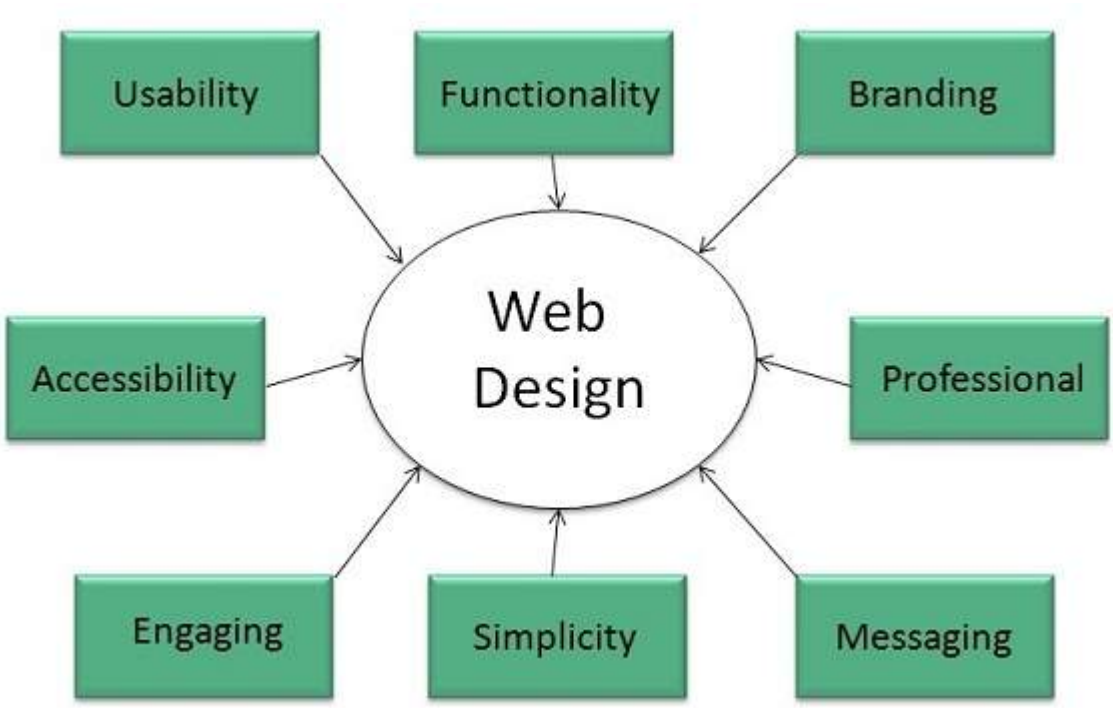
WEBSITE DESIGNING

Web designing has direct link to visual aspect of a web site. Effective web design is necessary to communicate ideas effectively.



> *Web desinging is subset of web development. However these terms are used interchangeably.*

Key Points

Design Plan should include the following:

- Details about information architecture.
- Planned structure of site.
- A site map of pages

Wireframe

Wireframe refers to a visual guide to appearace of web pages. It helps to define structre of web site, linking between web pages and layout of visual elements.

Following things are included in a wireframe:

- Boxes of primary graphical elements
- Placement of headlines and sub headings
- Simple layout structure
- Calls to action
- Text blocks

> *Wireframe can be created using program like Visio but you can also use a pen and*

> *paper.*

Web Designing Tools

Here is the list of tools that can be used to make effective web designs:

Coda 2
Coda 2 is a powerful web development & designing tool, comes with better user interface, text editing, file management, clips, sites, design and better Mysql support.**OmniGraffle**
OmmniGraffle is mainly used for wireframing. The downside of this tool is that It doesnot have interactive prototyping and It is available only for Mac.**Pen and Paper**
Pen and paper can be used to draw the appeance of the how the web site will look like.**Vim**
Vim is great web desiging tool.It supports full customizable auto-intending of code, multiple buffers for storing cut/copied code, and recording of actions for automated repetition.

S.N.	Tool Description
1.	**Photoshop CC** This is a great web designing tool provided by Adobe. The latest Photoshop CC 2014 supports many new features such as smart objects, layer comps, smart guides, Typekit integration, font search, and workflow enhancements.
2.	**Illustrator CC** Illustrator CC is also a web designing tool comes with powerful features like AutoCad libraries, white overprint, fill and stroke proxy swap for text, automatic corner generation, unembed images and touch type tools etc.
3.	
4.	
5.	**Sublime Text** Sublime Text is a source code editor with Python application programming interface. It's functionality can be extended using plugins.
6.	
7.	
8.	**Imageoptim** It is basically used for optimizing images on a website in order to load them faster by finding best compression parameters and by removing unnecessary comments.
9.	**Sketch 3** Sketch 3 is a web desiging tool developed specifically for designing interfaces, websites, icons etc.
10.	**Heroku** It is also a great web development tool which supports Ruby, Node.js, Python, java and PHP.
11.	**Axure** It supports prototyping, documentation, and wireframing tools for making interactive website design.
12.	**Hype 2** The Hype 2 offers: Easiest way to Animate & add interactivity, Hardness the power of HTML5, Mobile responsiveness, and WYSIWYG features.
13.	**Image Alpha** This tool helps to reduce file sizes of 24-bit PNG files. It does so by applying lossy compression and convert it to PNG8+alpha format which more efficient.

14.	**Hammer** This tool is suitable for non programmers and good only for small projects.
15.	**JPEGmini Lite** It is an image optimizing tool and supports photos in any resolution up to 28 Megapixels.
16.	**BugHerd** This tool helps to see how the projects is going and what everyone is working on. It also helps to identify issues in development.

Web Page Anatomy

A web site includes the following components:

Containing Block

Container can be in the form of page's body tag, an all containing div tag. Without container there would be no place to put the contents of a web page.

Logo

Logo refers to the identity of a website and is used across a company's various forms of marketing such as business cards, letterhead, brouchers and so on.

Naviagation

The site's **navigation system** should be easy to find and use. Oftenly the anvigation is placed rigth at the top of the page.

Content

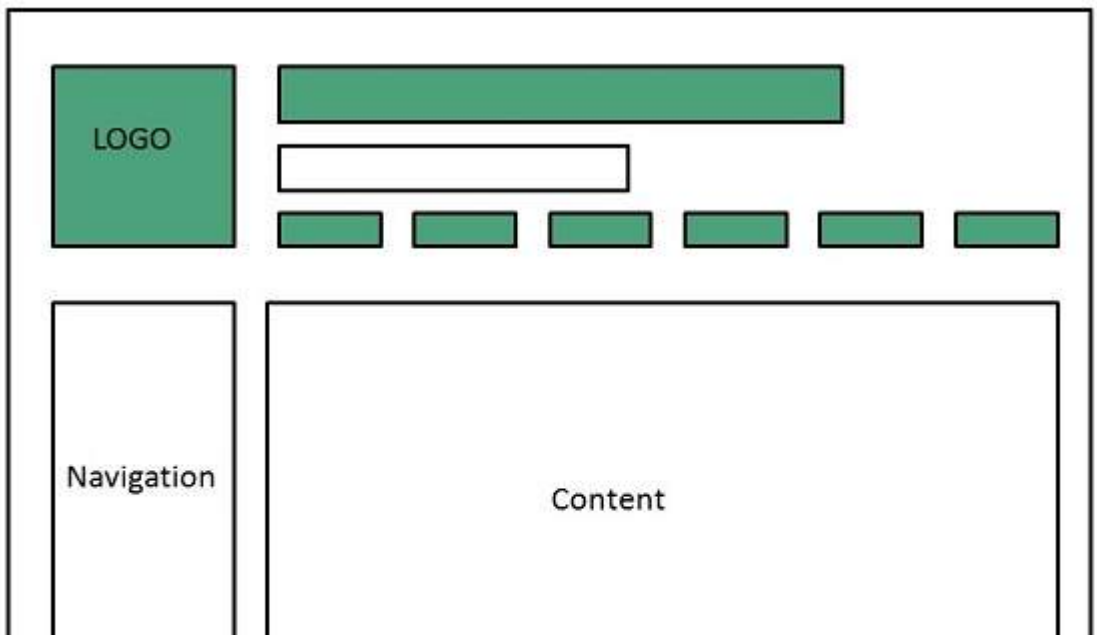
The content on a web site should be relevant to the purpose of the web site.

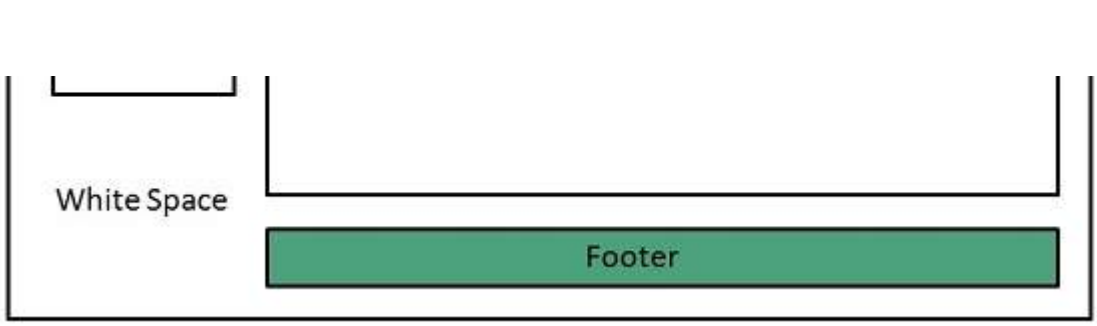
Footer

Footer is located at the bottom of the page. It usually contains copyright, contract and legal information as well as few links to the main sections of the site.

Whitespace

It is also called as **negative space** and refers to any area of page that is not covered by type or illustrations.





Web design Mistakes

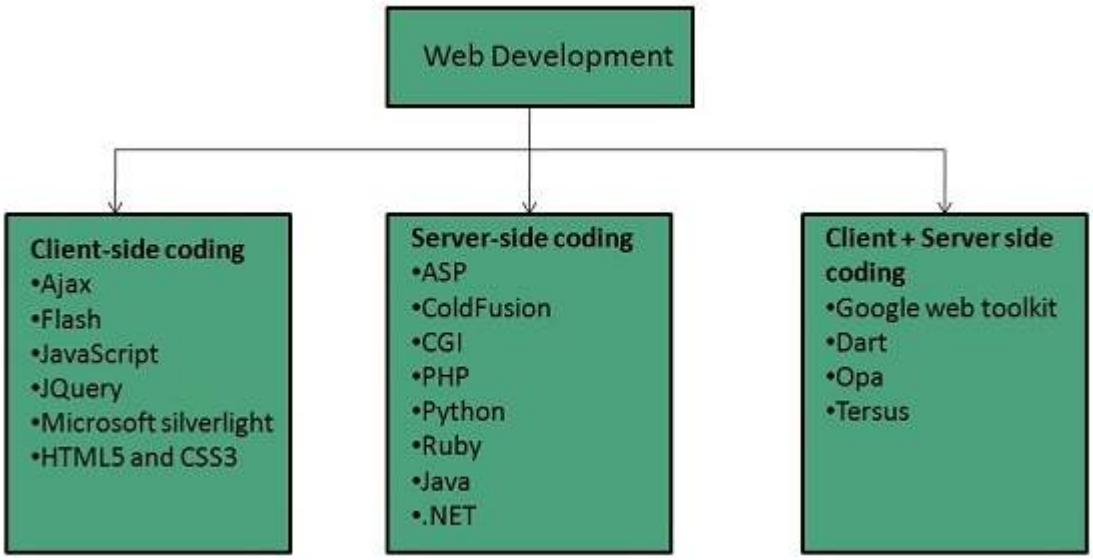
One should be aware of the following common mistakes should always keep in mind:

- Website not working in any other browser other internet explorer.
- Using cutting edge technology for no good reason
- Sound or video that starts automatically
- Hidden or disguised navigation
- 100% flash content.

WEBSITE DEVELOPMENT

Web development

Web development refers to building website and deploying on the web. Web development requires use of scripting languages both at the server end as well as at client end.

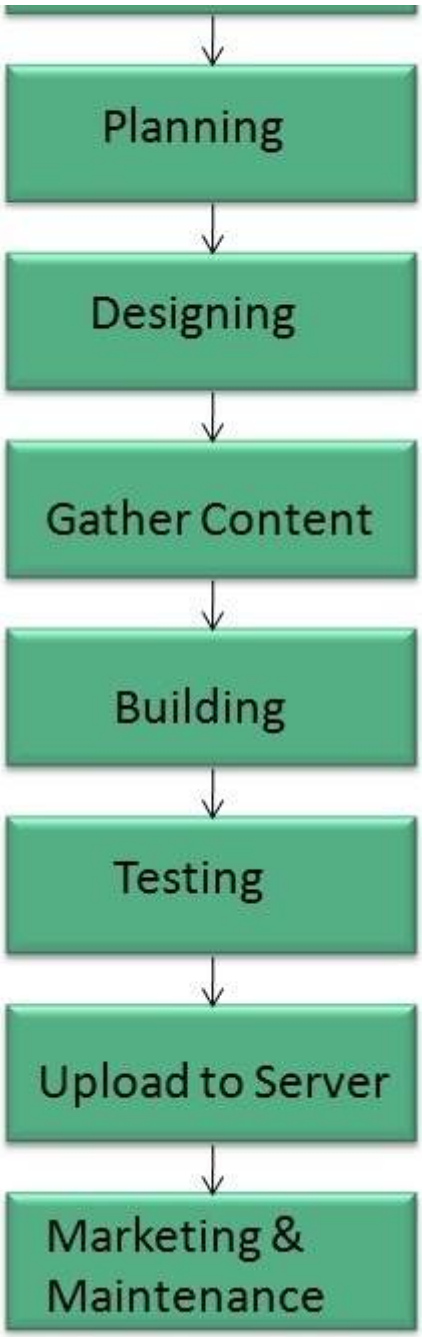


Before developing a web site once should keep several aspects in mind like:

- What to put on the web site?
- Who will host it?
- How to make it interactive?
- How to code it?
- How to create search engine friendly web site?
- How to secure the source code frequently?
- Will the web site design display well in different browsers?
- Will the navigation menus be easy to use?
- Will the web site loads quickly?
- How easily will the site pages print?
- How easily will visitors find important details specific to the web site?
- How effectively the style sheets be used on your web sites?

Web Development Process

Web development process includes all the steps that are good to take to build an attractive, effective and responsive website. These steps are shown in the following diagram:



Web development tools

Web development tools helps the developer to test and debug the web sites. Now a days the web development tooll come with the web browsers as add-ons. All web browsers have built in tools for this purpose.

Thsese tools allow the web developer to use HTML, CSS and JavaScript etc.. These are accessed by hovering over an item on a web page and selecting the “Inspect Element” from the context menu.

Featues

Following are the common featuers that every web development tool exhibits:

HTML and the DOM

HTML and DOM viewer allows you to see the DOM as it was rendered. It also allows to make changes to HTML and DOM and see the changes reflected in the page after the change is made.

Web Page Assests, Resources, and Network Information

Web development tools also helps to inspect the resources that are loaded and available on the web page.

Profiing and Auditing

Profiling refers to get information about the performance of a web page or web application and **Auditing** provides developers suggestions, after analyzing a page, for optimizations to decerease page load time and increase responsiveness.

Skills Required

For being a successful web developer, one should possess the following skills:

- Understanding of client and server side scripting.
- Creating, editing and modifying templates for a CMS or web development framework.
- Testing cross browser inconsistencies.
- Conducting observational user testing.
- Testing for compliance to specified standards such as accessibility standards in the client region.
- Programming interaction with javaScript, PHP, and Jquery etc.

HTML

Introduction

HTML stands for **Hyper Text Markup Language.** It is a formatting language used to define the appearance and contents of a web page. It allows us to organize text, graphics, audio, and video on a web page.

Key Points:

- The word Hypertext refers to the text which acts as a link.
- The word markup refers to the symbols that are used to define structure of the text. The markup symbols tells the browser how to display the text and are often called tags.
- The word Language refers to the syntax that is similar to any other language.

> *HTML was created by **Tim Berners-Lee** at **CERN.***

HTML Versions

The following table shows the various versions of HTML:

Version	Year
HTML 1.0	1991
HTML 2.0	1995
HTML 3.2	1997
HTML 4.0	1999
XHTML	2000
HTML5	2012

HTML Tags

Tag is a command that tells the web browser how to display the text, audio, graphics or video on a web page.

Key Points:

- Tags are indicated with pair of angle brackets.
- They start with a less than < character and end with a greater than > character.
- The tag name is specified between the angle brackets.
- Most of the tags usually occur in pair: the start tag and the closing tag.
- The start tag is simply the tag name is enclosed in angle bracket whereas the closing tag is specified including a forward slash /.
- Some tags are the empty i.e. they don’t have the closing tag.
- Tags are not case sensitive.

- The starting and closing tag name must be the same. For example hello is invalid as both are different.
- If you don't specify the angle brackets <> for a tag, the browser will treat the tag name as a simple text.
- The tag can also have attributes to provide additional information about the tag to the browser.

Basic tags

The following table shows the Basic HTML tags that define the basic web page:

Tag	Description
<html> </html>	Specifies the document as a web page.
<head> </head>	Specifies the descriptive information about the web documents.
<title> </title>	Specifies the title of the web page.
<body> </body>	Specifies the body of a web document.

The following code shows how to use basic tags.

```
<html>
      <head> Heading goes here…</head>
       <title> Title goes here…</title>
       <body> Body goes here…</body>
</html>
```

Formatting Tags

The following table shows the HTML tags used for formatting the text:

Tag	Description
	Specifies the text as bold. Eg. **this is bold text**
	It is a phrase text. It specifies the emphasized text. Eg. *Emphasized text*
	It is a phrase tag. It specifies an important text. Eg. **this is strong text**
	The content of italic tag is displayed in italic. Eg. *Italic text*
	Specifies the subscripted text. Eg. X_1
	Defines the superscripted text. Eg. X^2
<ins> </ins>	Specifies the inserted text. Eg. The price of pen is now ~~20~~15.
<del> </del>	Specifies the deleted text. Eg. The price of pen is now ~~20~~15.
<mark> </mark>	Specifies the marked text. Eg. It is raining

Table Tags

Following table describe the commonaly used table tags:

Tag	Description
<table> </table>	Specifies a table.
<tr> </tr>	Specifies a row in the table.
<th> </th>	Specifies header cell in the table.
<td> </td>	Specifies the data in an cell of the table.
<caption> </caption>	Specifies the table caption.
<colgroup> </colgroup>	Specifies a group of columns in a table for formatting.

List tags

Following table describe the commonaly used list tags:

Tag	Description
	Specifies an unordered list.
	Specifies an ordered list.
	Specifies a list item.
<dl> </dl>	Specifies a description list.
<dt> </dt>	Specifies the term in a description list.
<dd> </dd>	Specifies description of term in a description list.

Frames

Frames help us to divide the browser's window into multiple rectangular regions. Each region contains separate html web page and each of them work independently.

> *A set of frames in the entire browser is known as frameset. It tells the browser how to divide browser window into frames and the web pages that each has to load.*

The following table describes the various tags used for creating frames:

Tag	Description
<frameset> </frameset>	It is replacement of the <body> tag. It doesn't contain the tags that are normally used in <body> element; instead it contains the <frame> element used to add each frame.
<frame> </frame>	Specifies the content of different frames in a web page.
<base> </base>	It is used to set the default target frame in any page that contains links whose contents are displayed in another frame.

Forms

Forms are used to input the values. These values are sent to the server for processing. Forms uses

input elements such as text fields, check boxes, radio buttons, lists, submit buttons etc. to enter the data into it.

The following table describes the commonly used tags while creating a form:

Tag	Description
<form> </form>	It is used to create HTML form.
<input> </input>	Specifies the input field.
<textarea> </textarea>	Specifies a text area control that allows to enter multi-line text.
<label> </label>	Specifies the label for an input element.

Introduction

CSS is acronym of **Cascading Style Sheets.** It helps to define the presentation of HTML elements as a separate file known as CSS file having **.css** extension.

CSS helps to change formatting of any HTML element by just making changes at one place. All changes made would be reflected automatically to all of the web pages of the website in which that element appeared.

CSS Rules

CSS Rules are the styles that we have to create in order to create style sheets. These rules define appearance of associated HTML element. The general form of CSS syntax is as follows:

```
Selector {property: value;}
```

Key Points

- Selector is HTML element to which CSS rule is applied.
- Property specifies the attribute that you want to change corresponding to the selector.
- Property can take specified value.
- Property and Value are separated by a colon : .
- Each declaration is separated by semi colon ; .

Following are examples of CSS rules:

```
P { color : red;}

h1 (color : green; font-style : italic }

body { color : cyan; font-family : Arial; font- style : 16pt}
```

Embedding CSS into HTML

Following are the four methods to add CSS to HTML documents.

1. Inline Style Sheets
2. Embedded Style Sheets
3. External Style Sheets
4. Imported Style Sheets

Inline Style Sheets

Inline Style Sheets are included with HTML element i.e. they are placed inline with the element. To add inline CSS, we have to declare style attribute which can contain any CSS property.

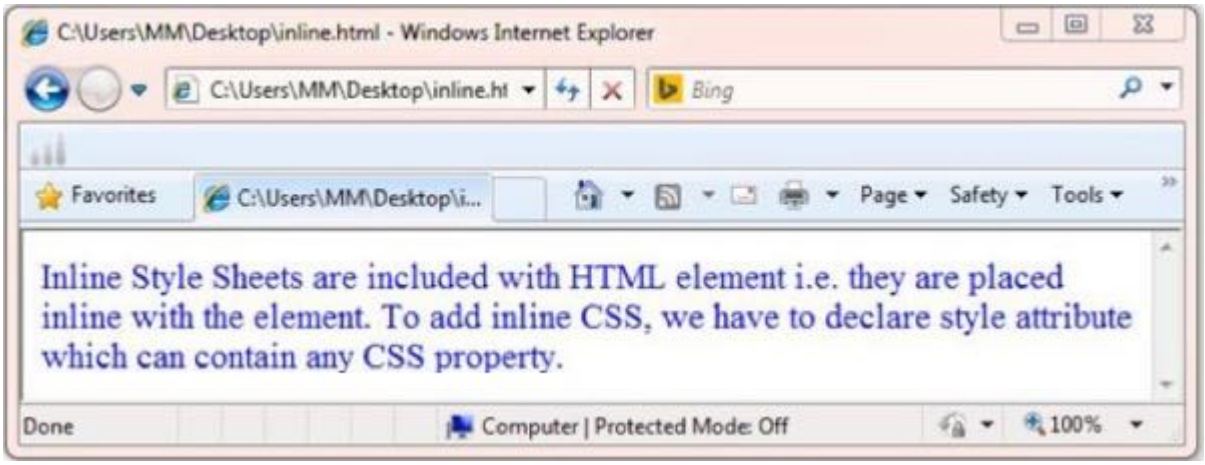
Syntax:

```
<Tagname STYLE = " Declaration1 ; Declaration2 ">  …. </Tagname>
```

Let's consider the following example using Inline Style Sheets:

```
<p style="color: blue; text-align: left; font-size: 15pt">
Inline Style Sheets are included with HTML element i.e. they are placed inline with the
element.
To add inline CSS, we have to declare style attribute which can contain any CSS property.
</p>
```

Output –



Embedded Style Sheets

Embedded Style Sheets are used to apply same appearance to all occurrence of a specific element. These are defined in <head> element by using the **<style>** element.

> *The **<style>** element must include **type** attribute. The value of **type** attribute specifies what type of syntax it includes when rendered by the browser.*

Syntax

```
<head> <title> …. </title>
<style type =”text/css”>
…….CSS Rules/Styles….
</head>
```

Let’s consider the following example using Embedded Style Sheets:

```
<style type="text/css">
p {color:green; text-align: left; font-size: 10pt}
h1 { color: red; font-weight: bold}
</style>
```

External Style Sheets

External Style Sheets are the separate **.css** files that contain the CSS rules. These files can be linked to any HTML documents using <link> tag with rel attribute.

Syntax:

```
<head> <link rel= “stylesheet”  type=”text/css” href= “url of css file”>
</head>
```

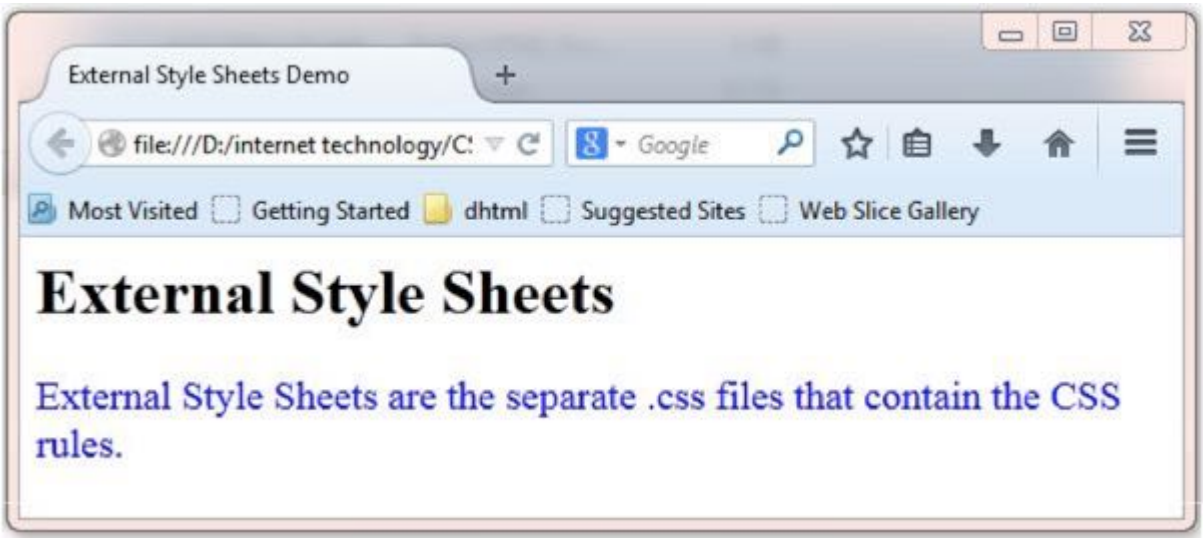
In order to create external css and link it to HTML document, follow the following steps:

- First of all create a CSS file and define all CSS rules for several HTML elements. Let’s name this file as external.css.

```
p
 {
   Color: orange;     text-align:  left;        font-size: 10pt;
}
h1
 {
   Color: orange;      font-weight: bold;
}
```

- Now create HTML document and name it as **externaldemo.html.**

```
<html>
<head>
<title> External Style Sheets Demo </title>
<link rel="stylesheet"  type="text/css" href="external.css">
</head>
<body>
<h1> External Style Sheets</h1>
<p>External Style Sheets are the separate .css files that contain the CSS rules.</p>
</body>
</html>
```



Imported Style Sheets

Imported Style Sheets allow us to import style rules from other style sheets. To import CSS rules we have to use @import before all the rules in a style sheet.

Syntax:

```
<head><title> Title Information </title>
<style type=”text/css”>
@import URL (cssfilepath)
```

```
… CSS rules…
</style>
</head>
</style>
```

Let’s consider the following example using Inline Style Sheets:

```
<html>
<head>
<title> External Style Sheets Demo </title>
<style>
@import url(external.css);
</style>
</head>
<body>
<h1> External Style Sheets</h1>
<p>External Style Sheets are the separate .css files that contain the CSS rules.</p>
</body>
</html>
```

JAVASCRIPT

Introduction

JavaScript is a lightweight, interpreted programming language with object-oriented capabilities that allows you to build interactivity into otherwise static HTML pages.

> *JavaScript code is not compiled but translated by the translator. This translator is embedded into the browser and is responsible for translating javascript code.*

Key Points

- It is Lightweight, interpreted programming language.
- It is designed for creating network-centric applications.
- It is complementary to and integrated with Java.
- It is complementary to and integrated with HTML
- It is an open and cross-platform

JavaScript Statements

JavaScript statements are the commands to tell the browser to what action to perform. Statements are separated by semicolon ; .

> *JavaScript statement constitutes the JavaScript code which is translated by the browser line by line.*

Example of JavaScript statement:

```
document.getElementById("demo").innerHTML = "Welcome";
```

Following table shows the various JavaScript Statements −

Sr.No.	Statement	Description
1.	switch case	A block of statements in which execution of code depends upon different cases. The interpreter checks each case against the value of the expression until a match is found. If nothing matches, a **default** condition will be used.
2.	If else	The **if** statement is the fundamental control statement that allows JavaScript to make decisions and execute statements conditionally.
3.	While	The purpose of a while loop is to execute a statement or code block repeatedly as long as expression is true. Once expression becomes false, the loop will be exited.
4.	do while	Block of statements that are executed at least once and continues to be executed while condition is true.
5.	for	Same as while but initialization, condition and increment/decrement is done in the same line.

6.	for in	This loop is used to loop through an object's properties.
7.	continue	The continue statement tells the interpreter to immediately start the next iteration of the loop and skip remaining code block.
8.	break	The break statement is used to exit a loop early, breaking out of the enclosing curly braces.
9.	function	A function is a group of reusable code which can be called anywhere in your programme. The keyword function is used to declare a function.
10.	return	Return statement is used to return a value from a function.
11.	var	Used to declare a variable.
12.	try	A block of statements on which error handling is implemented.
13.	catch	A block of statements that are executed when an error occur.
14.	throw	Used to throw an error.

JavaScript Comments

JavaScript supports both C-style and C++-style comments, thus:

- Any text between a // and the end of a line is treated as a comment and is ignored by JavaScript.
- Any text between the characters /* and */ is treated as a comment. This may span multiple lines.
- JavaScript also recognizes the HTML comment opening sequence <!--. JavaScript treats this as a single-line comment, just as it does the // comment.-->
- The HTML comment closing sequence --> is not recognized by JavaScript so it should be written as //-->.

Example

```
<script language="javascript" type="text/javascript">
<!--

// this is a comment. It is similar to comments in C++

/*
 * This is a multiline comment in JavaScript
 * It is very similar to comments in C Programming
 */
//-->
<script>
```

JavaScript variable

Variables are referred as named containers for storing information. We can place data into these containers and then refer to the data simply by naming the container.

Rules to declare variable in JavaScript

Here are the important rules that must be followed while declaring a variable in JavaScript.

- In JavaScript variable names are case sensitive i.e. a is different from A.
- Variable name can only be started with a underscore _ or a letter *fromatozorAtoZ*, or dollar $ sign.

- Numbers $0 to 9$ can only be used after a letter.
- No other special character is allowed in variable name.

Before you use a variable in a JavaScript program, you must declare it. Variables are declared with the var keyword as follows −

```
<script type="text/javascript">
<!--
var money;
var name, age;
//-->
</script>
```

Variables can be initialized at time of declaration or after declaration as follows −

```
<script type="text/javascript">
<!--
var name = "Ali";
var money;
money = 2000.50;
//-->
</script>
```

Javascript Data Type

There are two kinds of data types as mentioned below −

- Primitive Data Type
- Non Primitive Data Type

The following table describes **Primitive Data Types** available in javaScript

Sr.No.	Datatype Description
1.	**String** Can contain groups of character as single value. It is represented in double quotes.E.g. var x= “tutorial”.
2.	**Numbers** Contains the numbers with or without decimal. E.g. var x=44, y=44.56;
3.	**Booleans** Contain only two values either true or false. E.g. var x=true, y= false.
4.	**Undefined** Variable with no value is called Undefined. E.g. var x;
5.	**Null** If we assign null to a variable, it becomes empty. E.g. var x=null;

The following table describes **Non-Primitive Data Types** in javaScript

Sr.No.	Datatype Description
1.	**Array** Can contain groups of values of same type. E.g. var x={1,2,3,55};
2.	**Objects** Objects are stored in property and value pair. E.g. var rectangle = { length: 5, breadth: 3};

JavaScript Functions

Function is a group of reusable statements *Code* that can be called any where in a program. In javascript function keyword is used to declare or define a function.

Key Points

- To define a function use function keyword followed by functionname, followed by parentheses .
- In parenthesis, we define parameters or attributes.
- The group of reusabe statements *code* is enclosed in curly braces {}. This code is executed whenever function is called.

Syntax

```
function functionname (p1, p2) {
 function coding…
}
```

JavaScript Operators

Operators are used to perform operation on one, two or more operands. Operator is represented by a symbol such as +, =, *, % etc. Following are the operators supported by javascript −

- Arithmetic Operators
- Comparison Operators
- Logical *orRelational* Operators
- Assignment Operators
- Conditional *orternary* Operators
- Arithmetic Operators

Arithmatic Operators

Following table shows all the arithmetic operators supported by javascript −

Operator	Description	Example
+	Add two operands.	10 + 10 will give 20
-	Subtract second operand from the first.	10 – 10 will give 0
*	Multiply two operands.	10 * 30 will give 300
/	Divide numerator by denominator	10/10 will give 1
%	It is called modulus operator and gives remainder of the	10 % 10 will give 0

	division.	
++	Increment operator, increases integer value by one	10 ++ will give 11
--	Decrement operator, decreases integer value by one	10 – will give 9

Comparison Operators

Following table shows all the comparison operators supported by javascript −

Operator	Description	Example
==	Checks if values of two operands are equal or not, If yes then condition becomes true.	10 == 10 will give true
!=	**Not Equal to operator** Checks if the value of two operands is equal or not, if values are not equal then condition becomes true.	10 !=10 will give false
>	**Greater Than operator** Checks if the value of left operand is greater than the value of right operand, if yes then condition becomes true.	20 > 10 will give true
<	**Less than operator** Checks if the value of left operand is less than the value of right operand, if yes then condition becomes true.	10 < 20 will give true
>=	**Greater than or equal to operator** Checks if the value of left operand is greater than or equal to the value of right operand, if yes then condition becomes true.	10 >=20 will give false
<=	**Less than or equal to operator** Checks if the value of left operand is less than or equal to the value of right operand, if yes then condition becomes true.	10 <=20 will give true.

Logical Operators

Following table shows all the logical operators supported by javascript −

Operator	Description	Example
&&	Logical **AND** operator returns true if both operands are non zero.	10 && 10 will give true.
||	Logical **OR** operator returns true If any of the operand is non zero	10 || 0 will give true.
!	Logical **NOT** operator complements the logical state of its operand.	! 10 && 10 will give false.

Assignment Operators

Following table shows all the assignment operators supported by javascript −

Operator	Description	Example
=	**Simple Assignment operator**	C = A + B will assign

	Assigns values from right side operands to left side operand.	value of A + B into C
+=	**Add AND assignment operator** It adds right operand to the left operand and assign the result to left operand	C += A is equivalent to C = C + A
-=	**Subtract AND assignment operator** It subtracts right operand from the left operand and assign the result to left operand	C -= A is equivalent to C = C - A
*=	**Multiply AND assignment operator** It multiplies right operand with the left operand and assign the result to left operand	C *= A is equivalent to C = C * A
/=	**Divide AND assignment operator** It divides left operand with the right operand and assign the result to left operand	C /= A is equivalent to C = C / A
%=	**Modulus AND assignment operator** Modulus AND assignment operator, It takes modulus using two operands and assign the result to left operand	C %= A is equivalent to C = C % A

Conditional Operator

It is also called ternary operator, since it has three operands.

Operator	Description	Example
?:	Conditional Expression	If Condition is true? Then value X : Otherwise value Y

Control Structure

Control structure actually controls the flow of execution of a program. Following are the several control structure supported by javascript.

- if ... else
- switch case
- do while loop
- while loop
- for loop

If ... else

The if statement is the fundamental control statement that allows JavaScript to make decisions and execute statements conditionally.

Syntax

```
if (expression){
   Statement(s) to be executed if expression is true
}
```

Example

```
<script type="text/javascript">
<!--
var age = 20;
```

```
if( age > 18 ){
   document.write("<b>Qualifies for driving</b>");
}
//-->
</script>
```

Switch case

The basic syntax of the switch statement is to give an expression to evaluate and several different statements to execute based on the value of the expression. The interpreter checks each case against the value of the expression until a match is found. If nothing matches, a default condition will be used.

Syntax

```
switch (expression)
{
  case condition 1: statement(s)
                    break;
  case condition 2: statement(s)
                    break;
   ...
  case condition n: statement(s)
                    break;
  default: statement(s)
}
```

Example

```
<script type="text/javascript">
<!--
var grade='A';
document.write("Entering switch block<br/>");
switch (grade)
{
  case 'A': document.write("Good job<br/>");
            break;
  case 'B': document.write("Pretty good<br/>");
            break;
  case 'C': document.write("Passed<br/>");
            break;
  case 'D': document.write("Not so good<br/>");
            break;
  case 'F': document.write("Failed<br/>");
            break;
  default:  document.write("Unknown grade<br/>")
}
document.write("Exiting switch block");
//-->
</script>
```

Do while Loop

The do...while loop is similar to the while loop except that the condition check happens at the end of the loop. This means that the loop will always be executed at least once, even if the condition is false.

Syntax

```
do{
   Statement(s) to be executed;
} while (expression);
```

Example

```
<script type="text/javascript">
<!--
var count = 0;
document.write("Starting Loop" + "<br/>");
do{
  document.write("Current Count : " + count + "<br/>");
  count++;
}while (count < 0);
document.write("Loop stopped!");
//-->
</script>
```

This will produce following result −

```
Starting Loop
Current Count : 0
Loop stopped!
```

While Loop

The purpose of a while loop is to execute a statement or code block repeatedly as long as expression is true. Once expression becomes false, the loop will be exited.

Syntax

```
while (expression){
   Statement(s) to be executed if expression is true
}
```

Example

```
<script type="text/javascript">
<!--
var count = 0;
document.write("Starting Loop" + "<br/>");
while (count < 10){
  document.write("Current Count : " + count + "<br/>");
  count++;
}
document.write("Loop stopped!");
//-->
</script>
```

This will produce following result −

```
Starting Loop
Current Count : 0
Current Count : 1
Current Count : 2
Current Count : 3
Current Count : 4
Current Count : 5
Current Count : 6
Current Count : 7
Current Count : 8
Current Count : 9
Loop stopped!
```

For Loop

The for loop is the most compact form of looping and includes the following three important parts −

- The loop initialization where we initialize our counter to a starting value. The initialization statement is executed before the loop begins.

- The test statement which will test if the given condition is true or not. If condition is true then code given inside the loop will be executed otherwise loop will come out.
- The iteration statement where you can increase or decrease your counter.

Syntax

```
for (initialization; test condition; iteration statement){
   Statement(s) to be executed if test condition is true
}
```

Example

```
<script type="text/javascript">
<!--
var count;
document.write("Starting Loop" + "<br/>");
for(count = 0; count < 10; count++){
  document.write("Current Count : " + count );
  document.write("<br/>");
}
document.write("Loop stopped!");
//-->
</script>
```

This will produce following result which is similar to while loop −

```
Starting Loop
Current Count : 0
Current Count : 1
Current Count : 2
Current Count : 3
Current Count : 4
Current Count : 5
Current Count : 6
Current Count : 7
Current Count : 8
Current Count : 9
Loop stopped!
```

Creating Sample Program

Following is the sample program that shows time, when we click in button.

```
<html>
<body>
<button onclick="this.innerHTML=Date()">The time is?</button>
<p>Click to display the date.</p>
<button onclick="displayDate()">The time is?</button>
<script>
function displayDate() {
    document.getElementById("demo").innerHTML = Date();
}</script>

<p ></p>

</body>
</html>
```

Output

PHP

Introduction

PHP is acronym of **Hypertext Preprocessor** *PHP* is a programming language that allows web developers to create dynamic content that interacts with databases.PHP is basically used for developing web based software applications.

PHP started out as a small open source project that evolved as more and more people found out how useful it was. Rasmus Lerdorf unleashed the first version of PHP way back in 1994.

Key Points

- PHP is a recursive acronym for "PHP: Hypertext Preprocessor".
- PHP is a server side scripting language that is embedded in HTML. It is used to manage dynamic content, databases, session tracking, even build entire e-commerce sites.
- It is integrated with a number of popular databases, including MySQL, PostgreSQL, Oracle, Sybase, Informix, and Microsoft SQL Server.
- PHP is pleasingly zippy in its execution, especially when compiled as an Apache module on the Unix side. The MySQL server, once started, executes even very complex queries with huge result sets in record-setting time.
- PHP supports a large number of major protocols such as POP3, IMAP, and LDAP. PHP4 added support for Java and distributed object architectures *COMandCORBA*, making n-tier development a possibility for the first time.

Uses of PHP

PHP has now become a poupular scripting language among web developer due to the following reasons −

- PHP performs system functions, i.e. from files on a system it can create, open, read, write, and close them.
- PHP can handle forms, i.e. gather data from files, save data to a file, through email you can send data, return data to the user.
- You add, delete, modify elements within your database through PHP.
- Access cookies variables and set cookies.
- Using PHP, you can restrict users to access some pages of your website.
- It can encrypt data.

Characteristics

Five important characteristics make PHP's practical nature possible −

- Simplicity
- Efficiency
- Security
- Flexibility
- Familiarity

"Hello World" Script in PHP

To get a feel for PHP, first start with simple PHP scripts. Since "Hello, World!" is an essential example, first we will create a friendly little "Hello, World!" script.

As mentioned earlier, PHP is embedded in HTML. That means that in amongst your normal HTML *orXHTMLifyou'recutting−edge* you'll have PHP statements like this −

```
<html>

   <head>
      <title>Hello World</title>
   </head>

   <body>
      <?php echo "Hello, World!";?>
   </body>

</html>
```

It will produce following result −

```
Hello, World!
```

If you examine the HTML output of the above example, you'll notice that the PHP code is not present in the file sent from the server to your Web browser. All of the PHP present in the Web page is processed and stripped from the page; the only thing returned to the client from the Web server is pure HTML output.

All PHP code must be included inside one of the three special markup tags ate are recognised by the PHP Parser.

```
<?php PHP code goes here ?>
<?    PHP code goes here ?>
<script language="php"> PHP code goes here </script>
```