my whole heart.

I am alone, and feel the charm of existence in this spot, which was created for the bliss of souls like mine. I am so happy, my dear friend, so absorbed in the exquisite sense of mere tranquil existence, that I neglect my talents. I should be incapable of drawing a single



A wonderful serenity has taken possession of my entire soul, like these sweet mornings of spring which I enjoy with my whole heart. I am alone, and feel the charm of existence in this spot, which was created for the bliss of souls like mine. I am so happy, my dear friend, so absorbed in the exquisite sense of mere tranquil existence, that I neglect my talents. I should be incapable of drawing a single stroke at the present moment; and yet I feel that I never was a greater artist than now.

British Museum

When, while the lovely valley teems with vapour around me, and the meridian sun strikes the upper surface of the impenetrable foliage of my trees, and but a few stray gleams steal into the inner sanctuary, I throw myself down among the tall grass by the trickling stream; and, as I lie close to the earth, a thousand unknown plants are noticed by me: when I hear the buzz of the little world among the stalks, and grow familiar with the countless indescribable forms of the insects and flies, then I feel the presence of the Almighty, who formed us in his own image, and the breath of that universal love which bears and sustains us, as it floats around us in an eternity of bliss; and then, my friend, when darkness overspreads my eyes, and heaven and

```
Va <p>A wonderful serenity has taken possession of my …
<figure>
<img src="images/828.jpg" alt="" />
<figcaption>British Museum</figcaption>
</figure>
<p>When, while the lovely valley …

lue Descrip figure {
margin: 0;
```

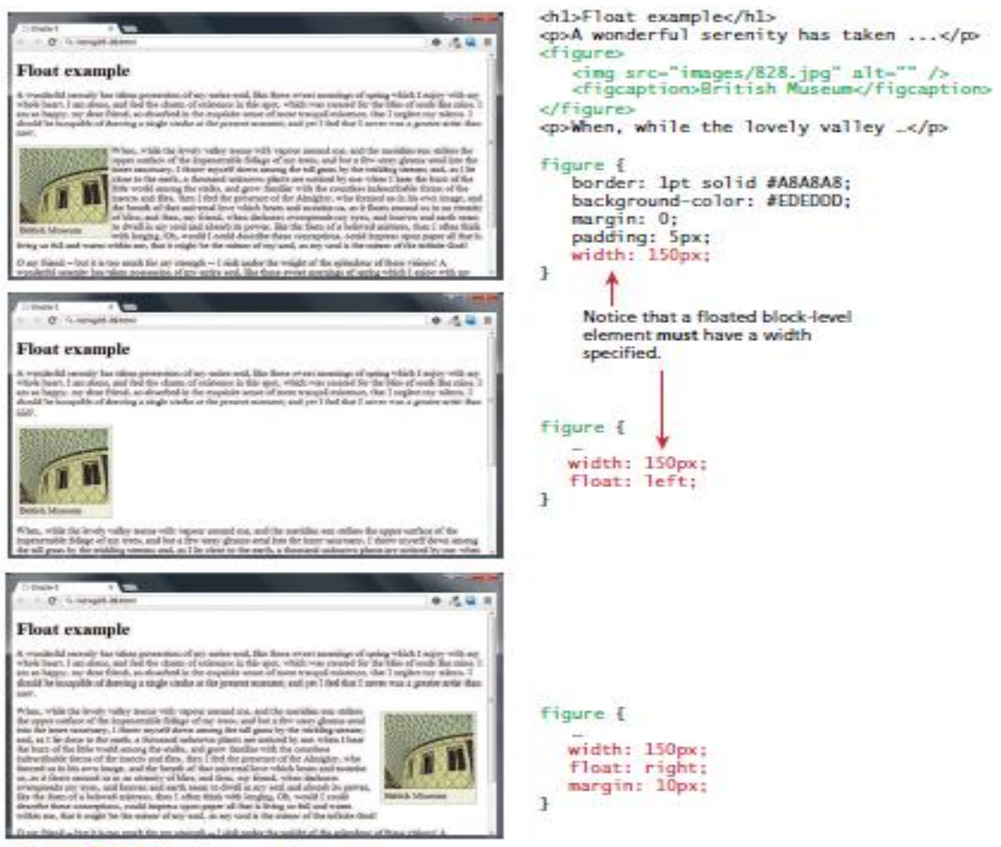
	border: 1pt solid #A8A8A8;
	background_color: #EDEDDD;
	padding: 5px;
	width: 150px;
	position: absolute;
	top: 150px;
	left: 200px;
	}
	figcaption {
	background_color: #EDEDDD;
	padding: 5px;
	position: absolute;
	top: 150px;
	left: 200px;
	}tion
	absolute The element is removed from normal flow and positioned in relation to its
	nearest positioned ancestor.
	fixed The element is fixed in a specific position in the window even when the document is scrolled.
	relative The element is moved relative to where it would be in the normal flow.
	static The element is positioned according to the normal flow. This is the default.
	table 5.1 Position Values			

2 (a)	What does floating an element do in CSS? How do you float an element? **Floating Elements:** • It is possible to displace an element out of its position in the normal flow via the CSS float **property**. • An element can be floated to the left or floated to the right. • When an item is floated, it is moved all the way to the far left or far right of its containing block and the rest of the content is "re-flowed" around the floated element, as can be seen in Figure 5.9. • Notice that a floated block-level element must have a width specified; if you do not, then the width will be set to auto, which will mean it implicitly fills the entire width of the containing block, and there thus will be no room available to flow content around the floated item. Also note in the final example in Figure 5.9 that the margins on the floated	[06]	CO4	L2

element are respected by the content that surrounds the floated element.

Floating within a Container:

- It should be reiterated that a floated item moves to the left or right of its container (also called its **containing block**). In Figure 5.9, the containing block is the HTML document itself so the figure moves to the left or right of the browser window.


Float example
Notice that a floated block-level element must have a width specified.

```
<h1>Float example</h1>
<p>A wonderful serenity has taken ...</p>
<figure>
   <img src="images/828.jpg" alt="" />
   <figcaption>British Museum</figcaption>
</figure>
<p>When, while the lovely valley …</p>

figure {
   border: 1pt solid #A8A8A8;
   background-color: #EDEDDD;
   margin: 0;
   padding: 5px;
   width: 150px;
}

figure {
  …
  width: 150px;
  float: left;
}

figure {
  …
  width: 150px;
  float: right;
  margin: 10px;
}
```

FIGURE 5.9 Floating an element

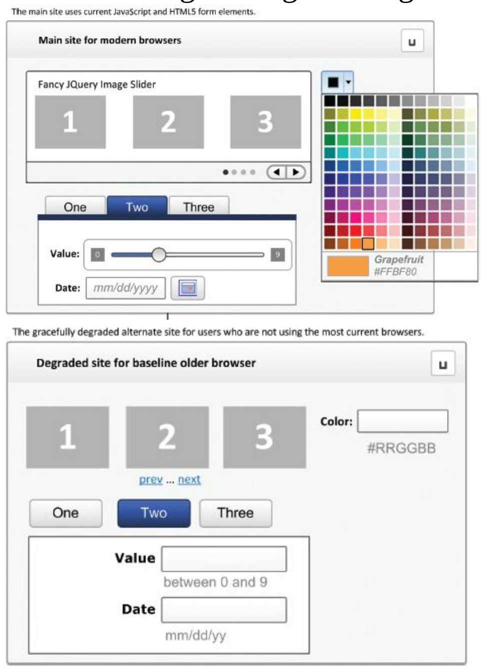
2 (b)	Write short notes on graceful degradation and progressive enhancement.	[04]	CO3	L2

Graceful degradation:

- With this strategy you develop your site for the abilities of current browsers.
- For those users who are not using current browsers, we need to provide an alternative site or pages for those using older browsers that lack the JS

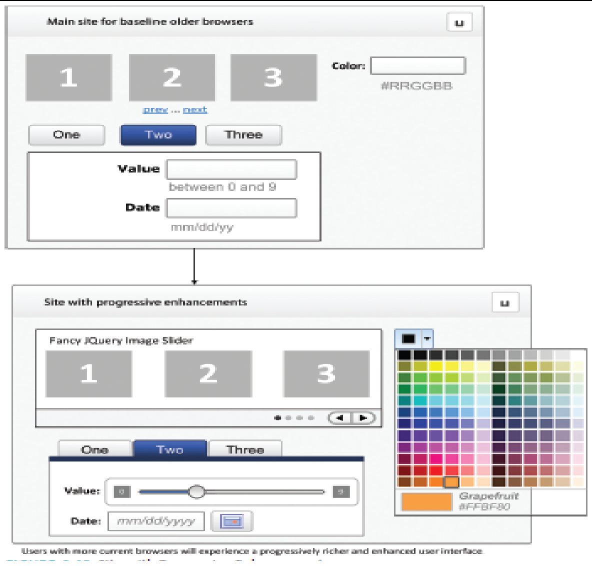
used on the main site.

- The idea here is that the site is “degraded” (i.e., loses capability) “gracefully” (i.e., without pop-up JavaScript error codes or without condescending messages telling users to upgrade their browsers).


The main site uses current JavaScript and HTML5 form elements.
Main site for modern browsers
Fancy JQuery Image Slider
1
2
3
One
Two
Three
Value:
Date:
mm/dd/yyyy
Grapefruit
#FFBF80
The gracefully degraded alternate site for users who are not using the most current browsers.
Degraded site for baseline older browser
1
2
3
prev ... next
Color:
#RRGGBB
One
Two
Three
Value
between 0 and 9
Date
mm/dd/yy

Progressive enhancement:

- The developer creates the site using CSS, JavaScript, and HTML features that are supported by all browsers of a certain age or newer.
- The developers can now “progressively” (i.e., for each browser) “enhance” (i.e., add functionality) to their site based on the capabilities of the users’ browsers.
- For instance, users using the current version of Opera and Chrome might see the fancy HTML5 color input form elements (since both support it at present), users using current versions of other browsers might see a jQuery plug-in that has similar functionality, while users of IE 7 might just see a simple text box.

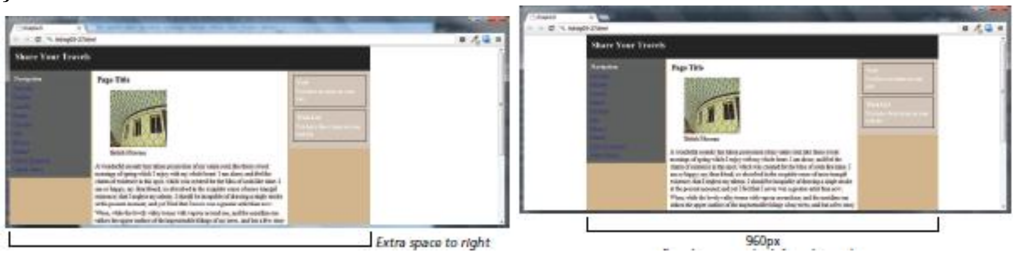
	 Main site for baseline older browsers 1 2 3 Color: #RRGGBB prev ... next One Two Three Value between 0 and 9 Date mm/dd/yy Site with progressive enhancements Fancy JQuery Image Slider 1 2 3 One Two Three Value: Date: mm/dd/yyyy Grapefruit #FFBF80 Users with more current browsers will experience a progressively richer and enhanced user interface			
3 (a)	Explain Grid systems in CSS with examples. • Grid systems make it easier to create multicolumn layouts. • There are many CSS grid systems; some of the most popular are A. Bootstrap (**twitter.github.com/bootstrap**), B. Blueprint (**www.blueprintcss.org**), and C. 960 (**960.gs**). • Print designers typically use grids as a way to achieve visual uniformity in a design. • In print design, the very first thing a designer may do is to construct, for instance, a 5- or 7- or 12-column grid in a page layout program like InDesign or Quark Xpress. • CSS frameworks provide similar grid features. The 960 framework uses either a 12- or 16-column grid. Bootstrap uses a 12-column grid. Blueprint uses a 24-column grid. • The grid is constructed using <div> elements with classes defined by the framework. • In bootstrap and 960, elements are laid out in rows; elements in a rows; elements in a row will span from 1 to 12 columns. • In 960 system, a row is terminated with < div class=”clear”></div>. • In bootstrap sysem, content must be placed within the <div class= row“> row container. <u>Using 960 grid</u> <head> <link rel="stylesheet" href="reset.css" /> <link rel="stylesheet" href="text.css" /> <link rel="stylesheet" href="960.css" /> </head> <body>	[06]	CO4	L2

	`<div class="container_12">`
	`<div class="grid_2">`
	left column
	`</div>`
	`<div class="grid_7">`
	main content
	`</div>`
	`<div class="grid_3">`
	right column
	`</div>`
	`<div class="clear"></div>`
	`</div>`
	`</body>`
	Using the Bootstrap grid
	`<head>`
	`<link href="bootstrap.css" rel="stylesheet">`
	`</head>`
	`<body>`
	`<div class="container">`
	`<div class="row">`
	`<div class="col-md-2">`
	left column
	`</div>`
	`<div class="col-md-7">`
	main content
	`</div>`
	`<div class="col-md-3">`
	right column
	`</div>`
	`</div>`
	`</div>`
	`</body>`
	
	• Both of these frameworks allow columns to be nested, making it quite easy to construct the most complex of layouts.
	• Bootstrap provides more than just a grid system.
	• It also has a wide variety of very useful additional styling classes such as classes for drop-down menus, fancy buttons and form elements and integration with a variety of JQuery plug-ins.			

3 (b)	Write short notes on CSS Layout. • One of the main problems faced by web designers is that the size of the screen used to view the page can vary quite a bit. • Some users will visit a site on a 21-inch wide screen monitor that can display 1920 × 1080 pixels (px); others will visit it on an older iPhone with a 3.5 screen and a resolution of 320 × 480 px.	[04]	CO2	L2

- Users with the large monitor might expect a site to take advantage of the extra size; users with the small monitor will expect the site to scale to the smaller size and still be usable.
- Satisfying both users can be difficult; the approach to take for one type of site content might not work as well with another site with different content.
- Basic models:
    A. Fixed Layout:
- In a **fixed layout**, the basic width of the design is set by the designer.
- A common width used is something in the 960 to 1000 pixel range, which fits nicely in the common desktop monitor resolution (1024 × 768).
- This content can then be positioned on the left or the center of the monitor.
- Fixed layouts are created using pixel units, typically with the entire content within a <div> container whose width property set to some width.

```
<body>
<div id="wrapper">
<header>
...
</header>
<div id="main">
...
</div>
<footer>
...
</footer>
</div>
</body>


div#wrapper {
width: 960px;
background_color: tan;
}
```



The advantage of a fixed layout is that

a) It is easier to produce and generally has a predictable visual result.
b) It is also optimized for typical desktop monitors.

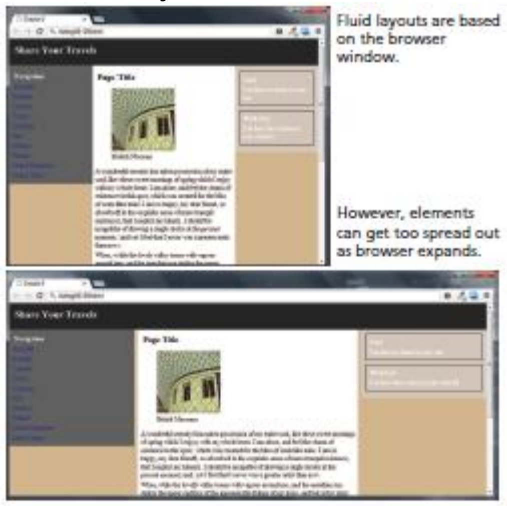
Fixed layouts have drawbacks:

c) For larger screens, there may be an excessive amount of blank space to the left and/or right of the content.

d) Much worse is when the browser window shrinks below the fixed width; the user will have to horizontally scroll to see all the content.

B. Liquid layout:

- In this approach, widths are not specified using pixels, but percentage values.
- Percentage values in CSS are a percentage of the current browser width, so a layout in which all widths are expressed as percentages should adapt to any browser size.


Fluid layouts are based on the browser window.
However, elements can get too spread out as browser expands.
Share Your Travels
Page Title

Advantage of a liquid layout

- It adapts to different browser sizes, so there is neither wasted white space nor any need for horizontal scrolling.

Disadvantage of a liquid layout

- Liquid layouts can be more difficult to create because some elements, such as images, have fixed pixel sizes.
- Another problem will be noticeable as the screen grows or shrinks dramatically, in that the line length (which is an important contributing factor to readability) may become too long or too short.

4 (a)	Write a javascript code that displays text “VTU BELGAVI” with increasing font size in the interval of 100 ms in blue color, when the font size reaches 50 pt it should stop.	[06]	CO3	L2

```
<!DOCTYPE HTML>
<html>
<head>
<title>program</title>
</head>
```

```
<body>
        <p id="demo"></p>
        <script>
                var var1 = setInterval(inTimer, 1000);
                var fs = 5;
                var ids = document.getElementById("demo");
                function inTimer() {
                        ids.innerHTML = 'TEXT
                        GROWING'; ids.setAttribute('style',
                        "font-size: " + fs + "px; color: red");
                        fs += 5;
                        if(fs >= 50 ){
                                clearInterval(var1);

                                var2 = setInterval(deTimer, 1000);
                        }
                }
                function deTimer() {
                        fs -= 5;
                        ids.innerHTML = 'TEXT
                        SHRINKING'; ids.setAttribute('style',
                        "font-size: " + fs + "px; color: blue");
                        if(fs === 5 ){
                                clearInterval(var2);
                        }
                }
        </script>
</body>
</html>
```

4 (b)	Write a PHP program to greet the user based on time.	[04]	CO3	L2

```
<html>

<head>

<title>program</title>

</head>

<body action=”call.php” method=”POST”>

Enter the user name:

<input type=”text” name=”na”>

</body>
```

	`</html>`
	
	Call.php
	`<?php`
	`$name=$_POST[‘name’];`
	`$time=date(‘H’);`
	`If($time>0 && &time<10)`
	`Echo “Good morning”.$name;`
	`Else if($time>10 && &time<13)`
	`Echo “Good afternoon”.$name;`
	`Else if($time>13 && &time<18)`
	`Echo “Good evening”.$name;`
	`else`
	`Echo “Good night”.$name;`
	`?>`			

5 (a)	Briefly explain function in PHP with examples. **Functions** In PHP there are two types of function: 1. User-defined functions 2. Built-in functions. A **user-defined function** is one that you the programmer define. A **built-infunction** is one of the functions that come with the PHP environment **<u>Function Syntax</u>** To create a new function you must think of a name for it, and consider what it willdo. Functions can return values to the caller, or not return a value. They can be set upto take or not take parameters.	[10]	CO4	L2

```
function getNiceTime() {
return date("H:i:s");
}
```

The definition of a function to return the current time as a string

```
function outputFooterMenu() {
echo '<div id="footer">';
echo '<a href=#>Home</a> | <a href=#>Products</a> | ';
echo '<a href=#>About us</a> | <a href=#>Contact us</a>';
echo '</div>';
}
```

The definition of a function without a return value

Calling a Function

To call a function you must use its name with the () brackets. Since getNiceTime()returns a string, you can assign that return value to a variable, or echo that returnvalue directly, as shown below.

```
$output = getNiceTime();
echo getNiceTime();
```

If the function doesn't return a value, you can just call the function:

```
outputFooterMenu();
```

Parameters

It is more common to define functions with parameters, since functions are more powerful and reusable when their output depends on the input they get. **Parameters** are the mechanism by which values are passed into functions, and there are some complexities that allow us to have multiple parameters, default values, and to pass objects by reference instead of value.

To define a function with parameters, you must decide how many parameters you want to pass in, and in what order they will be passed. Each parameter must be named.

```
function getNiceTime($showSeconds) {

if ($showSeconds==true)

return date("H:i:s");

else

return date("H:i");

}
```

A function to return the current time as a string with an integer parameterThus to call our function, you can now do it in two ways:

```
echo getNiceTime(1); // this will print seconds

echo getNiceTime(0); // will not print seconds
```

In fact any nonzero number passed in to the function will be interpreted as true since the parameter is not type specific.

Parameter Default Values

```
function getNiceTime($showSeconds=1){

if ($showSeconds==true)

return date("H:i:s");

else

return date("H:i");

}
```

A function to return the current time with a parameter that includes a default. If you do include a value in your function call, the default will beoverridden by whatever that value was.

Passing Parameters by Reference

By default, arguments passed to functions are **passed by value** in PHP.

```
function changeParameter($arg) {

$arg += 300;
echo "<br/>arg=" . $arg;

}

$initial = 15;

echo "<br/>initial=" . $initial; // output: initial=15

changeParameter($initial); // output: arg=315

echo "<br/>initial=" . $initial; // output: initial=15
```

Passing a parameter by value

The mechanism in PHP to specify that a parameter is passed byreference is to add an ampersand (&) symbol next to the parameter name in thefunction declaration.

```
function changeParameter(&$arg) {

$arg += 300;

echo "<br/>arg=". $arg;

}

$initial = 15;

echo "<br/>initial=" . $initial; // output: initial=15

changeParameter($initial); // output: arg=315

echo "<br/>initial=" . $initial; // output: initial=315
```

Passing a parameter by reference

Variable Scope within Functions

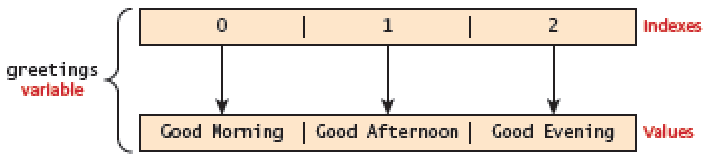
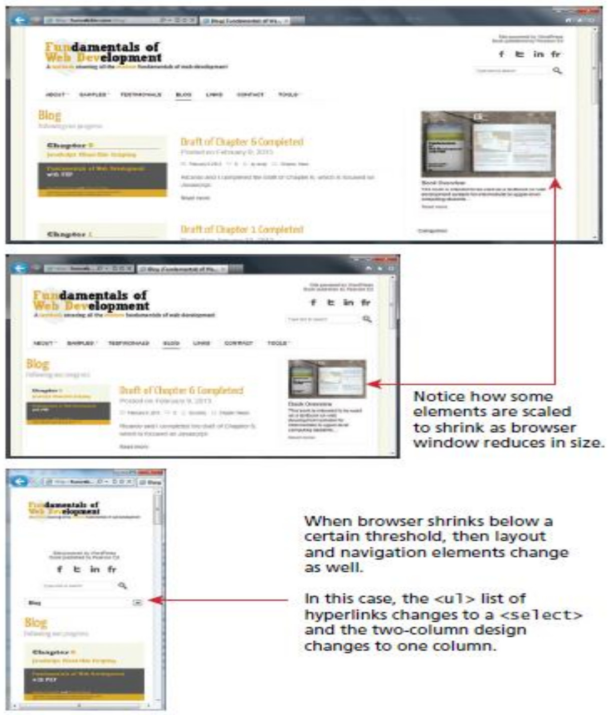
It will come as no surprise that all variables defined within a function (such asparameter variables) have **function scope**, meaning that they are only accessiblewithin the function.

```
$count= 56;

function testScope() {
```

	echo $count; *// outputs 0 or generates run-time warning/error*
	}
	testScope();
	echo $count; *// outputs 56*
	While variables defined in the main script are said to have **global scope**,
	$count= 56;
	function testScope() {
	global $count;
	echo $count; *// outputs 56*
	}
	testScope();
	echo $count; *// outputs 56*
	Using the global keyword			

6 (a)	Discuss the advantages and disadvantages of client side scripting. **There are many advantages of client-side scripting:** ■ Processing can be offloaded from the server to client machines, thereby reducing the load on the server. ■ The browser can respond more rapidly to user events than a request to a remote server ever could, which improves the user experience. JavaScript can interact with the downloaded HTML in a way that the server cannot, creating a user experience more like desktop software than simple HTML ever could. **The disadvantages of client-side scripting are mostly related to how programmers use JavaScript in their applications. Some of these include:** ■ There is no guarantee that the client has JavaScript enabled, meaning any required functionality must be housed on the server, despite the	[05]	CO2	L2

	possibility that it could be offloaded. ■ The idiosyncrasies between various browsers and operating systems make it difficult to test for all potential client configurations. What works in one browser, may generate an error in another. ■ JavaScript-heavy web applications can be complicated to debug and maintain. JavaScript has often been used through inline HTML hooks that are embedded into the HTML of a web page. Although this technique has been used for years, it has the distinct disadvantage of blending HTML and JavaScript together, which decreases code readability, and increases the difficulty of web development.			
6 (b)	Explain arrays in JavaScript with examples. **Arrays** • Arrays are one of the most used data structures, and they have been included in JavaScript as well. Objects can be created using the new syntax and calling the object constructor. The following code creates a new, empty array named greetings: *var greetings = new Array();* • To initialize the array with values, the variable declaration would look like the following: *var greetings = new Array("Good Morning", "Good Afternoon");* • or, using the square bracket notation: *var greetings = ["Good Morning", "Good Afternoon"];* **Accessing and Traversing an Array** • To access an element in the array you use the familiar square bracket notation with the index to be accessed can be mentioned inside the brackets. *alert (greetings[0]);* • Traverse through the items sequentially. The following for loop quickly loops through an array, accessing the ith element each time using the Array object's length property to determine the maximum valid index. It will alert "Good Morning" and "Good Afternoon" to the user. *for (var i = 0; i < greetings.length; i++){* *alert(greetings[i]);* *}*	[05]	CO2	L2

	 0 1 2 Indexes greetings variable Good Morning Good Afternoon Good Evening Values Figure illustrates an array with indexes and the corresponding values. **Modifying an Array** • To add an item to an existing array, you can use the push method. *greetings.push("Good Evening");* • The pop method can be used to remove an item from the back of an array. • Additional methods that modify arrays include concat(), slice(), join(), reverse(), shift(), and sort().			
7 (a)	Briefly explain responsive design in CSS with examples. • The page "responds" to changes in the browser size that go beyond the width scaling of a liquid layout. • One of the problems of a liquid layout is that images and horizontal navigation elements tend to take up a fixed size, and when the browser window shrinks to the size of a mobile browser, liquid layouts can become unusable. • In a responsive layout, images will be scaled down and navigation elements will be replaced as the browser shrinks.  Notice how some elements are scaled to shrink as browser window reduces in size. When browser shrinks below a certain threshold, then layout and navigation elements change as well. In this case, the <ul> list of hyperlinks changes to a <select> and the two-column design changes to one column. There are four key components that make responsive design work. They are:	[06]	CO2	L2

1. Liquid layouts
2. Scaling images to the viewport size
3. Setting viewports via the <meta> tag
4. Customizing the CSS for different viewports using media queries

Responsive designs begin with a liquid layout, that is, one in which most elements have their widths specified as percentages.

```
img
{
   max-width: 100%;
}
```

Setting viewports:

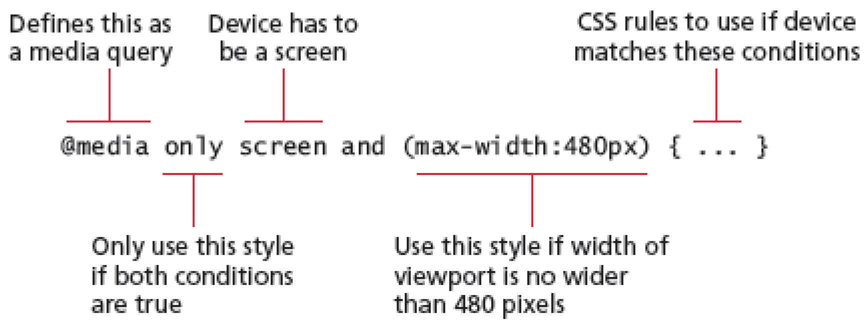
- A key technique in creating responsive layouts makes use of the ability of current mobile browsers to shrink or grow the web page to fit the width of the screen.
- The way this works is the mobile browser renders the page on a canvas called the **viewport**.

```
<html>
<head>
<meta name="viewport" content="width=device-width" />
```

- Device-width sets the pixels wide as the device screen width.
- If the device has a screen that is 320px wide, the viewport width will be 320px.

Media Query

- The other key component of responsive designs is **CSS media queries**.
- A media query is a way to apply style rules based on the medium that is displaying the file.



```
@media only screen and (max-width:480px) { ... }
```

Defines this as a media query
Device has to be a screen
CSS rules to use if device matches these conditions
Only use this style if both conditions are true
Use this style if width of viewport is no wider than 480 pixels

- Figure illustrates the syntax of a typical media query.
- These queries are Boolean expressions and can be added to your CSS files or to the <link> element to conditionally use a different external CSS file based on the capabilities of the device.

Feature	Description
`width`	Width of the viewport
`height`	Height of the viewport
`device-width`	Width of the device
`device-height`	Height of the device
`orientation`	Whether the device is portrait or landscape
`color`	The number of bits per color

- Table is a partial list of the browser features you can examine with media queries. Many of these features have min- and max- versions.
- Contemporary responsive sites will typically provide CSS rules for phone displays first, then tablets, then desktop monitors, an approach called **progressive enhancement**, in which a design is adapted to progressively more advance devices.

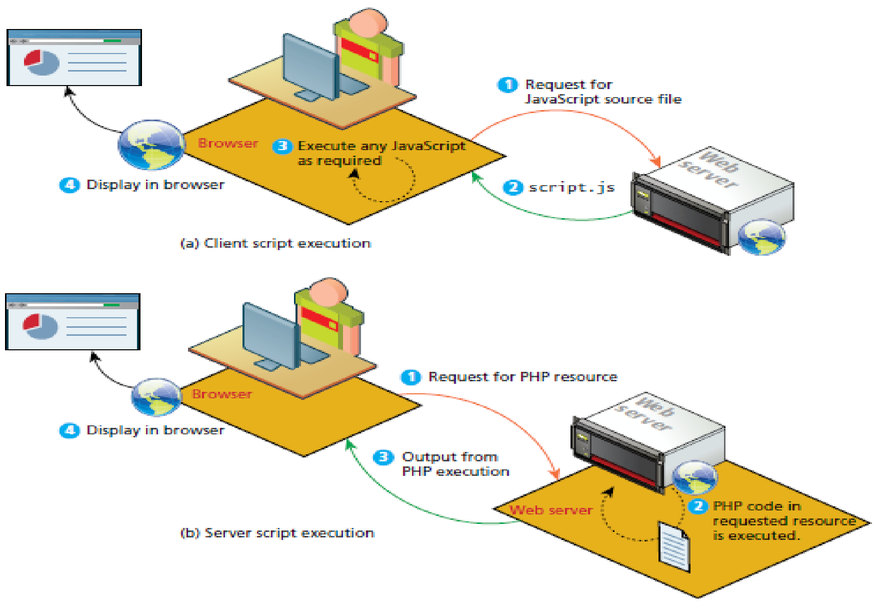
styles.css

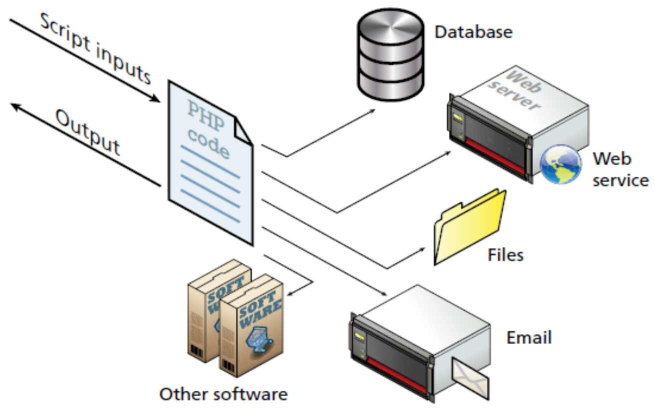
```
/* rules for phones */
@media only screen and (max-width:480px)
{
  #slider-image { max-width: 100%; }
  #flash-ad { display: none; }
    ...
}


/* CSS rules for tablets */
@media only screen and (min-width: 481px)
    and (max-width: 768px)
{
    ...
}



/* CSS rules for desktops */
@media only screen and (min-width: 769px)
{
    ...
}
```

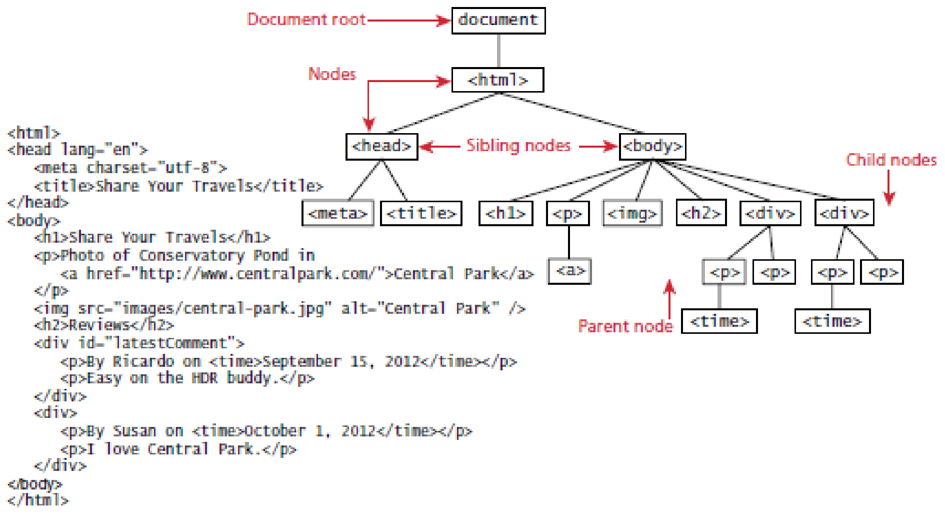
- Instead of having all the rules in a single file, put them in separate files and add media queries to <link> elements.

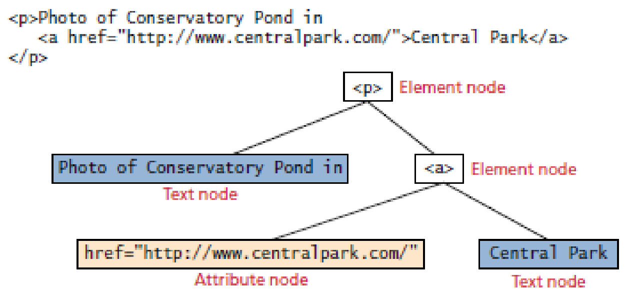
```
<link rel="stylesheet" href="mobile.css"  media="screen and (max-width:480px)" />
<link rel="stylesheet" href="tablet.css"  media="screen and (min-width:481px)
    and (max-width:768px)" />
<link rel="stylesheet" href="desktop.css" media="screen and (min-width:769px)" />

<!--[if lt IE 9]>
<link rel="stylesheet" media="all" href="style-ie.css"/>
<![endif]-->
```

Handles Internet Explorer 8 and earlier using IE conditional comments.

7 (b)	Explain client side scripting with neat diagram **Client and Server Scripts:** The fundamental difference between client and server scripts is that in a client-side script the code is executed on the client browser, whereas in a server-side script, it is executed on the web server. server-side source code remains hidden from the client as it is processed on the server. The clients never get to see the code, just the HTML output from the script.  1 Request for JavaScript source file Browser 3 Execute any JavaScript as required 4 Display in browser 2 script.js Web server (a) Client script execution 1 Request for PHP resource Browser 4 Display in browser 3 Output from PHP execution Web server 2 PHP code in requested resource is executed. (b) Server script execution FIGURE 8.1 Comparison of (a) client script execution and (b) server script execution **Server-Side Script Resources** A server-side script can access any resources made available to it by the server. These resources can be categorized as data storage resources, web services, and software applications, as can be seen in Figure	[04]	CO4	L2

	 Script inputs Output PHP code Database Web server Web service Files Email SOFT WARE Other software FIGURE 8.2 Server scripts have access to many resources. The most commonly used resource is data storage, often in the form of a connection to a database management system. A database management system (DBMS) is a software system for storing, retrieving, and organizing large amounts of data.			

8 (a)	Elaborately explain DOM in JavaScript. Give examples for verbose technique.
	
	• There needs to be some way of programmatically accessing the elements and attributes within the HTML. This is accomplished through a programming interface (API) called the **Document Object Model (DOM)**.
	
	 Document root
	document
	Nodes
	<html>
	<head>
	Sibling nodes
	<body>
	Child nodes
	<meta>
	<title>
	<h1>
	<p>
	<img>
	<h2>
	<div>
	<div>
	<a>
	<p>
	<p>
	<p>
	<p>
	Parent node
	<time>
	<time>
	`<html>`
	`<head lang="en">`
	`   <meta charset="utf-8">`
	`   <title>Share Your Travels</title>`
	`</head>`
	`<body>`
	`   <h1>Share Your Travels</h1>`
	`   <p>Photo of Conservatory Pond in`
	`      <a href="http://www.centralpark.com/">Central Park</a>`
	`   </p>`
	`   <img src="images/central-park.jpg" alt="Central Park" />`
	`   <h2>Reviews</h2>`
	`   <div id="latestComment">`
	`      <p>By Ricardo on <time>September 15, 2012</time></p>`
	`      <p>Easy on the HDR buddy.</p>`
	`   </div>`
	`   <div>`
	`      <p>By Susan on <time>October 1, 2012</time></p>`
	`      <p>I love Central Park.</p>`
	`   </div>`
	`</body>`
	`</html>`
	
	Nodes:
	In the DOM, each element within the HTML document is called a **node**. If the DOM is a tree, then each node is an individual branch.
		[10]	CO4	L2


```
<p>Photo of Conservatory Pond in
   <a href="http://www.centralpark.com/">Central Park</a>
</p>
```

Thus, most of the tasks that we typically perform in JavaScript involve finding a node, and then accessing or modifying it via those properties and methods.

Property	Description
`attributes`	Collection of node attributes
`childNodes`	A `NodeList` of child nodes for this node
`firstChild`	First child node of this node
`lastChild`	Last child of this node
`nextSibling`	Next sibling node for this node
`nodeName`	Name of the node
`nodeType`	Type of the node
`nodeValue`	Value of the node
`parentNode`	Parent node for this node
`previousSibling`	Previous sibling node for this node.

Document Object:

The DOM document object is the root JavaScript object representing the entire HTML document.

It is globally accessible as document.

The attributes of this object include some information about the page including

- doctype and
- inputEncoding

Example:

var a = document.doctype.name

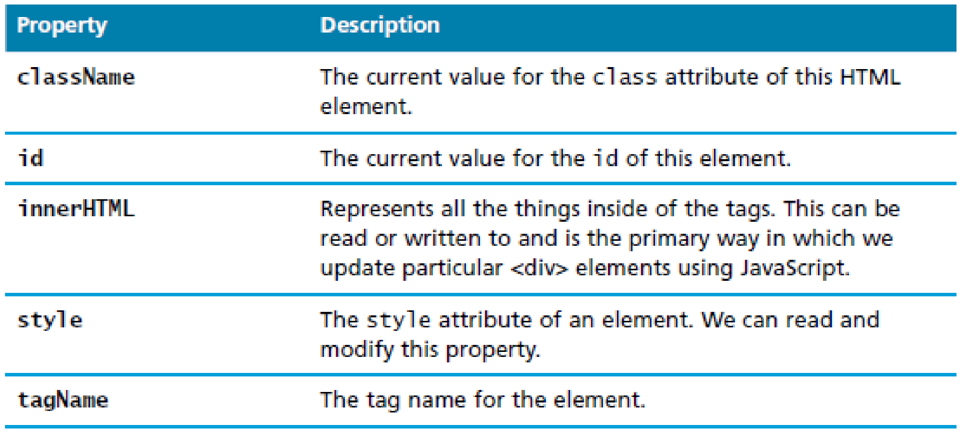
var b= document.inputEncoding

Method	Description
`createAttribute()`	Creates an attribute node
`createElement()`	Creates an element node
`createTextNode()`	Creates a text node
`getElementById(id)`	Returns the element node whose `id` attribute matches the passed `id` parameter
`getElementsByTagName(name)`	Returns a NodeList of elements whose tag name matches the passed `name` parameter

```
<div id= “latest”>
   <p>CMRIT</p>
</div>

var abc= document.getElementById(“latest”);
var list= document.getElementByTagName(“div”);
```

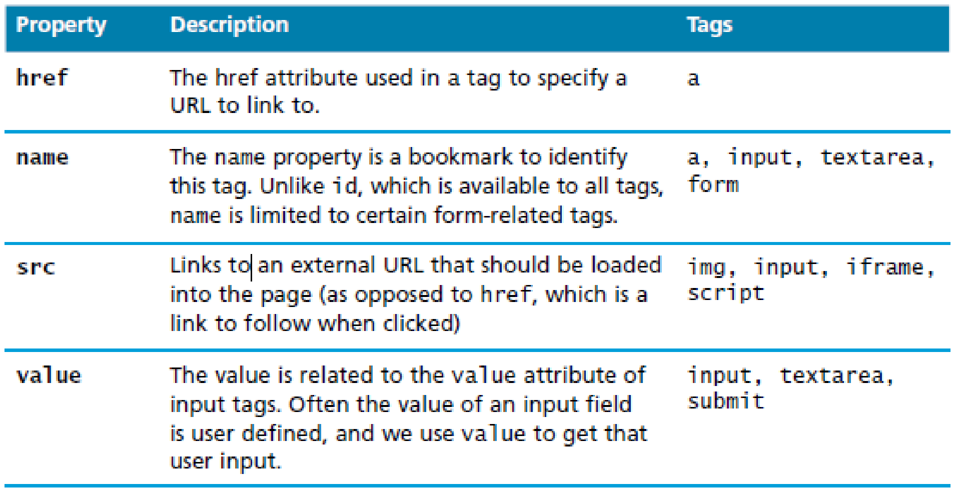
Element Node Object:

- The type of object returned by the method document.getElementById() is an element node object.
- Since ID’s must be unique in an HTML document, getElementById() returns a single node.



Property	Description
`className`	The current value for the `class` attribute of this HTML element.
`id`	The current value for the `id` of this element.
`innerHTML`	Represents all the things inside of the tags. This can be read or written to and is the primary way in which we update particular <div> elements using JavaScript.
`style`	The `style` attribute of an element. We can read and modify this property.
`tagName`	The tag name for the element.

HTML DOM Element properties for certain tags.



Property	Description	Tags
`href`	The href attribute used in a tag to specify a URL to link to.	a
`name`	The name property is a bookmark to identify this tag. Unlike `id`, which is available to all tags, name is limited to certain form-related tags.	`a, input, textarea, form`
`src`	Links to an external URL that should be loaded into the page (as opposed to `href`, which is a link to follow when clicked)	`img, input, iframe, script`
`value`	The value is related to the `value` attribute of input tags. Often the value of an input field is user defined, and we use `value` to get that user input.	`input, textarea, submit`

Modifying a DOM element:

- Document.write() method is used to create output to the HTML page from JS.
- Example:
  Changing the HTML using innerHTML.

```
            var lat= document.getElementById(“latest”);
            var old = lat.innerHTML;
            lat.innerHTML = old+ “<p> updated with JS</p>”
```

Output:
Cmrit
Updated with JS

A more Verbose Technique
DOM functions createTextNode(), removeChild(), and appendChild() allow us to modify an element in a more rigorous way.

```
<body>
  <ul id=”list”>
     <li>Coffee</li>
    <li>Tea</li>
</ul>
<button onClick=”fun()”>Click</button>
<script>
//to add a element node and textnode
function fun(){
     var node=document.createElement(“LI”);
     var textnode=document.createTextNode(“Water”);
     node.appendChild(textnode);
    document.getElementById(“list”).appendChild(node);
}
</script>

//to remove a child node
<script>
function fun(){
     var list1= document.getElementById(“list”);
      list1.removeChild(list.childNode[0]);
}
</script>
```