

# **Styling and Scripting for Web Development**

---

## **DECAP774**

**Edited by**  
**Ajay Kumar Bansal**



**L**OVELY  
**P**ROFESSIONAL  
**U**NIVERSITY

---



L O V E L Y  
P R O F E S S I O N A L  
U N I V E R S I T Y

# **Styling and Scripting for Web Development**

**Edited By:  
Ajay Kumar Bansal**

**Title:** STYLING AND SCRIPTING FOR WEB DEVELOPMENT

**Author's Name:** Bhanu Sharma

**Published By :** Lovely Professional University

**Publisher Address:** Lovely Professional University, Jalandhar Delhi GT road, Phagwara - 144411

**Printer Detail:** Lovely Professional University

**Edition Detail:** (I)

ISBN: 978-81-19334-33-9



Copyrights@ Lovely Professional University

# Content

<b>Unit 1:</b>	<b>Introduction to HTML</b>	1
	<i>Bhanu Sharma, Lovely Professional University</i>	
<b>Unit 2:</b>	<b>HTML Forms and Frames</b>	13
	<i>Bhanu Sharma, Lovely Professional University</i>	
<b>Unit 3:</b>	<b>HTML Colors and XHTML</b>	41
	<i>Bhanu Sharma, Lovely Professional University</i>	
<b>Unit 4:</b>	<b>Introduction of HTML5</b>	54
	<i>Bhanu Sharma, Lovely Professional University</i>	
<b>Unit 5:</b>	<b>Advanced HTML5</b>	69
	<i>Bhanu Sharma, Lovely Professional University</i>	
<b>Unit 6:</b>	<b>Introduction of CSS, box model and advanced CSS</b>	89
	<i>Bhanu Sharma, Lovely Professional University</i>	
<b>Unit 7:</b>	<b>Advanced CSS</b>	102
	<i>Bhanu Sharma, Lovely Professional University</i>	
<b>Unit 8:</b>	<b>Images and Media types in Advanced CSS</b>	115
	<i>Bhanu Sharma, Lovely Professional University</i>	
<b>Unit 9:</b>	<b>Introduction of JavaScript, basic elements and JavaScript objects</b>	143
	<i>Bhanu Sharma, Lovely Professional University</i>	
<b>Unit 10:</b>	<b>Functions and Arrays in Java Script</b>	155
	<i>Bhanu Sharma, Lovely Professional University</i>	
<b>Unit 11:</b>	<b>Java Script Events and Validations</b>	167
	<i>Bhanu Sharma, Lovely Professional University</i>	
<b>Unit 12:</b>	<b>JavaScript - Browser Object Model</b>	179
	<i>Bhanu Sharma, Lovely Professional University</i>	
<b>Unit 13:</b>	<b>JavaScript-validation and Menu Builder</b>	207
	<i>Bhanu Sharma, Lovely Professional University</i>	
<b>Unit 14:</b>	<b>Bootstrap</b>	220
	<i>Bhanu Sharma, Lovely Professional University</i>	

## Unit 01: Introduction to HTML

### CONTENTS

- Objectives
- Introduction
- 1.1 Anatomy of an HTML document
- 1.2 HTML Editors
- 1.3 Marking up text
- 1.4 HTML Attributes
- Summary
- Keywords
- Self Assessment
- Answers for Self Assessment
- Review Questions
- Further Readings

### Objectives

- Learn about what is HTML?
- How HTML will help in Web development
- Understand about different tags

### Introduction

HTML (HyperText Markup Language) is the code that is used to structure a web page and its content. For example, content could be structured within a set of paragraphs, a list of bulleted points, or using images and data tables. HTML was first created by Tim Berners-Lee, Robert Cailliau, and others starting in 1989. It stands for Hyper Text Markup Language.

Hypertext means that the document contains links that allow the reader to jump to other places in the document or to another document altogether. The latest version is known as HTML5.

A Markup Language is a way that computers speak to each other to control how text is processed and presented. To do this HTML uses two things: tags and attributes.

### **So what is HTML?**

HTML is a markup language that defines the structure of your content. HTML consists of a series of elements, which you use to enclose, or wrap, different parts of the content to make it appear a certain way, or act a certain way. The enclosing tags can make a word or image hyperlink to somewhere else, can italicize words, can make the font bigger or smaller, and so on. For example, take the following line of content:

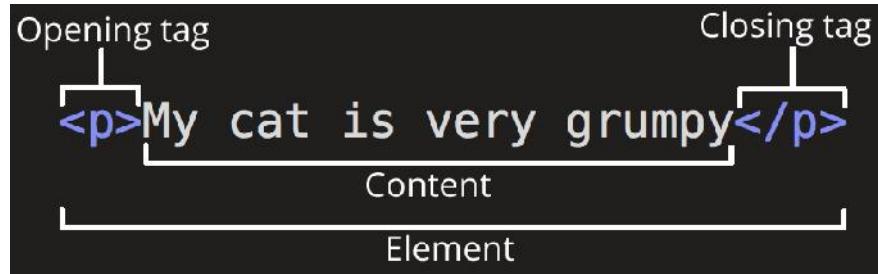
**My cat is very grumpy**

If we wanted the line to stand by itself, we could specify that it is a paragraph by enclosing it in paragraph tags:

```
<p>My cat is very grumpy</p>
```

### Anatomy of an HTML element

Let's explore this paragraph element a bit further.



*Figure 1 Anatomy of an HTML element*

The main parts of our element are as follows:

- **The opening tag:** This consists of the name of the element (in this case, p), wrapped in opening and closing angle brackets. This states where the element begins or starts to take effect – in this case where the paragraph begins.
- **The closing tag:** This is the same as the opening tag, except that it includes a forward slash before the element name. This states where the element ends – in this case where the paragraph ends. Failing to add a closing tag is one of the standard beginner errors and can lead to strange results.
- **The content:** This is the content of the element, which in this case, is just text.
- **The element:** The opening tag, the closing tag, and the content together comprise the element.

Elements can also have attributes that look like the following:

```
Attribute
<p class="editor-note">My cat is very grumpy</p>
```

Attributes contain extra information about the element that you don't want to appear in the actual content. Here, class is the attribute name and editor-note is the attribute value. The class attribute allows you to give the element a non-unique identifier that can be used to target it (and any other elements with the same class value) with style information and other things.

An attribute should always have the following:

- A space between it and the element name (or the previous attribute, if the element already has one or more attributes).
- The attribute name followed by an equal sign.
- The attribute value wrapped by opening and closing quotation marks.



**Notes:** Simple attribute values that don't contain ASCII whitespace (or any of the characters " ' ` = <> ) can remain unquoted, but it is recommended that you quote all attribute values, as it makes the code more consistent and understandable.

## Nested Elements

You can put elements inside other elements too — this is called nesting. If we wanted to state that our cat is very grumpy, we could wrap the word "very" in a  **element, which means that the word is to be strongly emphasized:**

```
<p>My cat is <strong>very</strong> grumpy. </p>
```

You do however need to make sure that your elements are properly nested. In the example above, we opened the 

element first, then the  **element; therefore, we have to close the  **element first, then the 

element. The following is incorrect:****

```
<p>My cat is <strong>very grumpy.</p></strong> ×
```

The elements have to open and close correctly so that they are clearly inside or outside one another. If they overlap as shown above, then your web browser will try to make the best guess at what you were trying to say, which can lead to unexpected results. So don't do it!

## Empty Elements

Some elements have no content and are called empty elements. Take the element that we already have in our HTML page:

```
<imgsrc="images/firefox-icon.png" alt="My test image">
```

This contains two attributes, but there is no closing tag and no inner content. This is because an image element doesn't wrap content to affect it. Its purpose is to embed an image in the HTML page in the place it appears.

## **1.1 Anatomy of an HTML document**

HTML elements are the building blocks of HTML pages. With HTML constructs, images and other objects such as interactive forms may be embedded into the rendered page. HTML provides a means to create structured documents by denoting structural semantics for text such as headings, paragraphs, lists, links, quotes and other items. HTML elements are delineated by tags, written using angle brackets. Tags such as and  directly introduce content into the page. Other tags such as 

surround and provide information about document text and may include other tags as sub-elements. Browsers do not display the HTML tags but use them to interpret the content of the page.

HTML can embed programs written in a scripting language such as JavaScript, which affects the behavior and content of web pages. Inclusion of CSS defines the look and layout of content. The World Wide Web Consortium (W3C), former maintainer of the HTML and current maintainer of the CSS standards, has encouraged the use of CSS over explicit presentational HTML since 1997.[ A form of HTML, known as HTML5, is used to display video and audio, primarily using the  element, in collaboration with javascript.

All HTML documents must start with a document type declaration: **<!DOCTYPE html>**.

The HTML document itself begins with **<html>** and ends with **</html>**.

The visible part of the HTML document is between **<body>** and **</body>**.

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>My test page</title>
</head>
<body>
<imgsrc="images/firefox-icon.png" alt="My test image">
```

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

Here, we have the following:

- **<!DOCTYPE html>** – doctype. It is a required preamble. In the mists of time, when HTML was young (around 1991/92), doctypes were meant to act as links to a set of rules that the HTML page had to follow to be considered good HTML, which could mean automatic error checking and other useful things. However these days, they don't do much and are basically just needed to make sure your document behaves correctly. That's all you need to know for now.
- **<html></html>** – the **<html>** element. This element wraps all the content on the entire page and is sometimes known as the root element.
- **<head></head>** – the **<head>** element. This element acts as a container for all the stuff you want to include on the HTML page that isn't the content you are showing to your page's viewers. This includes things like keywords and a page description that you want to appear in search results, CSS to style our content, character set declarations, and more.
- **<meta charset="utf-8">** – This element sets the character set your document should use to UTF-8 which includes most characters from the vast majority of written languages. Essentially, it can now handle any textual content you might put on it. There is no reason not to set this and it can help avoid some problems later on.
- **<title></title>** – the **<title>** element. This sets the title of your page, which is the title that appears in the browser tab the page is loaded in. It is also used to describe the page when you bookmark/favorite it.
- **<body></body>** – the **<body>** element. This contains all the content that you want to show to web users when they visit your page, whether that's text, images, videos, games, playable audio tracks, or whatever else.

## 1.2 HTML Editors

Web pages can be created and modified by using professional HTML editors.

However, for learning HTML we recommend a simple text editor like Notepad (PC) orTextEdit (Mac). We believe in that using a simple text editor is a good way to learn HTML. Text editors intended for use with HTML usually provide at least syntax highlighting. Some editors additionally feature templates, toolbars and keyboard shortcuts to quickly insert common HTML elements and structures. Wizards, tooltip prompts and autocompletion may help with common tasks.

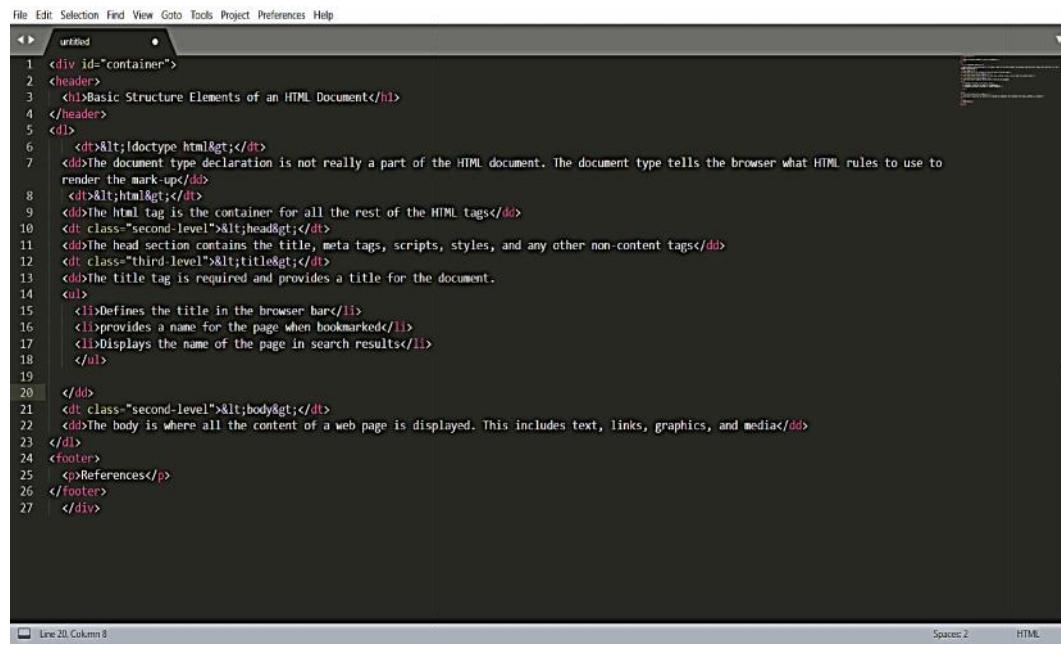
Text editors commonly used for HTML typically include either built-in functions or integration with external tools for such tasks as version control, link-checking and validation, code cleanup and formatting, spell-checking, uploading by FTP or WebDAV, and structuring as a project. Some functions, such as link checking or validation may use online tools, requiring a network connection.

Text editors require user understanding of HTML and any other web technologies the designer wishes to use like CSS, JavaScript and server-side scripting languages.

To ease this requirement, some editors allow editing of the markup in more visually organized modes than simple color highlighting, but in modes not considered WYSIWYG. These editors typically include the option of using palette windows or dialog boxes to edit the text-based parameters of selected objects. These palettes allow editing parameters in individual fields, or inserting new tags by filling out an onscreen form, and may include additional widgets to present and select options when editing parameters (such as previewing an image or text styles) or an outline editor to expand and collapse HTML objects and properties. Below are some of the text editors that can be used for HTML.

## Sublime Text 3

Sublime Text 3 as it is free and also offers cross-platform support for Windows, Mac, and Linux users.



The screenshot shows the Sublime Text 3 interface with a dark theme. The menu bar includes File, Edit, Selection, Find, View, Goto, Tools, Project, Preferences, and Help. The title bar says "untitled". The main editor area contains an HTML document with numbered line numbers from 1 to 27. Lines 1-5 define a container div. Lines 6-10 include a doctype declaration and a dd tag explaining it. Lines 11-14 describe the html tag as a container for head and body. Lines 15-19 detail the head section containing title, meta, and other tags. Lines 20-24 describe the body section. Lines 25-26 show a footer with a reference link. Line 27 closes the container div. A status bar at the bottom shows "Line 20, Column 8", "Spaces: 2", and "HTML".

```

1 <div id="container">
2 <header>
3   <h1>Basic Structure Elements of an HTML Document</h1>
4 </header>
5 <dd>
6   <dt>&lt;!DOCTYPE HTML&gt;</dt>
7   <dd>The document type declaration is not really a part of the HTML document. The document type tells the browser what HTML rules to use to render the mark-up</dd>
8   <dt>&lt;html&gt;</dt>
9   <dd>The html tag is the container for all the rest of the HTML tags</dd>
10  <dt class="second-level">&lt;head&gt;</dt>
11  <dd>The head section contains the title, meta tags, scripts, styles, and any other non-content tags</dd>
12  <dt class="third-level">&lt;title&gt;</dt>
13  <dd>The title tag is required and provides a title for the document.
14 <ul>
15   <li>Defines the title in the browser bar</li>
16   <li>Provides a name for the page when bookmarked</li>
17   <li>Displays the name of the page in search results</li>
18 </ul>
19
20 </dd>
21 <dt class="second-level">&lt;body&gt;</dt>
22 <dd>The body is where all the content of a web page is displayed. This includes text, links, graphics, and media</dd>
23 </dl>
24 <footer>
25   <p>References</p>
26 </footer>
27 | </div>

```

### Pros

- Easily customizable
- Beginner-friendly
- Pleasant color schemes to choose from.

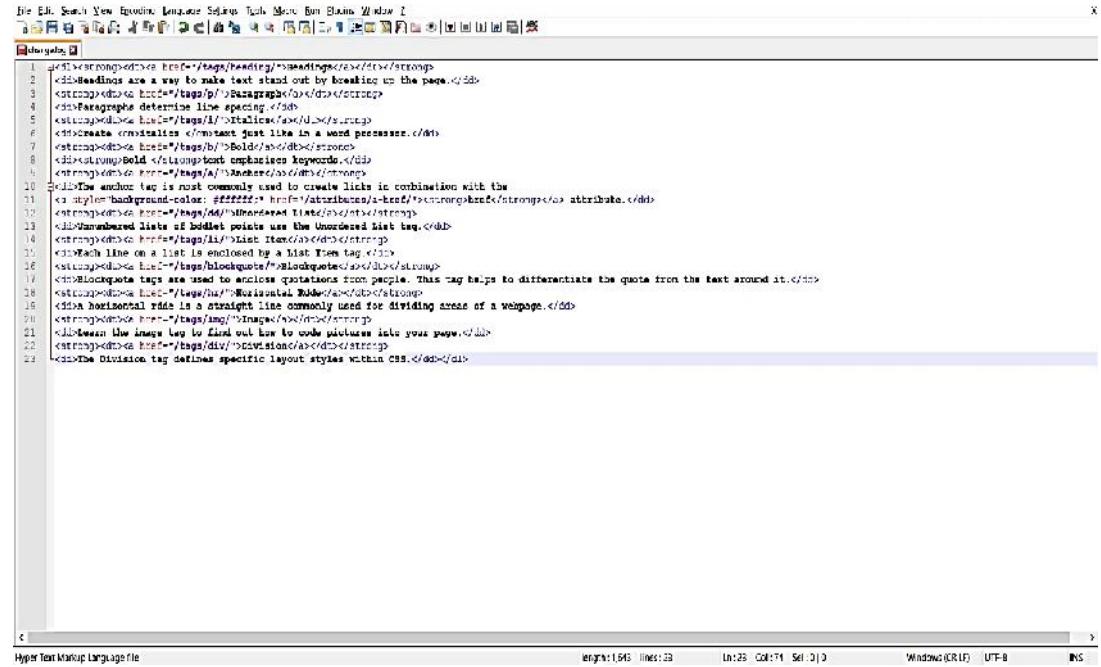
### Cons

- Can't print documents or code
- No toolbar or dashboard available.

### Notepad ++

Another common choice for HTML and other language coders is Notepad++. It is a tiny program to download and perform the functions you need for writing clean code.

## Styling and Scripting for Web Development



```

<!DOCTYPE html>
<html><head><title>My First Document</title></head>
<body>
    <p>This is my first paragraph. It has some text in it.</p>
    <hr>
    <p>This is another paragraph. It also has some text in it.</p>
</body>
</html>

```

The screenshot shows a Komodo Edit interface with a code editor window containing the provided HTML code. The status bar at the bottom displays file information: "HyperText Markup Language file", "Length: 1,562 Lines: 33", "Ln: 28 Col: 71 Sel: 0|3", "Windows (CR/LF)", "UTF-8", and "INS".

### Pros

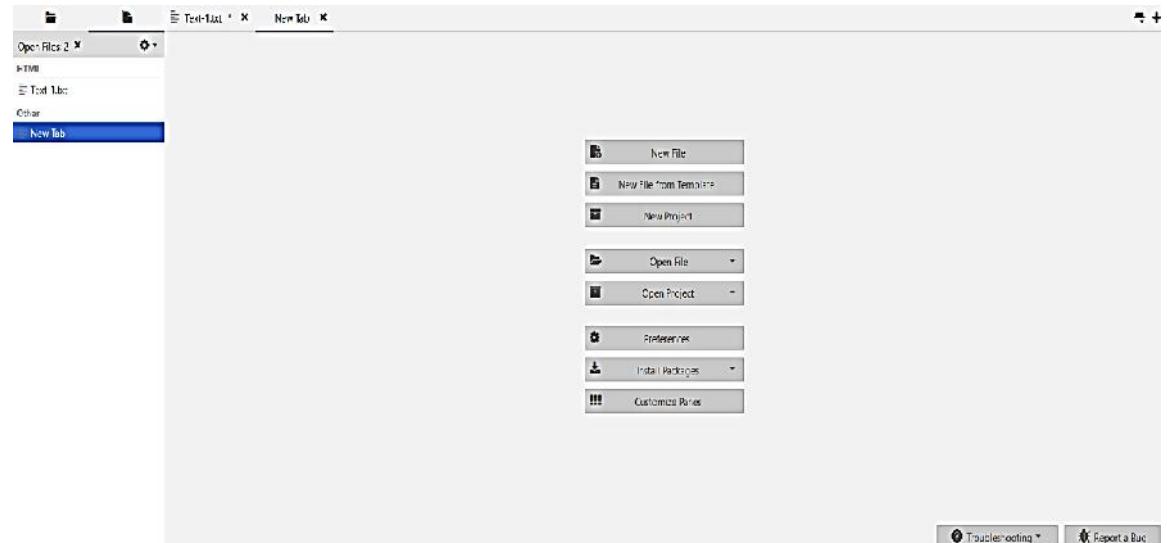
- Distraction-free interface
- Auto-completion feature
- Plugin options for extended functionalities.

### Cons

- Can be difficult to get used to for beginners
- No support for Mac.

### Komodo Edit

Komodo Edit is one of two editors released by the same label. They offer a simple, open-source editor with a variety of extensions and language support.



### Pros

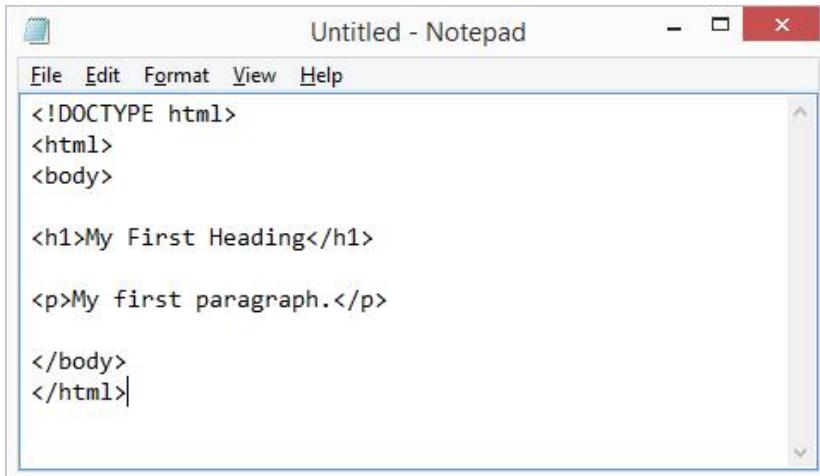
- Easy-to-grasp coding interface
- Available for Mac, Windows, and Linux
- Impressive language support.

## Cons

- No autocompletion by default
- Visual settings are difficult to find and change.

## Notepad

The best and easiest HTML editor that can be used in Notepad



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

<h1>My First Heading</h1>

<p>My first paragraph.</p>

</body>
</html>|
```

While creating a document just save the document with .HTML extension it will open in the web browser.

## 1.3 Marking up text

This section will cover some of the essential HTML elements you'll use for marking up the text.

### Headings

Heading elements allow you to specify that certain parts of your content are headings — or subheadings. In the same way that a book has the main title, chapter titles, and subtitles, an HTML document can too. HTML contains 6 heading levels, `<h1>` - `<h6>`, although you'll commonly only use 3 to 4 at most:

```
<!-- 4 heading levels: -->
<h1>My main title</h1>
<h2>My top level heading</h2>
<h3>My subheading</h3>
<h4>My sub-subheading</h4>
```

Note: Anything in HTML between `<!--` and `-->` is an HTML comment. The browser ignores comments as it renders the code. In other words, they are not visible on the page - just in the code. HTML comments are a way for you to write helpful notes about your code or logic.

### Paragraphs

As explained above, `<p>` elements are for containing paragraphs of text; you'll use these frequently when marking up regular text content:

```
<p>This is a single paragraph</p>
```

Add your sample text (you should have it from What will your website look like?) into one or a few paragraphs, placed directly below your <img> element.

### Links

Links are very important — they are what makes the web a web! To add a link, we need to use a simple element — <a> — "a" being the short form for "anchor". To make text within your paragraph into a link, follow these steps:

- Choose some text. We choose the text "Google".
- Wrap the text in an <a> element, as shown below:

<a> Google </a>

- Give the <a> element an href attribute, as shown below:

<a href=""> Google </a>

Fill in the value of this attribute with the web address that you want the link to link to:

<a href=" https://www.google.com/">Google</a>

You might get unexpected results if you omit the https:// or http:// part, called the protocol, at the beginning of the web address. After making a link, click it to make sure it is sending you where you wanted it to.

## 1.4 HTML Attributes

HTML attributes provide additional information about HTML elements.

### Properties of HTML Attributes

- All HTML elements can have attributes
- Attributes provide additional information about elements
- Attributes are always specified in the start tag
- Attributes usually come in name/value pairs like: name="value"

### The href Attribute

The <a> tag defines a hyperlink. The href attribute specifies the URL of the page the link goes to:

*Syntax:*

<a href=" https://www.google.com/">Google</a>

### src Attribute

The <img> tag is used to embed an image in an HTML page. The src attribute specifies the path to the image to be displayed:



### The width and height Attributes

The <img> tag should also contain the width and height attributes, which specifies the width and height of the image (in pixels):



## The alt Attribute

The required **alt** attribute for the **<img>** tag specifies an alternate text for an image, if the image for some reason cannot be displayed. This can be due to slow connection, or an error in the **src** attribute, or if the user uses a screen reader.

```

```

## Style Attribute

The style attribute is used to add styles to an element, such as color, font, size, and more.

```
<p style="color:red;">This is a red paragraph.</p>
```

## The title Attribute

The title attribute defines some extra information about an element.

The value of the title attribute will be displayed as a tooltip when you mouse over the element:

```
<p title="I'm a tooltip">This is a paragraph.</p>
```



**Task:** Write down the code to create your first HTML page by using all the tags discussed above

## Summary

- All HTML elements can have attributes
- The href attribute of **<a>** specifies the URL of the page the link goes to
- The src attribute of **<img>** specifies the path to the image to be displayed
- The width and height attributes of **<img>** provide size information for images
- The alt attribute of **<img>** provides an alternate text for an image
- The style attribute is used to add styles to an element, such as color, font, size, and more
- The lang attribute of the **<html>** tag declares the language of the Web page
- The title attribute defines some extra information about an element

## Keywords

- **Hyper Text markup language:** HTML (HyperText Markup Language) is the most basic building block of the Web. It defines the meaning and structure of web content.
- **HTML tag:** An HTML tag is a piece of markup language used to indicate the beginning and end of an HTML element in an HTML document.
- **Title:** the **<title>** element. This sets the title of your page, which is the title that appears in the browser tab the page is loaded in. It is also used to describe the page when you bookmark/favorite it.
- **Attribute:** An HTML attribute is a piece of markup language used to adjust the behavior or display of an HTML element. For example, attributes can be used to change the color, size, or functionality of HTML elements.
- **HTML Editors:** HTML editor is a software used for writing code in HTML which is used for structuring and creating websites. Even though codes can be written from scratch using a normal text editor, HTML editors provide a great deal of ease to the developers by ensuring hassle-free coding.

## **SelfAssessment**

1. What is HTML?
  - A. HTML describes the structure of a webpage
  - B. HTML is the standard markup language mainly used to create web pages
  - C. HTML consists of a set of elements that helps the browser how to view the content
  - D. All of the mentioned
  
2. HTML stands for \_\_\_\_\_
  - A. HyperText Markup Language
  - B. HyperText Machine Language
  - C. HyperText Marking Language
  - D. HighText Marking Language
  
3. What is the correct syntax of doctype in HTML?
  - A. </doctype html>
  - B. <doctype html>
  - C. <doctype html!>
  - D. <!doctype html>
  
4. Which of the following tag is used for inserting the largest heading in HTML?
  - A. head
  - B. <h1>
  - C. <h6>
  - D. heading
  
5. In which part of the HTML metadata is contained?
  - A. head tag
  - B. title tag
  - C. html tag
  - D. body tag
  
6. How do we write comments in HTML?
  - A. </.....>
  - B. <!.....>
  - C. </...../>
  - D. <.....!>
  
7. The correct sequence of HTML tags for starting a webpage is -
  - A. Head, Title, HTML, body
  - B. HTML, Body, Title, Head

- C. HTML, Head, Title, Body
- D. HTML, Head, Title, Body

8. Which character is used to represent the closing of a tag in HTML?

- A. \
- B. !
- C. /
- D. .

9. ALL HTML tags are enclosed in what?

- A. # and #
- B. ? and !
- C. < and >
- D. { and }

10. To create HTML page, you need \_\_\_\_\_

- A. Web browser
- B. text editor
- C. Both [A] and [B]
- D. None of the above

11. The BODY tag is usually used after \_\_\_\_\_

- A. HTML tag
- B. EM tag
- C. TITLE tag
- D. HEAD tag

12. Which tag tells the browser where the page starts and stops?

- A. <html>
- B. <body>
- C. <head>
- D. <title>

13. Which program do you need to write HTML?

- A. A graphics program
- B. Any text editor
- C. HTML -development suite 4
- D. All of the above

14. HTML uses

- A. User defined tags

- B. Pre-specified tags
- C. Fixed tags defined by the language
- D. Tags only for linking

15. Fundamental HTML Block is known as \_\_\_\_\_.

- A. HTML Body
- B. HTML Tag
- C. HTML Attribute
- D. HTML Element

### **Answers for Self Assessment**

- |       |       |       |       |       |
|-------|-------|-------|-------|-------|
| 1. D  | 2. A  | 3. D  | 4. C  | 5. A  |
| 6. B  | 7. C  | 8. C  | 9. C  | 10. C |
| 11. D | 12. A | 13. B | 14. C | 15. B |

### **Review Questions**

1. What do you mean by HTML? Discuss the use of HTML
2. Discuss the difference between a Tag and an Attribute in HTML.
3. Write down the steps or code to create a simple HTML document and print Welcome to my new HTML page.
4. Explain all the tags that are used to create an HTML Page along with there functionality.
5. Discuss about some of the HTML editors that can be used for web development.



### **Further Readings**

<https://www.computer-pdf.com/web-programming/html/>



### **Web Links**

<https://matfuvit.github.io/UVIT/predavanja/literatura/TutorialsPoint%20HTML.pdf>

## Unit 02: HTML Forms and Frames

### **CONTENTS**

Objectives

Introduction

2.1     HTML Tables

2.2     Creating Tables

2.3     Tables and the Border Attribute

2.4     Linking Document

2.5     Inserting inline images

2.6     Creating Image Links

2.7     IMG Attributes

2.8     HTML Lists

2.9     HTML List Classes

2.10    HTML List Types

2.11    HTML Block and Inline Elements

2.12    HTML Layout Elements and Techniques

2.13    HTML Forms

2.14    I frames in HTML

Summary

Keywords

Self Assessment

Answers for Self Assessment

Review Questions

Further Readings

### Objectives

- Describe about HTML Tables
- Explain how HTML Tables are created
- Explain the steps involved in creating HTML Tables
- Describe about Linking Documents
- Describe about Creating Link Lists

### Introduction

In this unit, we describe the statements for creating and updating tables. We assume that the user knows about the data which has to be stored and what the structure of the data is, i.e., what tables are to be created and what the appropriate columns are.

### **2.1    HTML Tables**

Tables are defined with the <table> tag. A table is divided into rows (with the <tr> tag), and each

row is divided into data cells (with the `<td>` tag). The letters `td` stands for “table data,” which is the content of a data cell. A data cell can contain text, images, lists, paragraphs, forms, horizontal rules, tables, etc.


**Example:**

```
<table border="1">
<tr>
<th>Heading</th>
<th>Another Heading</th>
</tr>
<tr>
<td>row 1, cell 1</td>
<td>row 1, cell 2</td>
</tr>
<tr>
<td>row 2, cell 1</td>
<td>row 2, cell 2</td>
</tr>
</table>
```

**Output**

<u>Heading</u>	<u>Another Heading</u>
row 1, cell 1	row 1, cell 2
row 2, cell 1	row 2, cell 2

## **2.2 Creating Tables**

The basic structure of an HTML table consists of the following tags:      Table tags:  
`<TABLE></TABLE>`

- Row tags: `<TR></TR>`
- Cell tags: `<TD></TD>`

Constructing an HTML table consists of describing the table between the beginning table tag,

`<TABLE>`, and the ending table tag, `</TABLE>`. Between these tags, you then construct each row and each cell in the row. To do this, you would first start the row with the beginning row tag,

LOVELY PROFESSIONAL UNIVERSITY 47

Unit 5: Creating Tables

`<TR>`, and then build the row by creating each cell with the beginning cell tag, `<TD>`, adding the Notes

data for that cell, and then closing the cell with the ending cell tag, `</TD>`. When you finish all

of the cells for a row, you would then close the row with the ending row tag, </TR>.



**Notes:** For each new row, you would repeat the process of beginning the row, building each cell in the row, and closing the row.



**Example:**

The following table is an example of a basic table with three rows and two columns of data.

Data 1 Data 2

Data 3 Data 4

Data 5 Data 6

The codes that generated this table will look like this:

```
<TABLE>
<TR>
<TD>Data 1</TD>
<TD>Data 2</TD>
</TR>
<TR>
<TD>Data 3</TD>
<TD>Data 4</TD>
</TR>
<TR>
<TD>Data 5</TD>
<TD>Data 6</TD>
</TR>
</TABLE>
```

This table contains no border, title, or headings. If you wish to add any of these elements to your table, you need to include additional HTML codes.

### **2.3 Tables and the Border Attribute**

If you do not specify a border attribute the table will be displayed without any borders. Sometimes this can be useful, but most of the time, you want the borders to show.

To display a table with borders, you will have to use the border attribute:

```
<table border="1">
<tr>
<td>Row 1, cell 1</td>
<td>Row 1, cell 2</td>
</tr>
</table>
```

## Headings in a Table

Headings in a table are defined with the <th> tag.

```
<table border="1">
<tr>
<th>Heading</th>
<th>Another Heading</th>
</tr>
<tr>
<td>row 1, cell 1</td>
<td>row 1, cell 2</td>
</tr>
<tr>
<td>row 2, cell 1</td>
<td>row 2, cell 2</td>
</tr>
</table>
```

### Output

Heading	Another Heading
row 1, cell 1	row 1, cell 2
row 2, cell 1	row 2, cell 2

## Empty Cells in a Table

Table cells with no content are not displayed very well in most browsers.

```
<table border="1">
<tr>
<td>row 1, cell 1</td>
<td>row 1, cell 2</td>
</tr>
<tr>
<td>row 2, cell 1</td>
<td></td>
</tr>
</table>
```

### Output

row 1, cell 1	row 1, cell 2
row 2, cell 1	



**Note:** that the borders around the empty table cell are missing (NB! Mozilla Firefox displays the border).

To avoid this, add a non-breaking space (&nbsp;) to empty data cells, to make the borders visible:

```
<table border="1">
<tr>
<td>row 1, cell 1</td>
<td>row 1, cell 2</td>
</tr>
<tr>
<td>row 2, cell 1</td>
<td>&nbsp;</td>
</tr>
</table>
```

#### Output

row 1, cell 1	row 1, cell 2
row 2, cell 1	

## Table Tags

Table 1 shows different table tags that can be used in HTML

Tag	Description
<table>	Defines a table
<th>	Defines a table header
<tr>	Defines a table row
<td>	Defines a table cell
<caption>	Defines a table caption
<colgroup>	Defines groups of table columns
<col>	Defines the attribute values for one or more columns in a table
<thead>	Defines a table head
<tbody>	Defines a table body
<tfoot>	Defines a table footer

Table 1 Table tags in HTML



Task: Illustrate the use of various table tags.

## 2.4 Linking Document

Once you have the ability to create HTML pages, you'll want to learn how to create links between them, so that you can start building a site. Links are the essence of HTML – they are what makes it unique.

What makes the web so effective is the ability to define links from one page to another, and to follow links at the click of a button. A single click can take you right across the world!

## Links

Links are defined with the `<a>` tag. Let's create a link to the page defined in the file "example1.html": This link to

```
<a href="Example1.html">My first homepage</a>.
```

The text between the `<a>` and the `</a>` is used as the caption for the link. It is common for the caption to be in blue underlined text. It is by the way a good idea to not have any blank spaces in the names of your HTML files, as this might create problems with some web servers. You can use an underscore, `"_"`, to separate words in your file names. To link to a page on another web site you need to give the full web address (the URL). For instance, to link to "Google" you need to write: A link to `<a href="http://www.google.com">Google</a>`. If you want the user's browser to open a new window for the linked page, (that way the user finds back to your page as soon as he or she closes the new window), use the attribute `target`:

```
A link to <a href="http://www.google.com target='_blank'">Google</a>.
```

## 2.5 Inserting inline images

`<Img>` causes an "inline image" to be inserted into the output. The image will be retrieved and rendered as if it were just another part of the text. Inline images can occur within headings, or paragraphs, almost anywhere in fact, except body (in other words, you can have a 'free floating' `<img>` tag – it must be contained within some other element.)

Like `<hr>`, this is an empty element. That is, there must be no end-tag. You will sometimes see `<hr>` used with an end-tag (e.g. as a container around a caption), but this is obsolete usage.

`<Img>` has 1 required attribute `src` as well as 3 optional attributes.

**src** The `src` attribute is used to specify the URL of the image (i.e. the address or filename the browser uses to retrieve the image file),

e.g. ``

``

**alt** is used to provide an text alternative to the image for readers whose browsers do not support graphics (or for visually impaired readers using alternative display devices). Although not required, the use of the `alt` attribute is nearly always appropriate and is strongly recommended. The only exception might be cases where the image is strictly decorative or of generic character. In this case the default text chosen by the browser (typically something like "[IMAGE]" may be adequate).

**align** `align` can take one of three values:

- top
- middle
- bottom

and is used to indicate how the browser should align the image with the adjacent text.

**bottom**: align the bottom of the image with the bottom of text

**middle**: align the middle of the image with the middle of text

**top**: align the top of the image with the top of text.

Images can also be retrieved using by means of a hypertext link using the `<a>` tag. The key difference between an inline image and an image retrieved with the `<a>` tag is that an inline image requires no action on the part of the reader; it is retrieved with the page just as if it were part of the text. With the `<a>` tag, the reader has to make a special action (e.g. clicking on a button) to retrieve the image.



**Task:** Give examples of the attributes used with `<img>` tag.

## 2.6 Creating Image Links

Image links are constructed as you might expect, by embedding an `<img>` tag inside of an anchor element `<a>`. Like HTML text links, image links require opening and closing anchor tags, but instead of placing text between these opening and closing tags, the developer needs to place an image tag – with a valid source attribute value of course.



**Example:**

```
<!DOCTYPE html>
<html>
<body>
<p>Create a link of an image: <a href="default.html">
</a></p>
<p>No border around the image, but still a link: <a href="default.html">
</a>
</p>
</body>
</html>
```

### Output

Create a link of an image:

No border around the image, but still a link:

## 2.7 IMG Attributes

`<IMG>` indicates that an image – such as a photograph, icon, animation, cartoon, or other graphic – is to be displayed at that location. The `<IMG>` tag should contain within it further parameters as part of the command: `SRC= "URL/graphic.gif or .jpg"`: contains the URL (Uniform Resource Locator or web address) and name of the graphic image file, such as `graphic.gif` or `graphic.jpg`. Most commonly, the photograph, icon, or other graphic is a “`gif`” (Graphics Interchange Format image) or a “`jpg`” (Joint Photographic Experts Group image), both of which are recognized by most browsers. Some browsers also will recognize a “`bmp`” (Bitmap image) and a “`tif`” or “`tiff`” (Tag Image File Format image). Usually, the location source of the graphic file is in an adjacent directory such as “`graphics`,” or it possibly might be in the same directory. Assuming the image is a `.jpg` image, if the graphic is in an adjacent “`graphics`” directory, the tag would read: `<IMG SRC= "graphics/ graphic.jpg">`. If the image is located within the same directory as the document, the tag would read simply: `<IMG SRC="graphic.jpg">`. If the location of the image is somewhere else on the web, the tag might read something like this: `<IMG SRC= "http://www.tedmontgomery.com/tutorial/graphics/graphic.gif">`.

`ALIGN="LEFT" | "RIGHT" | "TOP" | "TEXTTOP" | "MIDDLE" | "ABSMIDDLE" | "BASELINE" | "BOTTOM" | "ABSBOTTOM"`: places the graphic image at a specified position, in relation either to the page margins or to the text. (Some browsers will not recognize all of these parameters.) “`LEFT`” aligns the image with the left margin of the page and allows text to wrap around the right side of the image. “`RIGHT`” aligns the image with the right margin of the page and allows the text to wrap around the left side of the image. Note: The only way to center a graphic horizontally on a page is to use `<CENTER>&</ CENTER>` tags around the `<IMG SRC>` tag. However, centering a graphic in this manner will prevent text from being

wrapped around either side of it. Also, any ALIGN="RIGHT" or ALIGN="LEFT" parameter within the <IMG> tag will override the effect of the centering tags. "TOP" aligns the top of the image with the top of the tallest item in the line. "TEXTTOP" aligns the top of the image with the top of the tallest text in the line; usually, but not always, the same as the "TOP" parameter. "MIDDLE" aligns the middle of the image with the baseline of the current line. "ABSMIDDLE" aligns the middle of the image with the middle of the current line. "BASELINE" aligns the bottom of the image with the baseline of the current line. "BOTTOM" is the same as the "BASELINE" parameter. "ABSBOTTOM" aligns the bottom of the image with the bottom of the current line; usually, but not always, the same as the "BASELINE" or "BOTTOM" parameter.

WIDTH="W": defines the width "W" of the image in pixels.

HEIGHT="H": defines the height "H" of the image in pixels.

BORDER="B": creates a border around the image, with a uniform width of "B" in pixels. (In case the image is incorporated as a hyperlink, the unvisited, active, and visited colors of the border will be the same as that of the other text hyperlinks on the page.) The default setting is BORDER= "2" (that is, a border 2 pixels wide).

HSPACE="H": creates a space, with width "H" in pixels, between the image and any text immediately to the right and/or left of it. (HSPACE means "horizontal space.")

VSPACE="V": creates a space, with height "V" in pixels, between the image and any text immediately above and/or below it. (VSPACE means "vertical space.")

ALT="alternate description": supplies a description of the image, which will be displayed instead of the image on non-graphical browsers. On typical graphical browsers, this description will appear before the image has loaded and also when the arrow is placed anywhere on the image.

TITLE="title": same function as the ALT tag, which is not recognized by some browsers.

ISMAP: indicates a server-side image map.

USEMAP: indicates a client-side image map

## 2.8 HTML Lists

Lists commonly are found in documents, including web pages. They are an easy and effective way to itemize such things as elements, components, or ingredients. Words or phrases which need to be set apart from the rest of the body of text can be emphasized with a "bullet" (a heavy dot used for calling attention to a particular section of text). An empty tag called a "list" tag is used to do this: <LI>: creates a bullet in front of text which is to be set apart for emphasis and causes all text after it to be indented, either until another list tag is detected or until the end of the list is reached. It is used to itemize elements of "unordered" and "ordered" lists.



**Notes:** A <BR> tag is not inserted at the end of an item beginning with a <LI> tag, as a line break automatically occurs at that point.

## 2.9 HTML List Classes

The HTML List classes allow you to easily create lists within your HTML pages. These classes provide methods to get and set various attributes of the lists and the items within the lists. In particular, the parent class HTML List provides a method to produce a compact list that displays items in as small a vertical space as possible.

**Methods for HTML List include:**

Compact the list

Add and remove items from the list

Add and remove lists from the list (making it possible to nest lists)

**Methods for HTML List Item include:**

Get and set the contents of the item

Get and set the direction of the text interpretation

Get and set the language of the input element Use the subclasses of HTML List and HTML List Item to create your HTML lists:

OrderedList and OrderedListItem

UnorderedList and UnorderedListItem

### **OrderedList and OrderedListItem**

Use the `OrderedList` and `OrderedListItem` classes to create ordered lists in your HTML pages.

**Methods for `OrderedList` include:**

Get and set the starting number for the first item in the list

Get and set the type (or style) for the item numbers

Methods for `OrderedListItem` include:

Get and set the number for the item

Get and set the type (or style) for the item number

### **UnorderedList and UnorderedListItem**

Use the `UnorderedList` and `UnorderedListItem` classes to create unordered lists in your HTML pages.

**Methods for `UnorderedList` include:**

Get and set the type (or style) for the items

**Methods for `UnorderedListItem` include:**

Get and set the type (or style) for the item

## **2.10 HTML List Types**

HTML provides three different types of lists to choose from when building a page, including unordered, ordered, and definition lists. Unordered lists are for lists of items where order isn't of important. While ordered lists place strong importance on the order of items. In the case where there is a list of terms and descriptions, perhaps for a glossary, definition lists are available.

### **Unordered List**

Unordered lists are purely a list of related items, in which their order does not matter nor do they have a numbered or alphabetical list element. Creating an unordered list in HTML is accomplished using the unordered list, `ul`, block level element. Each list item within an unordered list is individually marked up using the list item, `li`, block level element. By default most browsers represent each list item with a solid dot.

#### **Example**

```
<ul>
<li>Tamilnadu</li>
<li>Uttar Pradesh</li>
<li>West-Bengal</li>
</ul>
```

### **Ordered List**

The ordered list element, ol, works just like the unordered list element, including how each individual list item is created. The main difference between an ordered list and an unordered list is that with an ordered list the order of which items are represented is important. Instead of showing a dot as the default list item element, an ordered list uses numbers. Using CSS, these numbers can then be changed to letters, Roman numerals, and so on.

#### Example

```
<ol type="1">
<li>Tamilnadu</li>
<li>Uttar Pradesh</li>
<li>West-Bengal</li>
</ol>
```

#### Description List

Creating a definition list in HTML is accomplished using the dl element. Instead of using the li element to mark up list items, the definition list actually requires two elements: the definition term element, dt, and the definition description element, dd. A definition list may contain numerous terms and descriptions, one after the other. Additionally, a definition list may have multiple terms per description as well as multiple descriptions per term. A single term may have multiple meanings and warrant multiple definitions. In comparison, a single description may be suitable for multiple terms.

```
<dl>
<dt>study</dt>
<dd>the devotion of time and attention to acquiring knowledge on an academic subject, esp. by means of books</dd>
<dt>design</dt>
<dd>a plan or drawing produced to show the look and function or workings of a building, garment, or other object before it is built or made</dd>
<dd>purpose, planning, or intention that exists or is thought to exist behind an action, fact, or material object</dd>
<dt>business</dt>
<dt>work</dt>
<dd>a person's regular occupation, profession, or trade</dd>
</dl>
```

## **2.11 HTML Block and Inline Elements**

Every HTML element has a default display value, depending on what type of element it is.

There are two display values: block and inline.

#### Block-level Elements

A block-level element always starts on a new line, and the browsers automatically add some space (a margin) before and after the element.

A block-level element always takes up the full width available (stretches out to the left and right as far as it can).

Two commonly used block elements are: <p> and <div>.

- The <p> element defines a paragraph in an HTML document.
- The <div> element defines a division or a section in an HTML document.

The <p> element is a block-level element.

The <div> element is a block-level element.

### Inline Elements

An inline element does not start on a new line. An inline element only takes up as much width as necessary.

This is a <span> element inside a paragraph

### The <div> Element

The <div> element is often used as a container for other HTML elements. The <div> element has no required attributes, but style, class and id are common.

```
<!DOCTYPE html>
<html>
<body>
<div style="background-color:black;color:white;padding:20px;">
<h2>Demo</h2>
<p>This is a demo for block element in HTML</p>
</div>
</body>
</html>
```

#### Output

Demo

This is a demo for block element in HTML

### The <span> Element

The <span> element is an inline container used to mark up a part of a text, or a part of a document.

The <span> element has no required attributes, but style, class and id are common.

When used together with CSS, the <span> element can be used to style parts of the text:

Example

```
<!DOCTYPE html>
<html>
<body>
<h1>The span element</h1>
<p>The sky is <span style="color:blue;font-weight:bold">blue</span> in color and the trees are <span style="color:darkolivegreen;font-weight:bold">dark green</span> in color. </p>
</body>
```

</html>

**Output**

## The span element

The sky is **blue** in color and the trees are **dark green** in color.

## 2.12 HTML Layout Elements and Techniques

Websites often display content in multiple columns (like a magazine or a newspaper).

### HTML Layout Elements

HTML has several semantic elements that define the different parts of a web page:



- **<header>** - Defines a header for a document or a section
- **<nav>** - Defines a set of navigation links
- **<section>** - Defines a section in a document
- **<article>** - Defines an independent, self-contained content
- **<aside>** - Defines content aside from the content (like a sidebar)
- **<footer>** - Defines a footer for a document or a section
- **<details>** - Defines additional details that the user can open and close on demand
- **<summary>** - Defines a heading for the **<details>** element

### Page Layout Information:

**Header:** The part of a front end which is used at the top of the page. **<header>** tag is used to add header section in web pages.

**Navigation bar:** The navigation bar is same as menu list. It is used to display the content information using hyperlink.

**Index / Sidebar:** It holds additional information or advertisements and is not always necessary to be added into the page.

**Content Section:** The content section is the main part where content is displayed.

**Footer:** The footer section contains the contact information and other query related to web pages. The footer section always put on the bottom of the web pages. The **<footer>** tag is used to set the footer in web pages.



### Example

```

<!DOCTYPE html>
<html lang="en">
<head>
<title>CSS Template</title>
<meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<style>
* {
  box-sizing: border-box;
    
```

```
}
```

```
body {  
    font-family: Arial, Helvetica, sans-serif;  
}  
  
/* Style the header */  
header {  
    background-color: #666;  
    padding: 30px;  
    text-align: center;  
    font-size: 35px;  
    color: white;  
}  
  
/* Create two columns/boxes that floats next to each other */  
nav {  
    float: left;  
    width: 30%;  
    height: 300px; /* only for demonstration, should be removed */  
    background: #ccc;  
    padding: 20px;  
}  
  
/* Style the list inside the menu */  
nav ul {  
    list-style-type: none;  
    padding: 0;  
}  
  
article {  
    float: left;  
    padding: 20px;  
    width: 70%;  
    background-color: #f1f1f1;  
    height: 300px; /* only for demonstration, should be removed */  
}  
  
/* Clear floats after the columns */  
section::after {
```

```
content: "";
display: table;
clear: both;
}

/* Style the footer */
footer {
background-color: #777;
padding: 10px;
text-align: center;
color: white;
}

/* Responsive layout - makes the two columns/boxes stack on top of each other instead of next to
each other, on small screens */
@media (max-width: 600px) {
nav, article {
width: 100%;
height: auto;
}
}

</style>
</head>
<body>

<h2>Layout in HTML</h2>


In this example, we have created a header, two columns/boxes and a footer. On smaller screens, the columns will stack on top of each other.</p>
<header>
<h2>HTML</h2>
</header>

<section>
<nav>
<ul>
<li><a href="#">Home</a></li>
<li><a href="#">Gallery</a></li>
<li><a href="#">Contact us</a></li>
</ul>
</nav>


```

```

<article>
<h1>Layout</h1>
<p>Page layout is the part of graphic design that deals with the arrangement of visual elements on a page. Page layout is used to make the web pages look better. It establishes the overall appearance, relative importance, and relationships between the graphic elements to achieve a smooth flow of information and eye movement for maximum effectiveness or impact</p>
</article>
</section>

<footer>
<p>Footer</p>
</footer>

</body>
</html>

```

#### Layout In HTML

In this example, we have created a header, two column sections and a footer. On smaller screens, the columns will stack on top of each other.



## 2.13 HTML Forms

HTML stands for HyperText Markup Language. It is used to design web pages using a markup language. It is a combination of Hypertext and Markup language. HTML uses predefined tags and elements that tell the browser how to properly display the content on the screen. And form is one of them. So, in this article, we will learn what is exactly HTML form what are the elements of forms and how can we use HTML form in our webpage.

### What is HTML <form>?

<form> is a HTML element to collect input data with containing interactive controls. It provides facilities to input text, number, values, email, password, and control fields such as checkboxes, radio buttons, submits buttons, etc. or in other words, form is a container that contains input elements, like text, email, number, radio buttons, checkboxes, submit buttons, etc. Forms are generally used when you want to collect data from the user. For example, a user wants to buy a bag online, so he/she has to first enter their shipping address in the address form and then add their payment detail in the payment form to place an order.

Forms are created by placing input fields within paragraphs, preformatted text, lists and tables. This gives considerable flexibility in designing the layout of forms.

### Syntax:

```
<form>
<!--form elements-->
</form>
```

#### Form elements

These are the following HTML <form> elements:

- <label>: It defines label for <form> elements.
- <input>: It is used to get input data from the form in various type such as text, password, email, etc by changing its type.
- <button>: It defines a clickable button to control other elements or execute a functionality.
- <select>: It is used to create a drop-down list.
- <textarea>: It is used to get input long text content.
- <fieldset>: It is used to draw a box around other form elements and group the related data.
- <legend>: It defines caption for fieldset elements.
- <datalist>: It is used to specify pre-defined list options for input controls.
- <output>: It displays the output of performed calculations.
- <option>: It is used to define option in drop-down list.
- <optgroup>: It is used to define group related options in a drop down list.

### Textbox in HTML Form

In an HTML form, we use <input> tag by assigning type attribute value to text to input single line input. To define type attribute see the below syntax.

#### Syntax:

```
<input type="text" />
```

Or shorthand for "text" type:

```
<input />
```

### Password in an HTML Form

We can change type value text to password to get the input password



#### Example

```
<!DOCTYPE html>
<html>
<head>
<title>Page Title</title>
</head>
<body>
<h2>Welcome To HTML Forms</h2>
<form>
<p>
<label>Username : <input type="text" /></label>
</p>
<p>
<label>Password : <input type="password" /></label>
</p>
<p>
<button type="submit">Submit</button>

```

```

</p>
</form>
</body>
</html>

```

## Welcome To HTML Forms

Username :

Password :

### Radio Button in an HTML Form

To create a radio button, we use the `<input>` tag following by radio type to provide users to choose a limited number of choices.

Syntax:

```
<input type="radio" name="radio_button_name" value="radio_button_value" />
```

**Note:** The radio button must have shared the same name to be treated as a group.

**Note:** The value attribute defines the unique value associated with each radio button. The value is not shown to the user, but is the value that is sent to the server on “submit” to identify which radio button that was selected.



Example:

```

<!DOCTYPE html>
<html>
<head>
<title>Page Title</title>
</head>
<body>
<h2>Select your gender</h2>
<form>
    <label>Male<input type="radio" name="gender" value="male" /></label>
    <label>Female<input type="radio" name="gender" value="female" /></label>
</form>
</body>
</html>

```

### Select your gender

Male  Female

In this example, we will create a radio button to choose your gender.

### Checkbox in an HTML Form

To create a checkbox in an HTML form, we use the <input> tag following by the input type checkbox. It is a square box to ticked to activate this. It used to choose more options at a time.

#### Syntax:

```
<input type="checkbox" name="select_box_name" value="select_box_value" />
```

Note: the "name" and "value" attributes are used to send the checkbox data to the server.



#### Example:

In this example, we use checkboxes to select language.

```
<!DOCTYPE html>
<html>
<head>
<title>Page Title</title>
</head>
<body>
<h2>Choose Language</h2>
<form>
<ul style="list-style-type:none;">
<li><input type="checkbox" name="language" value="hindi" />Hindi</li>
<li><input type="checkbox" name="language" value="english" />English</li>
<li><input type="checkbox" name="language" value="sanskrite" />Sanskrit</li>
</ul>
</form>
</body>
</html>
```

### Choose Language

- Hindi
- English
- Sanskrit

## Combobox in an HTML Form

Combobox is used to create a drop-down menu in your form which contains multiple options. So, to create an Combobox in an HTML form, we use the `<select>` tag with `<option>` tag. It is also known as a drop-down menu.

Syntax:

```
<select name="select_box_name">
<option value="value1">option1</option>
<option value="value2">option2</option>
<option value="value3">option3</option>
</select>
```



**Example:**

```
<!DOCTYPE html>
<html>
<head>
<title>Page Title</title>
</head>
<body>
<h2>Select Your Nationality</h2>
<form>
<select name="language">
    <option value="indian">Indian</option>
    <option value="nepali">Nepali</option>
    <option value="others">Others</option>
</select>
</form>
</body>
</html>
```

## Select Your Nationality



## Submit button in an HTML Form

In the HTML form, submit button is used to submit the details of the form to the form handler. A form handler is a file on the server with a script that is used to process input data.

Syntax:

```
<button type="submit">submit</button>
<!DOCTYPE html>
<html>
<head>
<title>Page Title</title>
```

```

</head>
<body>
<h2>Welcome To HTML forms</h2>
<form>
<p>
    <label>Username: <input type="text" /></label>
<p>
    <label>Password: <input type="password" /></label>
</p>
<p>
    <button type="submit">submit</button>
</p>
</form>
</body>
</html>

```

## Welcome To HTML forms

Username:

Password:

### TextArea in an HTML Form

In the HTML form, a text area is used to add comments or reviews, or addresses in the form. Or in other words, the text area is a multi-line text input control. It contains an unlimited number of characters, and the text renders in a fixed-width font and the size of the text area is given by the `<rows>` and `<cols>` attributes. To create a text area in the form use the `<textarea>` tag.

Syntax:

```

<textarea name="textarea_name">content</textarea>
<!DOCTYPE html>
<html>
<head>
<title>Page Title</title>
</head>
<body>
<h2>Welcome To HTML Forms</h2>
<form>
    <textarea      name="welcomeMessage"      rows="3"      cols="40">Kindly      give      your
feedback</textarea>
</form>

```

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

## Welcome To HTML Forms

Kindly give your feedback

### **2.14 Iframes in HTML**

HTML Iframe is used to display a nested webpage (a webpage within a webpage). The HTML <iframe> tag defines an inline frame, hence it is also called as an Inline frame.

An HTML iframe embeds another document within the current HTML document in the rectangular region.

The webpage content and iframe contents can interact with each other using JavaScript.

#### **Iframe Syntax**

An HTML iframe is defined with the <iframe> tag:

```
<iframesrc="URL"></iframe>
```

Here, "src" attribute specifies the web address (URL) of the inline frame page.

#### **Set Width and Height of iframe**

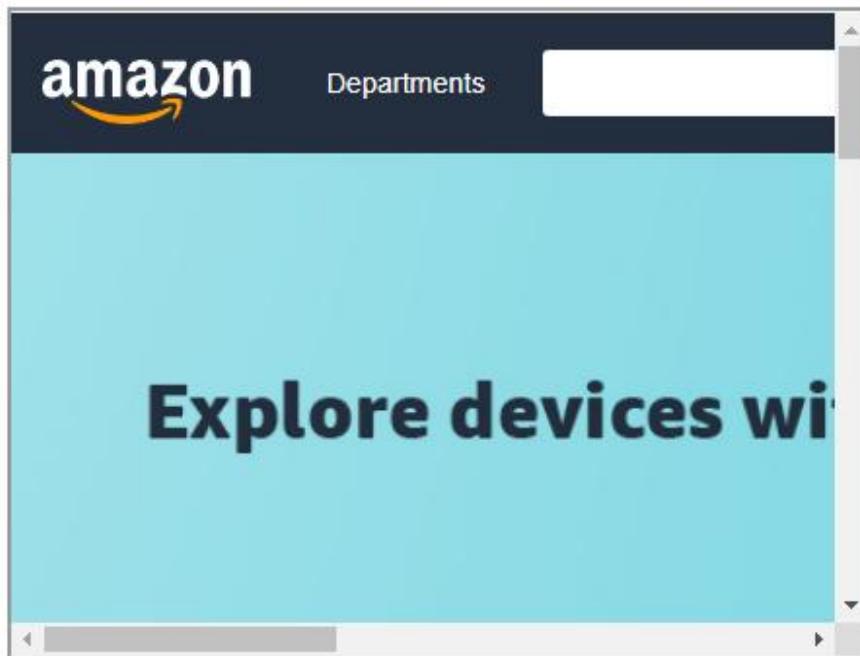
You can set the width and height of iframe by using "width" and "height" attributes. By default, the attributes values are specified in pixels but you can also set them in percent. i.e. 50%, 60% etc.

#### **Example: (Pixels)**

```
<!DOCTYPE html>  
<html>  
<body>  
<h2>HTML Iframes example</h2>  
<p>Use the height and width attributes to specify the size of the iframe:</p>  
<iframesrc="https://www.amazon.com/" height="300" width="400"></iframe>  
</body>  
</html>
```

## HTML Iframes example

Use the height and width attributes to specify the size of the iframe:



You can also change the size, color, style of the iframe's border.

### Embed YouTube video using iframe

You can also add a YouTube video on your webpage using the <iframe> tag. The attached video will be played at your webpage and you can also set height, width, autoplay, and many more properties for the video.

Following are some steps to add YouTube video on your webpage:

- Goto YouTube video which you want to embed.
- Click on SHARE ➔ under the video.
- Click on Embed <> option.
- Copy HTML code.
- Paste the code in your HTML file
- Change height, width, and other properties (as per requirement).



### Example

```
<!DOCTYPE html>
<html>
<head>

</head>
<body style="background-color: #f0f8ff">
    <h3>Play videos using iframe</h3>
```

Unit 02: HTML Forms and Frames

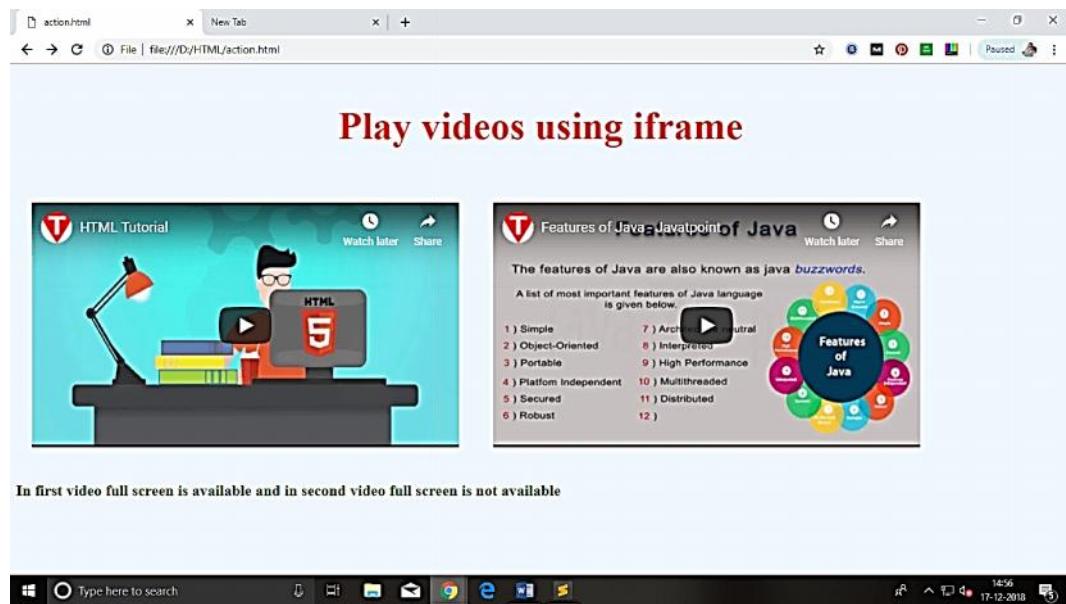
```
<iframe width="550" height="315"
src="https://www.youtube.com/embed/JHq3pL4cdy4" frameborder="0" allow="accelerometer;
autoplay; encrypted-media; gyroscope; picture-in-picture" allowfullscreen
style="padding:20px;"></iframe>
```

```
<iframe width="550" height="315" src="https://www.youtube.com/embed/O5hShUO6wxs"
frameborder="0" allow="accelerometer; autoplay; encrypted-media; gyroscope; picture-in-picture"
style="padding:20px;"></iframe>
```

<p>In first video full screen is available and in second video full screen is not available</p>

</body>

</html>

Summary

- Lists commonly are found in documents, including web pages.
- The HTML List classes allow you to easily create lists within your HTML pages.
- HTML List provides a method to produce a compact list that displays items in as small a vertical space as possible.
- HTML provides three different types of lists to choose from when building a page, including unordered, ordered, and definition lists.
- Ordered lists place strong importance on the order of items.
- Unordered lists are purely a list of related items, in which their order does not matter nor do they have a numbered or alphabetical list element.
- A definition list may contain numerous terms and descriptions, one after the other.
- One of the best ways to create an impact with your web page is to add some graphics.
- Tables are defined with the <table> tag.
- Constructing an HTML table consists of describing the table between the beginning table tag, <TABLE>, and the ending table tag, </TABLE>.
- Links are the essence of HTML — they are what makes it unique. Links are defined with the <a> tag.

- If you do not specify a border attribute the table will be displayed without any borders.
- <Img> causes an “inline image” to be inserted into the output.
- Image maps are images with clickable areas (sometimes referred to as “hotspots”) that usually link to another page.
- Email links are done much the same as links to other pages, using the <a href> tag.
- The key difference between an inline image and an image retrieved with the <a> tag is that an inline image requires no action on the part of the reader

## **Keywords**

- <IMG>: <IMG> indicates that an image—such as a photograph, icon, animation, cartoon, or other graphic—is to be displayed at that location.
- DL: Definition lists, created using the DL element, generally consist of a series of term/definition Pairs. HTML List classes: It allows you to easily create lists within your
- HTML pages. Inline image: An image which is displayed on a web browser is referred to as an “inline image”.
- alt: alt is used to provide an text alternative to the image for readers whose browsers do not support graphics.
- Cell tags: <TD></TD> Image links: Image links are constructed as you might expect, by embedding an <img> tag inside of an anchor element <a>.
- Links: Links are the essence of HTML — they are what makes it unique
- Row tags: <TR></TR>
- List tag: An empty tag called a “list” tag is used to do itemize elements of “unordered” and “ordered” lists.
- OL: An ordered list, created using the OL element, should contain information where order should be emphasized.
- Ordered List Item: It allows you to override the numbering and type for a specific item in the list. Unordered lists: Unordered lists are for lists of items where order isn’t of important.

## **Self Assessment**

1. Which HTML tag is used to define a table?
  - A. <tb>
  - B. <tl>
  - C. <table>
  - D. <tab>
  
2. With the help of which tag, is a row defined in HTML?
  - A. <row>
  - B. <table-row>
  - C. <tablerow>
  - D. <tr>

3. Choose the correct option.
- <th> is used for defining the heading of a table.
  - By default, contents written between <th> and </th> are bold and centered.
  - Both A and B
  - None Of these

4. Fill in the blanks with the help of options given below in order to get the following table when the below code is executed.

```
<!DOCTYPE html>
<html>
<body>
<table border="2">
<tr>
<th>Name</th>
<th>Phone no</th>
</tr>
<tr>
<td _____>John</td>
<td>9898989898</td>
</tr>
<tr>
<td>9876543210</td>
</tr>
</body>
</html>
```

Name	Phone no
John	9898989898
	9876543210

- colspan= "1"
- colspan= "2"
- rowspan= "1"
- rowspan= "2"

5. Which one of the following tags is used to add caption to a table?

- <table-caption>
- <tcaption>
- <caption>
- <tc>

6. Each cell of the table can be represented by using \_\_\_\_\_

- <tr>
- <td>
- <th>
- <thead>

7. The \_\_\_\_\_ tag defines an image in an HTML page.

- <image>
- <pic>

C <imge>

D <img>

8. Which of the following pair of attribute is required for img tag ?

- A.src and a
- B. img and src
- C. img and alt
- D. src and alt

9. If the image cannot be displayed then \_\_\_\_\_ specifies an alternate text for an image.

- A. text attribute
- B. value attribute
- C. alt attribute
- D. caption attribute

10. Alt Attribute is more useful in the situation where user have \_\_\_\_\_.

- A. High Speed Internet Connection
- B. Broadband Connection
- C. None of these
- D. Slow Internet Connection

11. Which one of the following is a type of lists that HTML supports?

- A. Ordered lists.
- B. Unordered lists.
- C. Description lists.
- D. All of the above

12. By which tag, an unordered list is represented?

- A. <u>
- B. <I>
- C. <ul>
- D. <ol>

13. By which tag, an ordered list is represented?

- A. <u>
- B. <I>
- C. <ul>
- D. <ol>

14. A HTML code is given below

Fill in the blanks with appropriate option to get the output as below:

```
<!DOCTYPE html>
<html>
<body>
<dl>
    _____
Mathematics
    _____
Calculus
</dl>
</body>
</html>
```

- A <dd>,</dd>,<dt>,</dt>
- B. <dt>,</dt>,<dd>,</dd>
- C. <li>,</li>,<dd>,</dd>
- D. <dt>,</dt>,<li>,</li>

15. What is the default item marker in unordered lists of HTML?

- A. Circle
- B. Marker
- C. disc
- D. None of the above

### **Answers for Self Assessment**

- |       |       |       |       |       |
|-------|-------|-------|-------|-------|
| 1. C  | 2. D  | 3. C  | 4. D  | 5. C  |
| 6. B  | 7. C  | 8. B  | 9. C  | 10. D |
| 11. D | 12. C | 13. D | 14. B | 15. C |

### **Review Questions**

1. On what information we should emphasize while preparing an order list? ‘
2. Discuss the methods for HTML List.
3. Discuss the methods to create ordered lists, unordered lists, and nested lists.
4. Discuss IMG Attributes.
5. Explain with examples about graphic image alignment parameters in the HTML.
6. Explain the three different types of HTML lists.
7. Explain how HTML Tables are created?
8. Define and Explain the steps involved in creating HTML Tables.
9. Discuss in brief about Linking Document.
10. What are the three attributes that can be specified with the <BODY> tag? Explain each of them.

11. Define and explain Hyperlinks and their types.
12. . Justify the use of link list creation in html document.
13. Explain how to display a table with borders.
14. . Discuss how to link to a page on another web site. Illustrate with example.
15. Explain the steps used to move to a specific location on a Web page



## **Further Readings**

Hall, 2009, Core Web Programming, 2/E, Pearson Education India. Jon Duckett, 2011, Beginning Web Programming with HTML, XHTML and CSS,

John Wiley & Sons. Robert F. Breedlove, 1996, Web Programming Unleashed, Sams.net.

Tim Downey, 2012, Guide to Web Development with Java: Understanding Website Creation, Springer.



## **Web Links**

<http://www.temple.edu/cs/web/tables.html>

[http://www.w3schools.com/html/tryit.asp?filename=tryhtml\\_imglink](http://www.w3schools.com/html/tryit.asp?filename=tryhtml_imglink)

<http://www.tizag.com/htmlT/htmlimagelinks.php>

[http://www.quackit.com/html/tutorial/html\\_image\\_maps.cfm](http://www.quackit.com/html/tutorial/html_image_maps.cfm)

<http://www.echoecho.com/htmllinks11.htm>

<http://interestingwebs.blogspot.in/2008/12/how-to-create-scrollable-linklist.html>

<http://www-sul.stanford.edu/tools/tutorials/html2.0/img.html>

## Unit 03: HTML Colors and XHTML

### **CONTENTS**

Objectives

Introduction

3.1     HTML Color Coding Methods

3.2     HTML Colors - Color Names

3.3     HTML HEX Colors

3.4     HTML Colors - RGB Values

3.5     What is XHTML?

3.6     Difference between HTML and XHTML

3.7     XHTML Syntax

3.8     XHTML Event

3.9     Keyboard Events

Summary

Keywords

Self Assessment

Answers for Self Assessment

Review Questions

Further Readings

### **Objectives**

- To learn about different HTML color tags
- Understand the usage of different color along with its code.
- Acquire the knowledge about HTML vs XHTML

### **Introduction**

Colors are very important to give a good look and feel to your website. You can specify colors onpage level using <body> tag or you can set colors for individual tags using bgcolor attribute.

The <body> tag has following attributes which can be used to set different colors:

- **bgcolor** - sets a color for the background of the page.
- **text** - sets a color for the body text.
- **alink** - sets a color for active links or selected links.
- **link** - sets a color for linked text.
- **vlink** - sets a color for visited links - that is, for linked text that you have already clicked on.

### **3.1     HTML Color Coding Methods**

There are following three different methods to set colors in your web page:

- **Color names** - You can specify color names directly like green, blue or red.

- **Hex codes** - A six-digit code representing the amount of red, green, and blue that makes up

the color.

- **Color decimal or percentage values** - This value is specified using the `rgb` property.

Now we will see these coloring schemes one by one.

### **3.2 HTML Colors - Color Names**

You can specify direct a color name to set text or background color. W3C has listed 16 basic colors names that will validate with an HTML validator but there are over 200 different color names supported by major browsers. Below is the list of standard colors used in HTML

	Black		Gray		Silver		White
	Yellow		Lime		Aqua		Fuchsia
	Red		Green		Blue		Purple
	Maroon		Olive		Navy		Teal

Figure 0.1 16 basic colors used in HTML



#### **Example**

Below is the example used for setting a background using HTML color tag

```
<body><p style="color:red;">Red paragraph text</p></body>
```

### **3.3 HTML HEX Colors**

A hexadecimal color is specified with: #RRGGBB, where the RR (red), GG (green) and BB (blue) hexadecimal integers specify the components of the color.

#### **HEX Color Values**

In HTML, a color can be specified using a hexadecimal value in the form:

#rrggbb

Where rr (red), gg (green) and bb (blue) are hexadecimal values between 00 and ff (same as decimal 0-255).

For example, #ff0000 is displayed as red, because red is set to its highest value (ff), and the other two (green and blue) are set to 00.

Another example, #00ff00 is displayed as green, because green is set to its highest value (ff), and the other two (red and blue) are set to 00.

- To display black, set all color parameters to 00, like this: #000000.

- To display white, set all color parameters to ff, like this: #ffffff.

A hexadecimal is a 6 digit representation of a color. The first two digits represent a red value, the next two are a green value, and the last are the blue value.

A hexadecimal value can be taken from any graphics software like Adobe Photoshop, Paintshop

Pro or MS Paint.

Each hexadecimal code will be preceded by a pound or hash sign #. Following is a list of few colors using hexadecimal notation.



### Example

Color	Color HEX
Black	#000000
Red	#FF0000
Green	#00FF00
Blue	#0000FF
Yellow	#FFFF00
Cyan	#00FFFF
Magenta	#FF00FF
Grey	#C0C0C0
White	#FFFFFF

Figure 0.2 Example of Hex color code



### Example

```
<!DOCTYPE html>
<html>
<head>
<title>HTML Colors by Hex</title>
</head>
<body text="#0000FF" bgcolor="#00FF00">
<p>Use different color hexa for body and table and see the result.</p>
<table bgcolor="#000000">
<tr>
<td>
<font color="#FFFFFF">This text will appear white on black background.</font>
</td>
</tr>
</table>
</body>
</html>
```

## 3.4 HTML Colors - RGB Values

This color value is specified using the rgb property. This property takes three values, one each for red, green, and blue. The value can be an integer between 0 and 255 or a percentage.

*Notes:* All the browsers does not support rgb property of color so it is recommended not to use it.

### List of colors with RGB values

Following is a list to show few colors using RGB values.

Color	Color RGB
Black	rgb(0, 0, 0)
Red	rgb(255, 0, 0)
Green	rgb(0, 255, 0)
Blue	rgb(0, 0, 255)
Yellow	rgb(255, 255, 0)
Cyan	rgb(0, 255, 255)
Magenta	rgb(255, 0, 255)
Grey	rgb(192, 192, 192)
White	rgb(255, 255, 255)



### Example

Here are the examples to set background of an HTML tag by color code using rgb values:

```
<!DOCTYPE html>
<html>
<head>
<title>HTML Colors by RGB code</title>
</head>
<body text="rgb(0,0,255)" bgcolor="rgb(0,255,0)">
<p>Use different color code for for body and table and see the result.</p>
<table bgcolor="rgb(0,0,0)">
<tr>
<td>
<font color="rgb(255,255,255)">This text will appear white on black background.</font>
</td>
</tr>
</table>
</body>
</html>
```

## 3.5 What is XHTML?

XHTML stands for EXtensibleHyperText Markup Language. It is a cross between HTML and XML language.

XHTML is almost identical to HTML but it is stricter than HTML. XHTML is HTML defined as an XML application. It is supported by all major browsers.

Although XHTML is almost the same as HTML but It is more important to create your code correctly, because XHTML is stricter than HTML in syntax and case sensitivity. XHTML documents are well-formed and parsed using standard XML parsers, unlike HTML, which requires a lenient HTML-specific parser.

### Why we use XHTML?

XHTML was developed to make HTML more extensible and increase interoperability with other data formats. There are two main reasons behind the creation of XHTML:

It creates a stricter standard for making web pages, reducing incompatibilities between browsers. So it is compatible for all major browsers.

It creates a standard that can be used on a variety of different devices without changes.

### Let's take an example to understand it.

HTML is mainly used to create web pages but we can see that many pages on the internet contain "bad" HTML (not follow the HTML rule).

This HTML code works fine in most browsers (even if it does not follow the HTML rules).

```
<html>
<head>
<title>This is an example of bad HTML</title>
<body>
<h1>Bad HTML
<p>This is a paragraph
</body>
```

The above HTML code doesn't follow the HTML rule although it runs. Now a day, there are different browser technologies. Some browsers run on computers, and some browsers run on mobile phones or other small devices. The main issue with the bad HTML is that it can't be interpreted by smaller devices.

So, XHTML is introduced to combine the strengths of HTML and XML.

XHTML is HTML redesigned as XML. It helps you to create better formatted code on your site.

XHTML doesn't facilitate you to make badly formed code to be XHTML compatible. Unlike with HTML (where simple errors (like missing out a closing tag) are ignored by the browser), XHTML code must be exactly how it is specified to be.

### **Advantages of XHTML**

- Here are the following advantages of XHTML, such as:
- While using XHTML, the code of web applications becomes more stylish and easy to reuse.
- It can help the developer create more advanced web projects due to the compatibility with various devices, and it also supports self-created markups like SVG (scalable vector graphics).
- XHTML code can easily be converted to PDFs, RSS, and RFT, which allows the developer to work with a vast range of files.
- XHTML reduce the loading time required by the browser to load an event which can result in overall speedy development, thus reducing time and energy
- It contains closing tags which is an advantage for beginners, and this also makes the code look clean and easy to reuse.

### **Disadvantages of XHTML**

XHTML also has some disadvantages, such as:

- Very few browsers use XHTML.
- Case sensitive as every part of code should be in lowercase.
- It is mandatory to write <DOCTYPE> declaration.
- And all the tags must be closed in the necessary order.

### **3.6 Difference between HTML and XHTML**

HTML and XHTML are both markup languages used to create web pages and applications. HTML and XHTML have some key differences that set them apart. Here are the following major differences between HTML and XHTML:

S.No.	HTML	XHTML
1.	Hypertext mark-up language --> HTML	Extensible Hypertext Mark-up Language --> XHTML.
2.	Tim Berners created in 1991	World wide web consortium or W3C created in 2000
4.	It is an extension of standard generalized markup language or SGML	It is a combination of extensible markup language XML and hypertext markup language HTML
5.	It stored in a document file format	It stored as a markup language format
6.	It is not case sensitive as there is no mandatory rule to write the entire mark up in uppercase or lower case. It can also be a combination of both.	It is case-sensitive, and every tag and attribute used inside must be in lowercase.
7.	It is not mandatory to add document label < DOCTYPE > at the top of every page. We can even skip it.	It is mandatory to add a document label < DOCTYPE > at the beginning of the page.
8.	We can close any tag anytime and anywhere as per our needs	It is mandatory to close all the tags in strict residing order as they were declared.
9.	We can add attributes without any quotes.	It is mandatory to add quotes on every attribute we declare
10.	.html and .htm are the extensions used by HTML	.xhtml, .xml and .xht are the file extensions used by XHTML
11	Lewd structure is used	It contains a very strict structure, and the developer cannot go out of the bounds of these structures.

### 3.7 XHTML Syntax

XHTML syntax is very similar to HTML syntax and all the valid HTML elements are also valid in XHTML. But XHTML is case sensitive so you have to pay a bit extra attention while writing an XHTML document to make your HTML document compliant to XHTML.

You must remember the following important points while writing a new XHTML document or converting existing HTML document into XHTML document:

- **All documents must have a DOCTYPE.:**

All XHTML documents must contain a DOCTYPE declaration at the start. There are three types of DOCTYPE declarations:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
```

- **All tags must be in lower case:**

XHTML is case-sensitive markup language. So, all the XHTML tags and attributes must be written in lower case.

```
<!-- Invalid in XHTML -->
<a href="/xhtml/xhtml_tutorial.html">XHTML Tutorial</a>
<!-- Valid in XHTML -->
<a href="/xhtml/xhtml_tutorial.html">XHTML Tutorial</a>
```

- **All documents must be properly formed**
- **All tags must be closed:**
- An XHTML must have an equivalent closing tag. Even empty elements should also have closing tags. Let's see an example:

```
<!-- Invalid in XHTML -->
<p>This paragraph is not written according to XHTML syntax.
<!-- Invalid in XHTML -->
<imgsrc="/images/xhtml.gif" >
<!-- Valid in XHTML -->
<p>This paragraph is not written according to XHTML syntax.</p>
<!-- Valid in XHTML-->
<imgsrc="/images/xhtml.gif" />
```

- **All attributes must be added properly:**

All the XHTML attribute's values must be quoted. Otherwise, your XHTML document is assumed as an invalid document.

```
<!-- Invalid in XHTML -->
<imgsrc="/images/xhtml.gif" width=250 height=50 />
<!-- Valid in XHTML -->
<imgsrc="/images/xhtml.gif" width="250" height="50" />
```

- **The name attribute has changed.**

- **Attributes cannot be shortened.**
- **All tags must be properly nested.**
- **Attribute Minimization:**

XHTML doesn't allow you to minimize attributes. You have to explicitly state the attribute and its value.

See this example:

```
<!--Invalid in XHTML-->
<option selected>
<!-- valid in XHTML-->
<option selected="selected">
```

### 3.8 XHTML Event

When you visit a website, you do things like click on text, images and hyperlinks, hover-over things, etc. These are examples of what JavaScript calls events.

We can write our event handlers in JavaScript or VBScript and can specify these event handlers as a value of event tag attribute. The XHTML 1.0 has a similar set of events which is available in HTML 4.01 specification.

#### The <body> and <frameset> Level Events

There are only two attributes which are used to trigger any JavaScript or VBScript code, when any event occurs at document level.

Attribute	Value	Description
onload	Script	Script runs when a XHTML document loads.
onunload	Script	Script runs when a XHTML document unloads.

#### The <form> Level Events

There are six attributes which are triggered when any event occurs at form level.

Attribute	Value	Description
onchange	Script	It is executed when the element changes.
onsubmit	Script	It is executed when the form is submitted.
onreset	Script	It is executed when the form is reset.

onselect	Script	It is executed when the element is selected.
onblur	Script	It is executed when the element loses focus.
onfocus	Script	It is executed when the element gets focus.

### 3.9 Keyboard Events

There are three events which are generated by keyboard. The keyboard events are not valid in base, bdo, br, frame, frameset, head, html, iframe, meta, param, script, style, and title elements.

Attribute	Value	Description
onkeydown	Script	This is executed when the user press the keyboard button.
onkeypress	Script	This is executed when the user press and release the keyboard button.
onkeyup	Script	This is executed when the user release the keyboard button.

### Mouse Events

There are some mouse generated events which executes when it comes in contact with any HTML tag. These events are not valid in base, bdo, br, frame, frameset, head, html, iframe, meta, param, script, style, and title elements.

Attribute	Value	Description
onclick	Script	It is executed on a mouse click.
ondblclick	Script	It is executed on a mouse double-click.
onmousedown	Script	It is executed when mouse button is pressed.
onmousemove	Script	It is executed when mouse pointer moves.
onmouseout	Script	It is executed when mouse pointer moves out of an element.

onmouseover	Script	It is executed when mouse pointer moves over an element.
onmouseup	Script	It is executed when mouse button is released.

### **Summary**

- You can specify direct a color name to set text or background color
- #rrggbb, Where rr (red), gg (green) and bb (blue) are hexadecimal values between 00 and ff (same as decimal 0-255).
- Hue is a degree on the color wheel from 0 to 360. 0 is red, 120 is green, and 240 is blue.
- Saturation is a percentage value, 0% means a shade of gray, and 100% is the full color.
- Lightness is also a percentage value, 0% is black, and 100% is white.
- XHTML code can easily be converted to PDFs, RSS, and RFT, which allows the developer to work with a vast range of files.
- XHTML reduce the loading time required by the browser to load an event which can result in overall speedy development, thus reducing time and energy

### **Keywords**

- **Onkeydown:** This is executed when the user press the keyboard button.
- **Onkeypress:** This is executed when the user press and release the keyboard button.
- **Onkeyup:** This is executed when the user release the keyboard button.
- **XHTML:** XHTML stands for EXtensibleHyperText Markup Language. It is a cross between HTML and XML language. XHTML is almost identical to HTML, but it is stricter than HTML. XHTML is HTML defined as an XML application. It is supported by all major browsers.
- **Onchange:** It is executed when the element changes.
- **Onsubmit:** It is executed when the form is submitted.
- **Onreset:** It is executed when the form is reset.
- **Onselect:** It is executed when the element is selected.
- **Onblur:** It is executed when the element loses focus.
- **Onfocus:** It is executed when the element gets focus.

### **Self Assessment**

1. How the HEX value of color represented in HTML?
  - A. #rrggbb
  - B. #rrbbgg
  - C. #ggrrb
  - D. #brrrgg
  
2. Choose the correct option.
  - A. In HTML, rr,gg,bb of #rrggbb represents red, green and black color in hex values.
  - B. In HTML, rr,gg,bb of #rrggbb represents red, grey and black color in hex values.
  - C. In HTML, rr,gg,bb of #rrggbb represents red, grey and blue color in hex values.

- D. In HTML, rr,gg,bb of #rrggb represents red, green and blue color in hex values.
3. Fill in the blanks from one of the options given below so that the background of the paragraph is filled with color of color code #3cb371.

```
<!DOCTYPE html>
<html>
<body>
<p style="_____>A paragraph</p>
</body>
</html>
```

- A. background-color:hex(#3cb371);  
B. bg-color:hex(#3cb371);  
C. background-color:#3cb371;  
D. bg-color:#3cb371;
4. #ffffff is equivalent to which color in rgb color representation?  
A. rgb(0,0,0)  
B. rgb(100,100,100)  
C. rgb(90,90,90)  
D. rgb(255,255,255)
5. Which color is #ff0000?  
A. White  
B. Black  
C. Red  
D. Blue
6. Which color do RGB(0,0,0) represent?  
A. White  
B. Black  
C. Red  
D. Blue
7. Choose the correct option.  
A. In rgba, a stands for alpha.  
B. Its value ranges from 0 to 1.  
C. Both a and b are correct.  
D. None of the above
8. Which of the following is correct regarding s parameter in hsl in HTML?

A. s stands for saturation.

B. s determines the intensity of the color

C. None of a and b is correct.

D. Both a and b are correct.

9. What color does 0% saturation give?

A. pure color.

B. white

C. shades of grey.

D. None of the above

10. HTML and XHTML stands for \_\_\_\_\_

A. Hyper Text Markup Language and Extensible HyperText Markup Language

B. Hyper Text Markup Language and Extensible HyperText Marking Language

C. Hyper Text Marking Language and EXtensibleHyperText Marking Language

D. Hyper Text Marking Language and Extensible HyperText Markup Language

11. XHTML is almost same as which version of HTML?

A. HTML 1.0

B. HTML 2.01

C. HTML 3.0

D. HTML 4.01

12. Which of the following is not the correct rule for XHTML?

A. Attributes should be quoted

B. Tags should nest not tag

C. Unused elements may be minimized

D. Unknown attributes are ignored by the browser

13. Which elements are mandatory in an XHTML document?

A. doctype, html, head, body, and title

B. doctype, html and body

C. doctype, html, head, and body

D. doctype, html, title, and body

14. What XHTML code is "well-formed"?

A. <p>A <b><i>short</i></b> paragraph

B. <p>A <b><i>short</i></b> paragraph</p>

C. <p>A <b><i>short</b></i> paragraph</p>

D. None of these

15. Which of the following is the right use of the lang attribute?

- A. <div language="en">Hello World!</div>
- B. <div lang="en" xml:lang="en">Hello World!</div>
- C. <div xml:language="en">Hello World!</div>
- D. <div xml:lan="en">Hello World!</div>

### **Answers for Self Assessment**

- |       |       |       |       |       |
|-------|-------|-------|-------|-------|
| 1. A  | 2. D  | 3. C  | 4. D  | 5. C  |
| 6. B  | 7. C  | 8. D  | 9. C  | 10. A |
| 11. D | 12. D | 13. A | 14. B | 15. B |

### **Review Questions**

1. Explain HTML color tag with its attributes.
2. Differentiate between HTML and XHTML.
3. Explain how XHTML is better than HTML with examples
4. Elaborate RGB and HSL with the help of example



### **Further Readings**

<https://www.dcpehvp.org/E-Content/BCA/BCA-II/Web%20Technology/the-complete-reference-html-css-fifth-edition.pdf>



### **Web Links**

<https://wtf.tw/ref/duckett.pdf>

## **Unit 4:Introduction of HTML5**

### **CONTENTS**

Objectives

Introduction

2.1 History of HTML5

2.2 Exploring New Features of HTML 5

2.1.1 Difference Between HTML and HTML5

2.3 Anatomy of an HTML Tag

2.4 HTML Document Structure

2.5 HTML Content Model

2.6 HTML Character Entity

2.7 HTML Links

2.8 Canvas

2.9 SVG

Summary

Self Assessment

Answers for Self Assessment

Review Questions

Further Readings

### **Objectives**

After studying this unit, you will be able to:

- Discuss the History of HTML5
- How to use HTML Tag
- Describe the structure of HTML
- Explain Content Model
- Discuss SVG

### **Introduction**

HTML5 is a markup language that was founded in 2004 by the Web Hypertext Application Technology Working Group (WHATWG) whose members include Apple, Mozilla Foundation and Opera Software. Then in Oct 2006 W3C (World Wide Web Consortium) decided to stop their work on XHTML and start collaborating with "WHATWG" to develop HTML5 as a technology. After that, the first version of HTML5 was published in 2008 that was written by Ian Hickson, Google, ian@hixie.ch. But it's not completed and changes are still remaining because according to WHATWG experts "HTML5 is a continually evolving technology that will never end". After the first draft of HTML5, it's time to make compatible browser that supports HTML5 features. And then Mozilla took the first step and introduced Firefox 3 which allows users to view HTML5 in the browser, but Safari, Google Chrome, and IE were far behind in supporting HTML5 features in their browsers. HTML5 was gaining popularity day by day. Then in April 2010, Steve Jobs declared that Flash will never be allowed on Apple's smart devices. He said that "Flash was designed for PCs using a mouse, not for touch screens using fingers". This statement is enough to change the mind of many companies and that's the reason they began the development of HTML5.

## **4.1 History of HTML5**

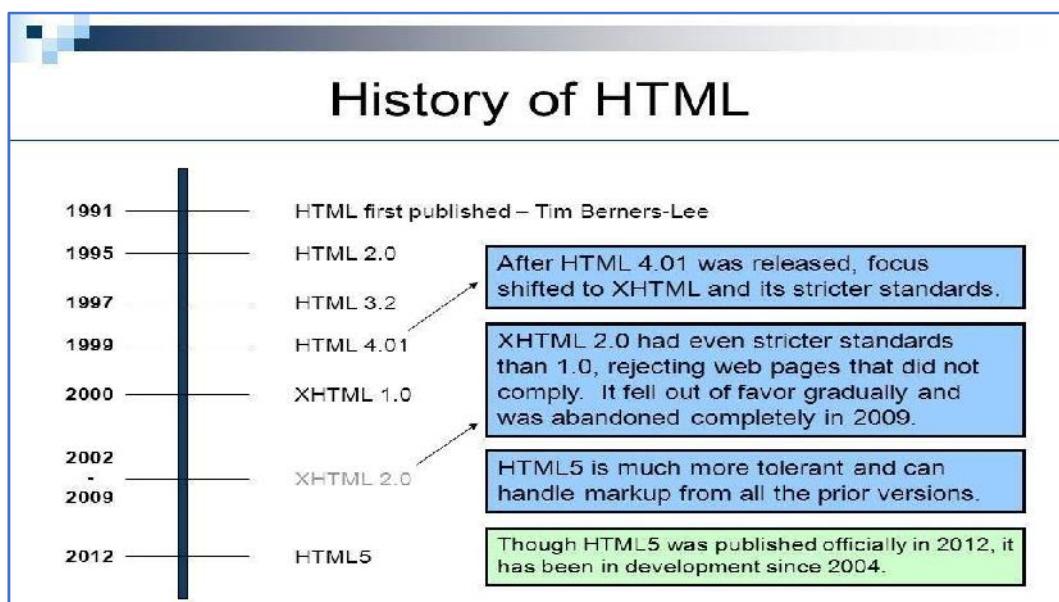
HTML 1.0 was released in 1993 with the intention of sharing information that can be readable and accessible by web browsers. But not many of the developers were involved in creating websites. So, the language was also not growing.

HTML 2.0, published in 1995, contains all the features of HTML 1.0 along with a few additional features, which remained the standard markup language for designing and creating websites until January 1997 and refined various core features of HTML.

HTML 3.0, where Dave Raggett introduced a fresh paper or draft on HTML. It included improved new features of HTML, giving more powerful characteristics for webmasters in designing web pages. But these powerful features of the new HTML slowed down the browser in applying further improvements.

HTML 4.01, which is widely used and was a successful version of HTML before HTML 5.0, which is currently released and used worldwide. HTML 5 can be said for an extended version of HTML 4.01, which was published in the year 2012.

HTML was created by Sir Tim Berners-Lee in late 1991 but was not released officially, published in 1995 as HTML 2.0. HTML 4.01 was published in late 1999 and was a major version of HTML.



## **4.2 Exploring New Features of HTML 5**

- 1) It has introduced new multimedia features which support audio and video controls by using <audio> and <video> tags.
- 2) There are new graphic elements including vector graphics and tags.
- 3) Enrich semantic content by including <header> <footer>, <article>, <section> and <figure> are added.
- 4) Drag and Drop - The user can grab an object and drag it further dropping it to a new location.
- 5) Geo-location services - It helps to locate the geographical allocation of a client.
- 6) Web storage facility which provides web application methods to store data on a web browser.
- 7) Uses SQL database to store data offline.

- 8) Allows drawing various shapes like triangles, rectangles, circles, etc.
- 9) Capable of handling incorrect syntax.
- 10) Easy DOCTYPE declaration i.e., <!doctype html>
- 11) Easy character encoding i.e., <meta charset="UTF-8">

### **4.3 Difference Between HTML and HTML5**

Features	HTML	HTML5
Definition	A hypertext markup language (HTML) is the primary language for developing web pages.	HTML5 is a new version of HTML with new functionalities with markup language with Internet technologies.
Multimedia support	Language in HTML does not have support for video and audio.	HTML5 supports both video and audio.
Storage	The HTML browser uses cache memory as temporary storage.	HTML5 has storage options like application cache, SQL database, and web storage.
Browser compatibility	HTML is compatible with almost all browsers because it has been present for a long time, and the browser has made modifications to support all the features.	In HTML5, we have many new tags, elements, and some tags that have been removed/modified, so only some browsers are fully compatible with HTML5.
Graphics support	In HTML, vector graphics are possible with tools like Silverlight, Adobe Flash, VML, etc.	In HTML5, vector graphics are supported by default.
Threading	In HTML, the browser interface and JavaScript run in the same thread.	The HTML5 has the JavaScript Web Worker API, which allows the browser interface to run in multiple threads.
Doctype	Doctype declaration in HTML lists too long <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">	The DOCTYPE declaration in HTML5 is very simple "<!DOCTYPE html>"
Character Encoding	Character encoding in HTML lists too long. <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">	Character encoding declaration is simple <meta charset="UTF-8">
Multimedia support	Audio and video are not the part of HTML4.	Audio and video are essential parts of HTML5, like: <Audio>, <Video>.



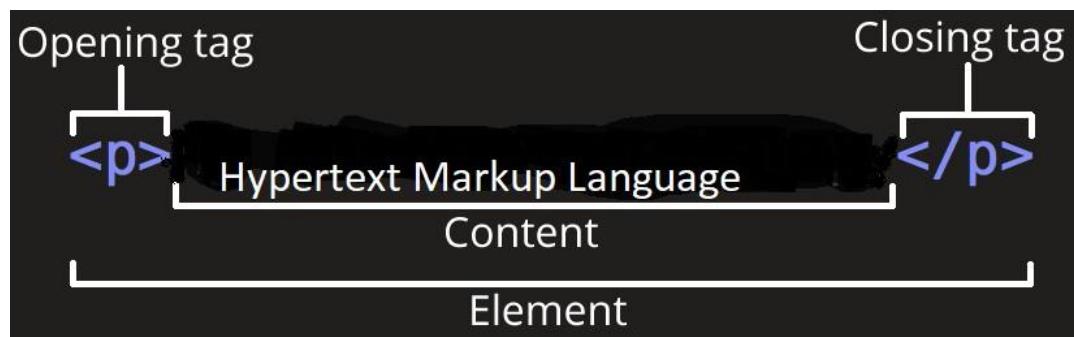
**Did you Know?** : HTML5 provides you with more features with respect to the HTML such as audio, video with the help of tags, drag and drop feature, Geolocation, browser support, etc.

#### **4.4 Anatomy of an HTML Tag**

HTML consists of a series of elements, which you use to enclose, or wrap, different parts of the content to make it appear a certain way, or act a certain way. The enclosing tags can make a word or image a hyperlink to somewhere else, can italicize words, can make the font bigger or smaller, and so on. For example, **Hypertext Markup Language**

If we wanted the line to stand by itself, we could specify that it is a paragraph by enclosing it in paragraph tags:

```
<p>Hypertext Markup Language</p>
```



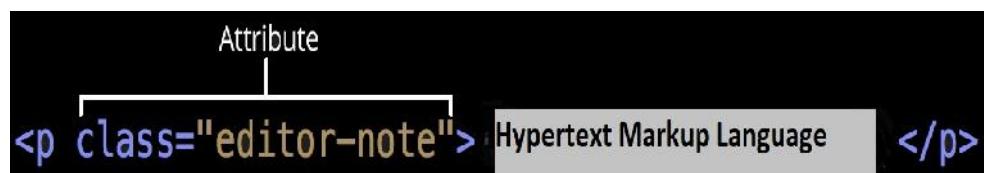
**The opening tag:** This consists of the name of the element (in this case, `p`), wrapped in opening and closing **angle brackets**. This states where the element begins or starts to take effect in this case where the paragraph begins.

**The closing tag:** This is the same as the opening tag, except that it includes a *forward slash* before the element name. This states where the element ends – in this case where the paragraph ends. Failing to add a closing tag is one of the standard beginner errors and can lead to strange results.

**The content:** This is the content of the element, which in this case, is just text.

**The element:** The opening tag, the closing tag, and the content together comprise the element.

Elements can also have attributes that look like the following:



Attributes contain extra information about the element that you don't want to appear in the actual content. Here, `class` is the attribute *name* and `editor-note` are the attribute *value*. The `class` attribute allows you to give the element a non-unique identifier that can be used to target it (and any other elements with the same class value) with style information and other things.

An attribute should always have the following:

1. A space between it and the element name (or the previous attribute, if the element already has one or more attributes).

2. The attribute name is followed by an equals sign.
3. The attribute value is wrapped by opening and closing quotation marks.



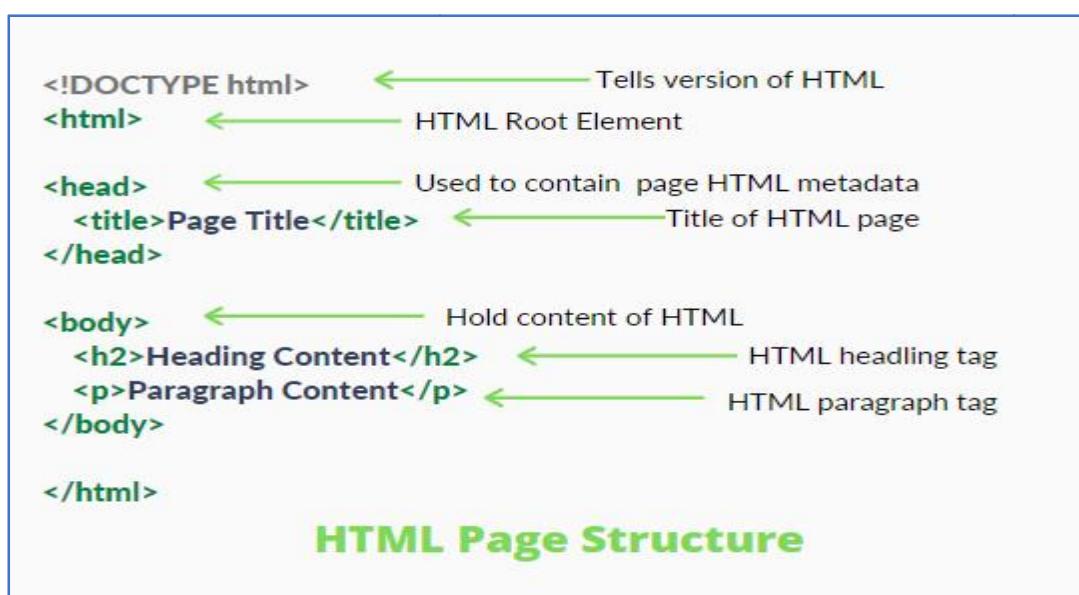
**Note:** Simple attribute values that don't contain ASCII whitespace.

## 4.5 HTML Document Structure

The <HTML> is a markup language that is used by the browser to manipulate text, images, and other content to display it in the required format.

**Tags in HTML:** Tags are one of the most important parts of an HTML Document. HTML uses some predefined tags which tell the browser about content display property, that is how to display a particularly given content.

In its simplest form, the following is an HTML document:



### The DOCTYPE

- The DOCTYPE tells the web browser which version of HTML the page is written in. In this class, we will be using 'XHTML Transitional', which allows us a little flexibility.

### The <html> Element

The <html> element tells the browser that the page will be formatted in HTML and, optionally, which world language the page content is in.

### The <head> and <body> Elements

- The <head> element surrounds all the special "behind the scenes" elements of a web document. Most of these elements do not get displayed directly on the webpage.
- The <body> element surrounds all the actual content (text, images, videos, links, etc.) that will be displayed on our webpage.

### The <meta> Element

- Immediately after the <head> line, we place this <meta> element:
- This line declares that the document is encoded in the UTF-8 (Unicode) character set.

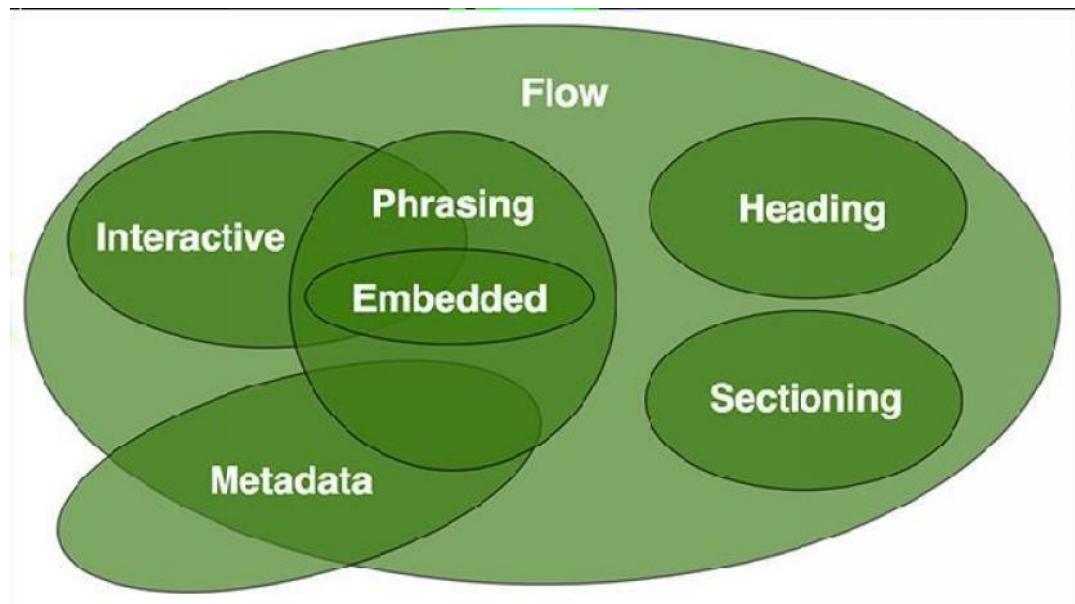
- There can be multiple `<meta>` lines on the same webpage. The `<meta>` element is often used to provide additional information such as page keywords, a page description, and the author(s) of a web document.

*The `<title>` Element*

- The `<title>` element defines what text will show in the web browser's title bar:

## **4.6 HTML Content Model**

The content model refers to the set of rules that define what type of content each element is allowed to have. Mostly, this translates into what other elements are allowed to be nested inside which other elements. Prior to the modern HTML specification, HTML elements were either block-level or inline elements.



**Modern HTML specification splits the two content models into seven models.**

**Metadata:** Metadata content is responsible for setting up the presentation (look) or behavior to the rest of the HTML page. It can also set up the relationship of the HTML document with other documents.

```
<!doctype html>
<html lang="en">
<head>
<meta name="application-name" content="HTML5forbeginners">
</head>
<body>
</body>
</html>
```

**Flow content:** Most contents of HTML documents are in this type. These contents influence other contents to flow. Sectioning content represents a section in the current document. Each sectioning content potentially has a heading content and footer.

**Heading Content:** Heading Content is used to provide different heading levels in HTML documents. These contents are reused to create headlines for a text.



For example, if you want to display an article on your HTML page, the article is written in normal plain text, but its title is written in bold and large text. In this case, you can use the heading elements to make the title distinct from the remaining text. A heading content is defined as - h1, h2, h3, h4, h5, h6, group.

**Phrasing Content:** Phrasing content is the text that you see in the document and in the HTML elements that markup texts at the intra-paragraph level. Runs of phrasing content make up paragraphs.

Phrasing content refers to those small pieces of text that are surrounded by other texts. For example, links. A link is often surrounded by texts. The element that contains phrasing content should contain either text or embedded content. Elements that contain this type of content are inline-level and must have an end tag.

**Embedded content:** Embedded content embeds resources from other sources or adds content from other mark-up languages. For example, image, video, etc.

The HTML elements that can contain embedded content are -

*audio, canvas, embed, iframe, img, math, object, svg, video*

**Interactive Content:** The contents on the webpage that can interact with users are interactive content. For example, links, buttons, etc. Interactive contents are seen inside the form.

## **4.7 HTML Character Entity**

- HTML symbols like mathematical operators, arrows, technical symbols, and shapes, are not present on a normal keyboard.
- Reserved characters in HTML must be replaced with character entities.
- Some characters are reserved in HTML.
- If you use the less than (<) or greater than (>) signs in your text, the browser might mix them with tags.

Character entities are used to display reserved characters in HTML. A character entity looks like this:

**&entity\_name;**

**OR**

**&#entity\_number;**

To display a less than sign (<) we must write: &lt; or &#60

**Advantage of using an entity name:** An entity name is easy to remember.

**A disadvantage of using an entity name:** Browsers may not support all entity names, but the support for entity numbers is good.

## **4.8 HTML LINKS**

Links are found in nearly all web pages. Links allow users to click their way from page to page. HTML links are hyperlinks. You can click on a link and jump to another document. When you move the mouse over a link, the mouse arrow will turn into a little hand.

The HTML <a> tag defines a hyperlink. It has the following syntax:

```
<a href="url">link text</a>
```

The most important attribute of the `<a>` element is the `href` attribute, which indicates the link's destination. The link text is the part that will be visible to the reader. Clicking on the link text, will send the reader to the specified URL address.

This example shows how to create a link to W3Schools.com.

```
<a href="https://www.w3schools.com/">Visit W3Schools.com!</a>
```

By default, the linked page will be displayed in the current browser window. To change this, you must specify another target for the link. The `target` attribute specifies where to open the linked document.

The `target` attribute can have one of the following values:

`_self` - Default. Opens the document in the same window/tab as it was clicked

`_blank` - Opens the document in a new window or tab

`_parent` - Opens the document in the parent frame

`_top` - Opens the document in the full body of the window

## **4.9 Canvas**

The HTML5 canvas element can be used to draw graphics on the webpage via JavaScript. The canvas was originally introduced by Apple for the Mac OS dashboard widgets and to power graphics in the Safari web browser. Later it was adopted by Firefox, Google Chrome, and Opera. Now the canvas is a part of the new HTML5 specification for next-generation web technologies. By default, the `<canvas>` element has 300px of width and 150px of height without any border and content. However, custom width and height can be defined using the CSS `height` and `width` property whereas the border can be applied using the CSS `border` property.

### ***Understanding Canvas Coordinates***

The canvas is a two-dimensional rectangular area. The coordinates of the top-left corner of the canvas are (0, 0) which is known as origin, and the coordinates of the bottom-right corner are (canvas width, canvas height). Here's a simple demonstration of canvas default coordinate system.

### ***HTML5 Canvas***

The HTML5 Canvas is an Immediate Mode bit-mapped area of the screen that can be manipulated with JavaScript and CSS. The *HTML5 Canvas* is an *Immediate Mode* bit-mapped area of the screen that can be manipulated with JavaScript and CSS.

#### ***Immediate Mode***

Immediate Mode refers to the way the canvas renders pixels on the screen. The HTML5 Canvas completely redraws the bitmapped screen on every frame using Canvas API calls from JavaScript. As a programmer, your job is to set-up the screen display before each frame is rendered.

Flash, Silverlight, and DOM `<div>` manipulation techniques use Retained Mode. In Retained Mode a list of individual display objects is stored and manipulated.

### ***HTML5 Canvas Properties***

Canvas Has Three Properties:

- `width`
- `height`
- `id`

Width and height read/write which means you can resize the Canvas on the fly

### **HTML5 Canvas Methods**

`getContext()` : You need the context to draw anything on the Canvas.

`toDataUrl()` : Outputs the bitmapped data of the Canvas to a string (can be used to create a screenshot)

CSS can be used in conjunction with Canvas object itself. However, individual drawings on the Canvas cannot be manipulated with CSS



Example: you can scale the Canvas using CSS

```
style="width: 400px; height:400px"
```

Does not resize but instead scales (like setting width ad height for a Flash embed)

Example:

```
<!DOCTYPE html>
<html>
<body>
<canvas id = "Ani" height = "200" width = "200"
        style = "border:5px solid red">
</canvas>
</body>
</html>
```

Output:



### **4.10 SVG**

SVG stands for Scalable Vector Graphics. SVG is used to define graphics for the Web. SVG is a W3C recommendation

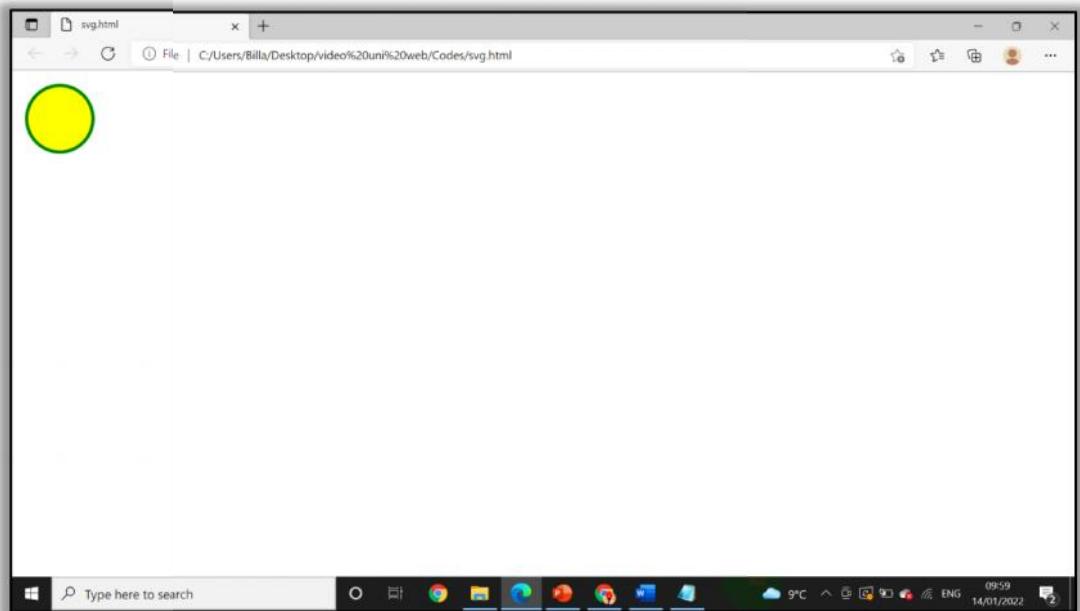
#### ***The HTML <svg> Element***

The HTML `<svg>` element is a container for SVG graphics. SVG has several methods for drawing paths, boxes, circles, text, and graphic images.



Example:

```
<!DOCTYPE html>
<html>
<body>
<svg width="100" height="100">
<circle cx="50" cy="50" r="40" stroke="green" stroke-width="4" fill="yellow" />
</svg>
</body>
</html>
```



## **4.11 HTML Forms**

What is HTML Form?

HTML Forms are required to collect different kinds of user inputs, such as contact details like name, email address, phone numbers, or details like credit card information, etc.

Forms contain special elements called controls like inputbox, checkboxes, radio-buttons, submit buttons, etc. Users generally complete a form by modifying its controls e.g. entering text, selecting items, etc. and submitting this form to a web server for further processing.

The <form> tag is used to create an HTML form. Here's a simple example of a login form:

```
<form>
<label>Username: <input type="text"></label>
<label>Password: <input type="password"></label>
<input type="submit" value="Submit">
</form>
```

### **Input Element**

This is the most commonly used element within HTML forms.

It allows you to specify various types of user input fields, depending on the type attribute. An input element can be of type text field, password field, checkbox, radio button, submit button, reset button, file select box, as well as several new input types introduced in HTML5.

The most frequently used input types are described below.

### **Text Fields**

Text fields are one line areas that allow the user to input text.

Single-line text input controls are created using an `<input>` element, whose type attribute has a value of text. Here's an example of a single-line text input used to take username:

```
<form>  
<label for="username">Username:</label>  
<input type="text" name="username" id="username">  
</form>
```

### **Password Field**

Password fields are similar to text fields. The only difference is; characters in a password field are masked, i.e. they are shown as asterisks or dots. This is to prevent someone else from reading the password on the screen. This is also a single-line text input controls created using an `<input>` element whose type attribute has a value of password.

Here's an example of a single-line password input used to take user password:

```
<form>  
<label for="user-pwd">Password:</label>  
<input type="password" name="user-password" id="user-pwd">  
</form>
```

### **Radio Buttons**

Radio buttons are used to let the user select exactly one option from a pre-defined set of options. It is created using an `<input>` element whose type attribute has a value of radio.

Here's an example of radio buttons that can be used to collect user's gender information:

```
<form>  
<input type="radio" name="gender" id="male">  
<label for="male">Male</label>  
<input type="radio" name="gender" id="female">  
<label for="female">Female</label>  
</form>
```

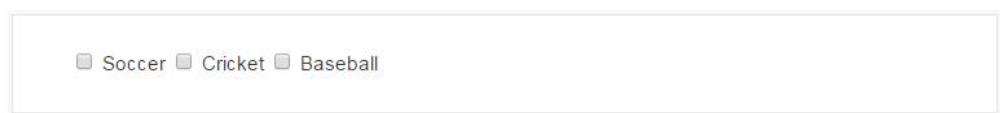
### **Checkboxes**

Checkboxes allows the user to select one or more option from a pre-defined set of options. It is created using an `<input>` element whose type attribute has a value of checkbox.

Here's an example of checkboxes that can be used to collect information about user's hobbies:

```
<form>  
<input type="checkbox" name="sports" id="soccer">  
<label for="soccer">Soccer</label>  
<input type="checkbox" name="sports" id="cricket">  
<label for="cricket">Cricket</label>  
<input type="checkbox" name="sports" id="baseball">
```

```
<label for="baseball">Baseball</label>
</form>
```



Soccer Cricket Baseball

### File Select box

The file fields allow a user to browse for a local file and send it as an attachment with the form data. Web browsers such as Google Chrome and Firefox render a file select input field with a Browse button that enables the user to navigate the local hard drive and select a file.

This is also created using an `<input>` element, whose type attribute value is set to file.

```
<form>
<label for="file-select">Upload:</label>
<input type="file" name="upload" id="file-select">
</form>
```



Upload: Choose File No file chosen

## Textarea

Textarea is a multiple-line text input control that allows a user to enter more than one line of text. Multi-line text input controls are created using an `<textarea>` element.

### Example

#### Try this code »

```
<form>
<label for="address">Address:</label>
<textarea rows="3" cols="30" name="address" id="address"></
textarea>
</form>
```

— The output of the above example will look something like this:



Address:

### Summary

- HTML is a markup language that is used for creating attractive web pages with the help of styling, and which looks in a nice format on a web browser.
- HTML tags are like keywords define that how a web browser will format and display the content.
- If You have used an open tag `<tag>`, then you must use a close tag `</tag>`.

- HTML attributes are special words that provide additional information about the elements or attributes that are the modifier of the HTML element.
- Each element or tag can have attributes, which define the behavior of that element.
- <canvas> gives you an easy and powerful way to draw graphics using JavaScript. It can be used to draw graphs, make photo compositions, or do simple (and not so simple) animations.
- SVG stands for Scalable Vector Graphics and it is a language for describing 2D graphics and graphical applications in XML and the XML is then rendered by an SVG viewer.
- SVG is mostly useful for vector type diagrams like Pie charts, Two-dimensional graphs in an X, Y coordinate system, etc.

### **Keywords**

- **DOCTYPE:** tells the browser which version of HTML the page is written in. In this class, we will be using 'XHTML Transitional', which allows us a little flexibility.
- The <head> element surrounds all the special "behind the scenes" elements of a web document. Most of these elements do not get displayed directly on the webpage.
- The <body> element surrounds all the actual content (text, images, videos, links, etc.) that will be displayed on our webpage.
- **H1** is displayed on the browser in bold format and the size of the text depends on the number of headings.
- **P Tag defines a paragraph**

### **4.12 Self Assessment**

1. Which of the following tag is used to define options in a drop-down selection list?
  - <select>
  - <list>
  - <dropdown>
  - <option>
2. HTML tags are enclosed in-
  - # and #
  - { and }
  - ! and ?
  - < and >
3. Which of the following tag is used to add rows in the table?
  - <td> and </td>
  - <th> and </th>
  - <tr> and </tr>
  - None of the above
4. The <hr> tag in HTML is used for -
  - new line
  - vertical ruler
  - new paragraph
  - horizontal ruler
5. Which of the following attribute is used to provide a unique name to an element?
  - class
  - id
  - type

- D. None of the above
6. Which of the following HTML tag is used to display the text with scrolling effect?
- A. <marquee>
  - B. <scroll>
  - C. <div>
  - D. None of the above
7. Which of the following HTML tag is the special formatting tag?
- A. <p>
  - B. <b>
  - C. <pre>
  - D. None of the above
8. How to insert a background image in HTML?
- A. <body background = "img.png">
  - B. <img background = "img.png">
  - C. <bg-image = "img.png">
  - D. None of the above
9. Which of the following is the correct way to create a list using the lowercase letters?
- A. <ol alpha = "a" >
  - B. <ol type = "a">
  - C. <ol letter = "a">
  - D. None of the above
10. Which of the following HTML attribute is used to define inline styles?
- A. style
  - B. type
  - C. class
  - D. None of the above
11. Which of the following is the paragraph tag in HTML?
- A. <p>
  - B. <b>
  - C. <pre>
  - D. None of the above
12. An HTML program is saved by using the \_\_\_\_ extension.
- A. .ht
  - B. .html
  - C. .hml
  - D. None of the above
13. A program in HTML can be rendered and read by -
- A. Web browser
  - B. Server
  - C. Interpreter
  - D. None of the above
14. What are the types of unordered or bulleted list in HTML?

- A. disc, square, triangle
- B. polygon, triangle, circle
- C. disc, circle, square
- D. All of the above

15. Which of the following is the correct way to create a list using the lowercase letters?

- A. <ol alpha = "a" >
- B. <ol type = "a">
- C. <ol letter = "a">
- D. None of the above

### **Answers for Self Assessment**

- |       |       |       |       |       |
|-------|-------|-------|-------|-------|
| 1. D  | 2. D  | 3. C  | 4. D  | 5. B  |
| 6. A  | 7. C  | 8. A  | 9. B  | 10. A |
| 11. A | 12. B | 13. A | 14. C | 15. B |

### **4.13 Review Questions**

1. What are some of the key new features in HTML5?
2. What are the different new form element types in HTML 5?
3. Explain the layout of HTML?
4. What were some of the key goals and motivations for the HTML5 specification?
5. How to create a hyperlink in HTML
6. What is the canvas element in HTML5?
7. What's the difference between the <svg> and <canvas> elements?
8. Give a simple implementation of the <video> tag to embed a video stored at [http://www.example.com/amazing\\_video.mp4](http://www.example.com/amazing_video.mp4). Give the video width of 640 pixels by 360 pixels.



### **Further Readings**

- HTML5 Black Book (Covers CSS3, JavaScript, XML, XHTML, AJAX, PHP, j Query)  
2Ed.
- Web Enabled Commercial Application Development Using HTML, Java Script, DHTML and PHP (4<sup>th</sup> Revised Edition)
- Trevor Burnham. *Async JavaScript*. The Pragmatic Bookshelf, Raleigh, NC and Dallas, TX, 2012.

## **Unit 5: Advanced HTML5**

### **Objectives**

After studying this unit, you will be able to:

- Discuss the History of HTML5
- How to use HTML Tag
- Describe the structure of HTML
- Explain Content Model
- Discuss SVG

### **5.1 Introduction**

HTML5 is a markup language that was founded in 2004 by the Web Hypertext Application Technology Working Group (WHATWG) whose members include Apple, Mozilla Foundation and Opera Software. Then in Oct 2006 W3C (World Wide Web Consortium) decided to stop their work on XHTML and start collaborating with "WHATWG" to develop HTML5 as a technology. After that, the first version of HTML5 was published in 2008 that was written by Ian Hickson, Google, ian@hixie.ch. But it's not completed and changes are still remaining because according to WHATWG experts "HTML5 is a continually evolving technology that will never end". After the first draft of HTML5, it's time to make compatible browsers that support HTML5 features. And then Mozilla took the first step and introduced Firefox 3 which allows users to view HTML5 in the browser, but Safari, Google Chrome, and IE were far behind in support of HTML5 features in their browsers. HTML5 was gaining popularity day by day. Then in April 2010, Steve Jobs declared that Flash will never be allowed on Apple's smart devices. He said that "Flash was designed for PCs using a mouse, not for touch screens using fingers". This statement is enough to change the mind of many companies and that's the reason they began the development of HTML5.

### **5.2 History of HTML5**

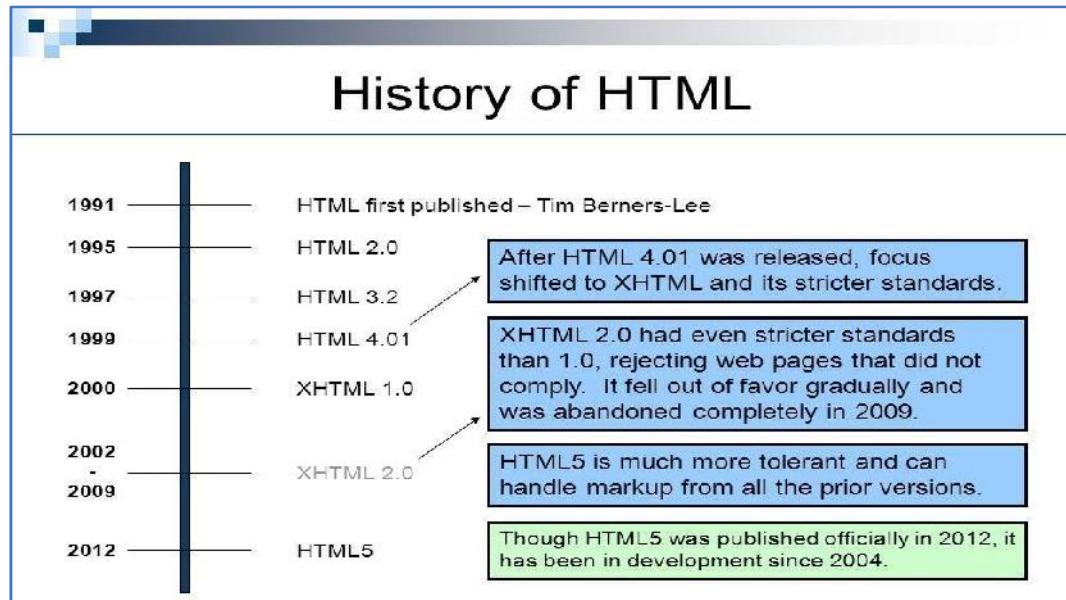
HTML 1.0 was released in 1993 with the intention of sharing information that can be readable and accessible via web browsers. But not many of the developers were involved in creating websites. So, the language was also not growing.

HTML 2.0, published in 1995, contains all the features of HTML 1.0 along with a few additional features, which remained the standard markup language for designing and creating websites until January 1997 and refined various core features of HTML.

HTML 3.0, where Dave Raggett introduced a fresh paper or draft on HTML. It included improved new features of HTML, giving more powerful characteristics for webmasters in designing webpages. But these powerful features of the new HTML slowed down the browser in applying further improvements.

HTML 4.01, which is widely used and was a successful version of HTML before HTML 5.0, which is currently released and used worldwide. HTML 5 can be said for an extended version of HTML 4.01, which was published in the year 2012.

HTML was created by Sir Tim Berners-Lee in late 1991 but was not officially published in 1995 as HTML 2.0. HTML 4.01 was published in late 1999 and was a major version of HTML.



### 5.3 Exploring New Features of HTML 5

- 1) It has introduced new multimedia features which support audio and video controls by using <audio> and <video> tags.
- 2) There are new graphic elements including vector graphics and tags.
- 3) Enrich semantic content by including <header><footer>, <article>, <section> and <figure> are added.
- 4) Drag and Drop - The user can grab an object and drag it further dropping it to a new location.
- 5) Geo-location services - It helps to locate the geographical location of a client.
- 6) Web storage facility which provides web application methods to store data on a web browser.
- 7) Uses SQL database to store data offline.
- 8) Allows drawing various shapes like triangles, rectangles, circles, etc.
- 9) Capable of handling incorrect syntax.
- 10) Easy DOCTYPE declaration i.e., <!doctype html>
- 11) Easy character encoding i.e., <meta charset="UTF-8">

### 5.4 Difference Between HTML and HTML5

Features	HTML	HTML5
Definition	A hypertext markup language (HTML) is the primary language for developing web pages.	HTML5 is a new version of HTML with new functionalities with markup language with Internet technologies.
Multimedia support	Language in HTML does not have support for video and audio.	HTML5 supports both video and audio.
Storage	The HTML browser uses cache memory as temporary storage.	HTML5 has storage options like application cache, SQL database, and web storage.

Browser compatibility	HTML is compatible with almost all browsers because it has been present for a long time, and the browser has made modifications to support all the features.	In HTML5, we have many new tags, elements, and some meta tags that have been removed/modified, so only some browsers are fully compatible with HTML5.
Graphics support	In HTML, vector graphics are possible with tools like Silverlight, Adobe Flash, VML, etc.	In HTML5, vector graphics are supported by default.
Threading	In HTML, the browser interface and JavaScript run in the same thread.	The HTML5 has the JavaScript Web Worker API, which allows the browser interface to run in multiple threads.
Doctype	Doctype declaration in HTML is too long. <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">	The DOCTYPE declaration in HTML5 is very simple "<!DOCTYPE html>"
Character Encoding	Character encoding in HTML is too long. <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">	Character encoding declaration is simple <meta charset="UTF-8">
Multimedia support	Audio and video are not the part of HTML4.	Audio and video are essential parts of HTML5, like: <Audio>, <Video>.



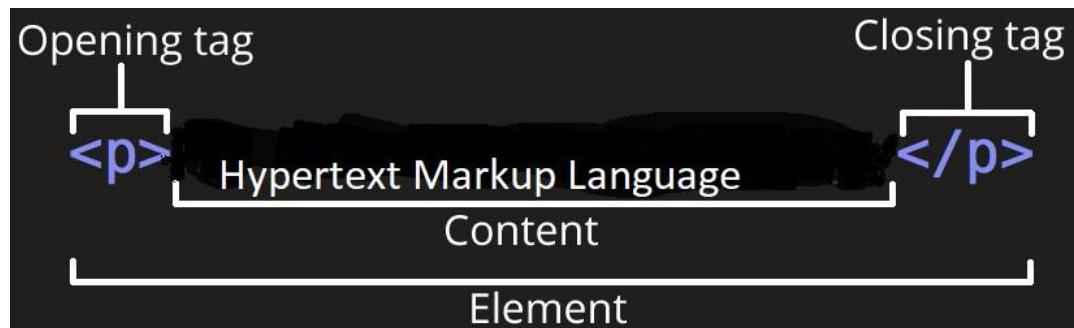
**Did you Know?** : HTML5 provides you with more features with respect to the HTML such as audio, video with the help of tags, drag and drop feature, Geolocation, browser support, etc.

## 5.5 Anatomy of an HTML Tag

HTML consists of a series of elements, which you use to enclose, or wrap, different parts of the content to make it appear a certain way, or act a certain way. The enclosing tags can make a word or image a hyperlink to somewhere else, italicize words, can make the font bigger or smaller, and so on. For example, **Hypertext Markup Language**

If we wanted the line to stand by itself, we could specify that it is a paragraph by enclosing it in paragraph tags:

```
<p>Hypertext Markup Language</p>
```



**The opening tag:** This consists of the name of the element (in this case, p), wrapped in opening and closing **angle brackets**. This states where the element begins or starts to take effect in this case where the paragraph begins.

**The closing tag:** This is the same as the opening tag, except that it includes a *forward slash* before the element name. This states where the element ends – in this case where the paragraph ends. Failing to add a closing tag is one of the standard beginner errors and can lead to strange results.

**The content:** This is the content of the element, which in this case, is just text.

**The element:** The opening tag, the closing tag, and the content together comprise the element.

Elements can also have attributes that look like the following:



Attributes contain extra information about the element that you don't want to appear in the actual content. Here, `class` is the attribute *name* and `editor-note` is the attribute *value*. The `class` attribute allows you to give the element a non-unique identifier that can be used to target it (and any other elements with the same class value) with style information and other things.

An attribute should always have the following:

1. A space between it and the element name (or the previous attribute, if the element already has one or more attributes).
2. The attribute name is followed by an equals sign.
3. The attribute value is wrapped by opening and closing quotation marks.



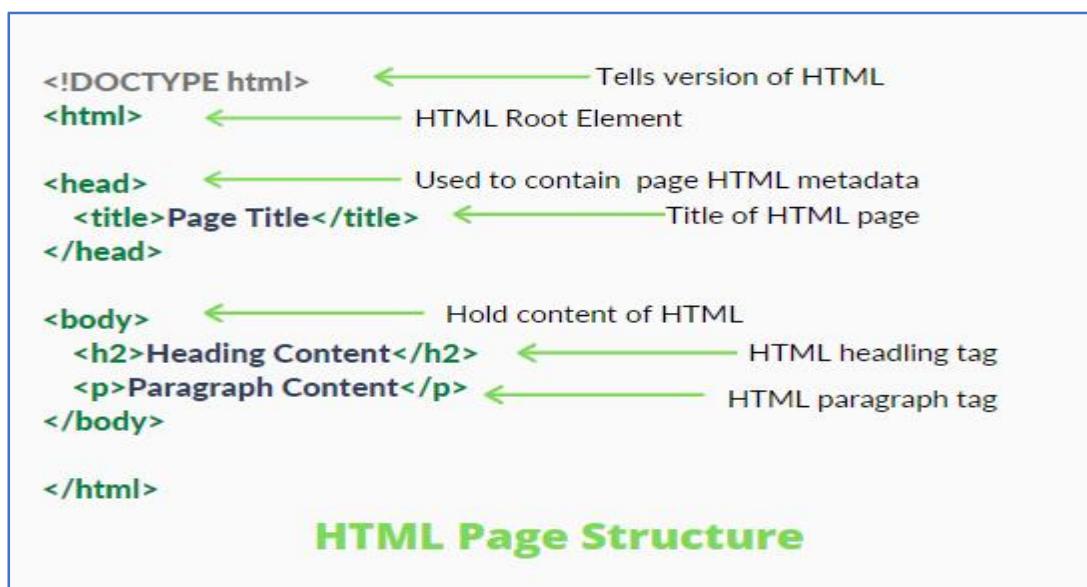
**Note:** Simple attribute values that don't contain ASCII whitespace.

## **5.6 HTML Document Structure**

The `<HTML>` is a markup language that is used by the browser to manipulate text, images, and other content to display it in the required format.

**Tags in HTML:** Tags are one of the most important parts of an HTML Document. HTML uses some predefined tags which tell the browser about content display property, that is how to display a particularly given content.

In its simplest form, the following is an HTML document:



### *The DOCTYPE*

- The DOCTYPE tells the web browser which version of HTML the page is written in. In this class, we will be using 'XHTML Transitional', which allows us a little flexibility.

### *The <html> Element*

The <html> element tells the browser that the page will be formatted in HTML and, optionally, which world language the page content is in.

### *The <head> and <body> Elements*

- The <head> element surrounds all the special "behind the scenes" elements of a web document. Most of these elements do not get displayed directly on the webpage.
- The <body> element surrounds all the actual content (text, images, videos, links, etc.) that will be displayed on our webpage.

### *The <meta> Element*

- Immediately after the <head> line, we place this <meta> element:
- This line declares that the document is encoded in the UTF-8 (Unicode) character set.
- There can be multiple <meta> lines on the same webpage. The <meta> element is often used to provide additional information such as page keywords, a page description, and the author(s) of a web document.

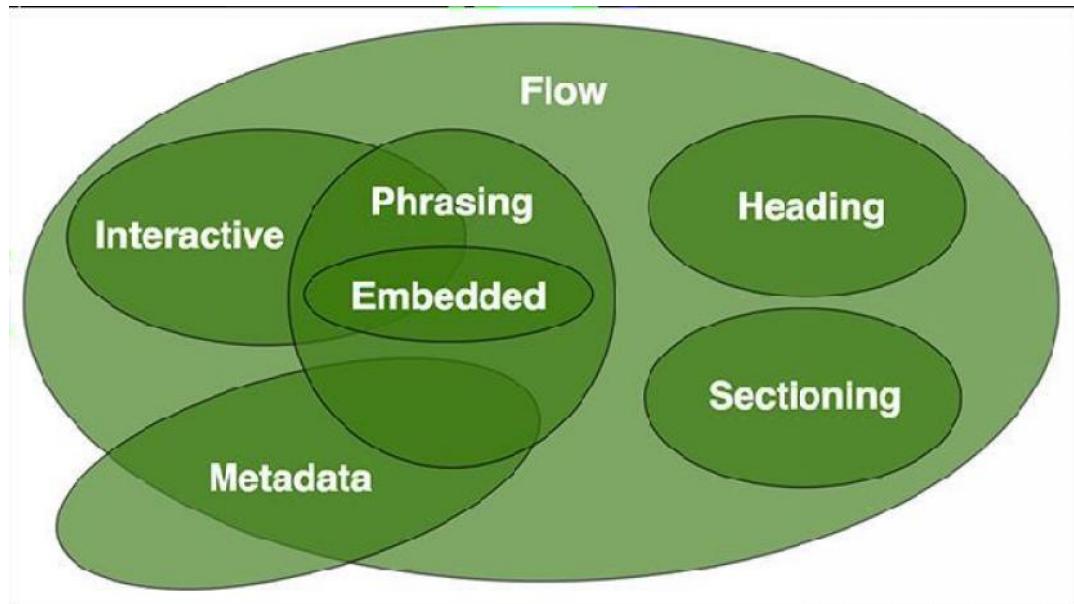
### *The <title> Element*

- The <title> element defines what text will show in the browser's title bar.

## **5.7 HTML Content Model**

The content model refers to the set of rules that define what type of content each element is allowed to have. Mostly, this translates into what other elements are allowed to be nested inside which other elements. Prior to the modern HTML specification, HTML elements were either block-level or inline elements.

**Modern HTML specifications split the set of content models into seven models.**



**Metadata:** Metadata content is responsible for setting up the presentation (look) or behavior to the rest of the HTML page. It can also set up the relationship of the HTML document with other documents.

```
<!doctype html>
<html lang="en">
<head>
<meta name="application-name" content="HTML5forbeginners">
</head>
<body>
</body>
</html>
```

**Flow content:** Most contents of HTML documents are in this type. These contents influence other contents to flow. Sectioning content represents a section in the current document. Each sectioning content potentially has a heading content and footer.

**Heading Content:** Heading Content is used to provide different heading levels in HTML documents. These contents are used to create headlines for a text.



For example, if you want to display an article on your HTML page, the article is written in normal plain text, but its title is written in bold and large text. In this case, you can use the heading elements to make the title distinct from the remaining text. A heading content is defined as - h1, h2, h3, h4, h5, h6, group.

**Phrasing Content:** Phrasing content is the text that you see in the document and in the HTML elements that markup texts at the intra-paragraph level. Runs of phrasing content make up paragraphs. Phrasing content refers to those small pieces of text that are surrounded by other texts. For example, links. A link is often surrounded by texts. The element that contains phrasing content should contain either text or embedded content. Elements that contain this type of content are inline-level and must have an end tag.

**Embedded content:** Embedded content embeds resources from other sources or adds

content from other mark-up languages. For example, images, videos, etc.

The HTML elements that can contain embedded content are-

*audio, canvas, embed, iframe, img, math, object, svg, video*

**Interactive Content:** The contents on the webpage that can interact with users are interactive content. For example, links, buttons, etc. Interactive contents are seen inside the form.

## **5.8 HTML Character Entity**

- HTML symbols like mathematical operators, arrows, technical symbols, and shapes, are not present on a normal keyboard.
- Reserved characters in HTML must be replaced with character entities.
- Some characters are reserved in HTML.
- If you use the less than (<) or greater than (>) signs in your text, the browser might mix them with tags.

Character entities are used to display reserved characters in HTML. A character entity looks like this:

**&entity\_name;**

**OR**

**&#entity\_number;**

To display less than sign (<) we must write: &lt; or &#60

**Advantage of using an entity name:** A named entity is easier to remember.

**A disadvantage of using an entity name:** Browsers may not support all entity names, but the support for entity numbers is good.

## **5.9 HTML LINKS**

Links are found in nearly all web pages. Links allow users to click their way from page to page. HTML links are hyperlinks. You can click on a link and jump to another document. When you move the mouse over a link, the mouse arrow will turn into a little hand.

The HTML  [tag defines a hyperlink. It has the following syntax:](#)

`<a href="url">link text</a>`

The most important attribute of the  [element is the href attribute, which indicates the link's destination. The link text is the part that will be visible to the reader. Clicking on the link text, will send the reader to the specified URL address.](#)

This example shows how to create a link to W3Schools.com.

`<a href="https://www.w3schools.com/">Visit W3Schools.com!</a>`

By default, the linked page will be displayed in the current browser window. To change this, you must specify another target for the link. The target attribute specifies where to open the linked document.

The target attribute can have one of the following values:

\_self - Default. Opens the document in the same window/tab as it was clicked

\_blank - Opens the document in a new window or tab

- \_parent - Opens the document in the parent frame
- \_top - Opens the document in the full body of the window

## **5.10 Canvas**

The HTML5 canvas element can be used to draw graphics on the webpage via JavaScript. The canvas was originally introduced by Apple for the Mac OS dashboard widgets and to power graphics in the Safari web browser. Later it was adopted by Firefox, Google Chrome, and Opera. Now the canvas is a part of the new HTML5 specification for next-generation web technologies. By default, the <canvas> element has 300px of width and 150px of height without any border and content. However, custom width and height can be defined using the CSS height and width property whereas the border can be applied using the CSS border property.

### ***Understanding Canvas Coordinates***

The canvas is a two-dimensional rectangular area. The coordinates of the top-left corner of the canvas are (0, 0) which is known as origin, and the coordinates of the bottom-right corner are (canvas width, canvas height). Here's a simple demonstration of canvas default coordinate system.

### ***HTML5 Canvas***

The HTML5 Canvas is an Immediate Mode bit-mapped area of the screen that can be manipulated with JavaScript and CSS. The *HTML5 Canvas* is an *Immediate Mode* bit-mapped area of the screen that can be manipulated with JavaScript and CSS.

#### ***Immediate Mode***

Immediate Mode refers to the way the canvas renders pixels on the screen. The HTML5 Canvas completely redraws the bitmapped screen on every frame using Canvas API calls from JavaScript. As a programmer, your job is to set-up the screen display before each frame is rendered.

Flash, Silverlight, and DOM <div> manipulation techniques use Retained Mode. In Retained Mode a list of individual display objects is stored and manipulated.

### ***HTML5 Canvas Properties***

Canvas Has Three Properties:

- width
- height
- id

Width and height read/write which means you can resize the Canvas on the fly

### ***HTML5 Canvas Methods***

getContext() : You need the context to draw anything on the Canvas.

toDataUrl() : Outputs the bitmapped data of the Canvas to a string (can be used to create a screenshot)

CSS can be used in conjunction with Canvas object itself. However, individual drawings on the Canvas cannot be manipulated with CSS



Example: you can scale the Canvas using CSS

style="width: 400px; height:400px"

Does not resize but instead scales (like setting width ad height for a Flash embed)

Example:

<!DOCTYPE html>

```
<html>
<body>
<canvas id = "Ani" height = "200" width = "200"
        style = "border:5px solid red">
</canvas>
</body>
</html>
```

Output:



## 5.11 SVG

SVG stands for Scalable Vector Graphics. SVG is used to define graphics for the Web. SVG is a W3C recommendation.

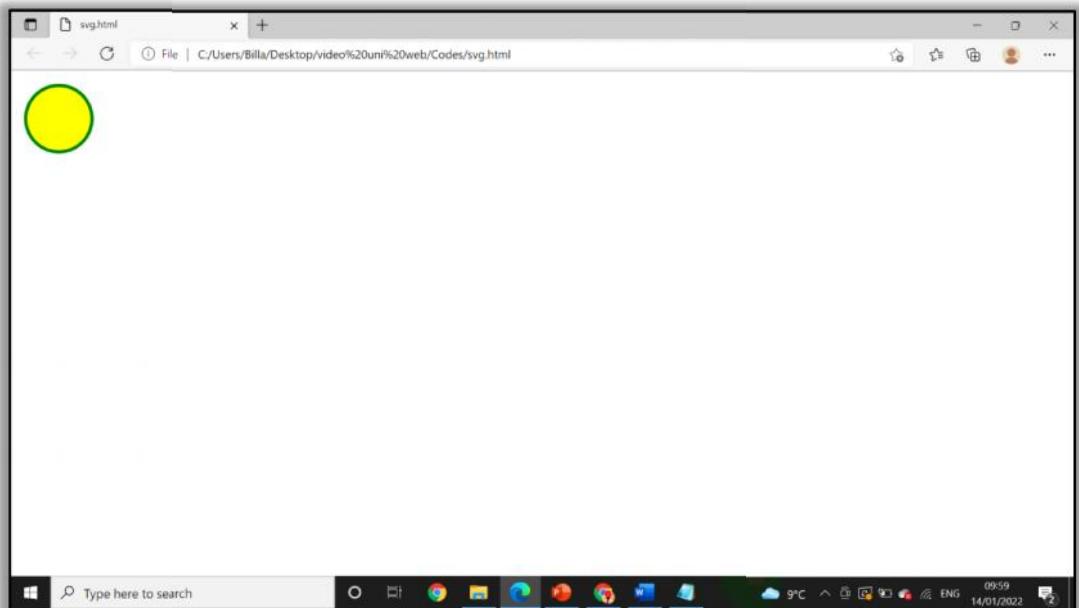
### *The HTML <svg> Element*

The HTML <svg> element is a container for SVG graphics. SVG has several methods for drawing paths, boxes, circles, text, and graphic images.



Example:

```
<!DOCTYPE html>
<html>
<body>
<svg width="100" height="100">
<circle cx="50" cy="50" r="40" stroke="green" stroke-width="4" fill="yellow" />
</svg>
</body>
</html>
```



## **5.12 HTML5 Web Storage**

What is Web Storage?

The HTML5's web storage feature lets you store some information locally on the user's computer, similar to cookies, but it is faster and much better than cookies. However, web storage is no more secure than cookies. Please check out the tutorial on PHP cookies to learn more about cookies.

The information stored in the web storage isn't sent to the web server as opposed to the cookies where data sent to the server with every request. Also, where cookies let you store a small amount of data (nearly 4KB), the web storage allows you to store up to 5MB of data.

**There are two types of web storage, which differ in scope and lifetime:**

Local storage — The local storage uses the `localStorage` object to store data for your entire website on a permanent basis. That means the stored local data will be available on the next day, the next week, or the next year unless you remove it.

Session storage — The session storage uses the `sessionStorage` object to store data on a temporary basis, for a single browser window or tab. The data disappears when session ends i.e. when the user closes that browser window or tab.

### **The `localStorage` Object**

As stated earlier, the `localStorage` object stores the data with no expiration date. Each piece of data is stored in a key/value pair. The key identifies the name of the information (like 'first\_name'), and the value is the value associated with that key (say 'Peter'). Here's an example:

```
<script>
// Check if the localStorage object exists
if(localStorage) {
    // Store data
localStorage.setItem("first_name", "Peter");

    // Retrieve data
alert("Hi, " + localStorage.getItem("first_name"));
```

```

} else {
alert("Sorry, your browser do not support local storage.");
}
</script>

```

**Example explained:**

The above JavaScript code has the following meaning:

- **localStorage.setItem(key, value)** stores the value associated with a key.
- **localStorage.getItem(key)** retrieves the value associated with the key.

You can also remove a particular item from the storage if it exists, by passing the key name to the `removeItem()` method, like `localStorage.removeItem("first_name")`.

However, if you want to remove the complete storage use the `clear()` method, like `localStorage.clear()`. The `clear()` method takes no arguments, and simply clears all key/value pairs from `localStorage` at once, so think carefully before you using it.

**The sessionStorage Object**

The `sessionStorage` object work in the same way as `localStorage`, except that it stores the data only for one session i.e. the data remains until the user closes that window or tab.

Let's try out the following example to understand how it basically works:

```

<script>
// Check if the sessionStorage object exists
if(sessionStorage) {
    // Store data
    sessionStorage.setItem("last_name", "Parker");

    // Retrieve data
    alert("Hi, " + localStorage.getItem("first_name") + " " + sessionStorage.getItem("last_name"));
} else {
    alert("Sorry, your browser do not support session storage.");
}
</script>

```

**5.13 HTML5 Server-Sent Events****What is Server-Sent Event?**

HTML5 server-sent event is a new way for the web pages to communicating with the web server. It is also possible with the `XMLHttpRequest` object that lets your JavaScript code make a request to the web server, but it's a one-for-one exchange – that means once the web server provides its response, the communication is over. `XMLHttpRequest` object is the core of all Ajax operations.

However, there are some situations where web pages require a longer-term connection to the web server. A typical example is stock quotes on finance websites where price updated automatically. Another example is a news ticker running on various media websites.

You can create such things with the HTML5 server-sent events. It allows a web page to hold an open connection to a web server so that the web server can send a new response automatically at any time, there's no need to reconnect, and run the same server script from scratch over and over again.

**Sending Messages with a Server Script**

Let's create a PHP file named "server\_time.php" and type the following script into it. It will simply report the current time of the web server's built-in clock in regular intervals.

```
<?php  
header("Content-Type: text/event-stream");  
header("Cache-Control: no-cache");  
  
// Get the current time on server  
$currentTime = date("h:i:s", time());  
  
// Send it in a message  
echo "data: " . $currentTime . "\n\n";  
flush();  
?>
```

The first two line of the PHP script sets two important headers. First, it sets the MIME type to text/event-stream, which is required by the server-side event standard. The second line tells the web server to turn off caching otherwise the output of your script may be cached.

Every message send through HTML5 server-sent events must start with the text data: followed by the actual message text and the new line character sequence (\n\n).

And finally, we have used the PHP flush() function to make sure that the data is sent right away, rather than buffered until the PHP code is complete.

## **5.14 Processing Messages in a Web Page**

The EventSource object is used to receive server-sent event messages.

Now let's create an HTML document named "demo\_sse.html" and place it in the same project directory where the "server\_time.php" file is located. This HTML document simply receives the current time reported by the web server and display it to the user.

```
<!DOCTYPE html>  
<html lang="en">  
  <head>  
    <title>Using Server-Sent Events</title>  
    <script>  
      window.onload = function() {  
        var source = new EventSource("server_time.php");  
        source.onmessage = function(event) {  
          document.getElementById("result").innerHTML += "New time received from web server: " +  
            event.data + "<br>";  
        };  
      };  
    </script>  
  </head>  
  <body>
```

```
<div id="result">
<!--Server response will be inserted here-->
</div>
</body>
</html>
```

## 5.15 HTML5 Geolocation

### What is Geolocation?

The HTML5 geolocation feature lets you find out the geographic coordinates (latitude and longitude numbers) of the current location of your website's visitor.

This feature is helpful for providing better browsing experience to the site visitor. For example, you can return the search results that are physically close to the user's location.

### Finding a Visitor's Coordinates

Getting the position information of the site visitor using the HTML5 geolocation API is fairly simple. It utilizes the three methods that are packed into the navigator.geolocation object – getCurrentPosition(), watchPosition() and clearWatch().

The following is a simple example of geolocation that displays your current position. But, first you need to agree to let the browser tell the web server about your position.

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>Get Current Position</title>
<script>
    function showPosition() {
        if(navigator.geolocation) {
            navigator.geolocation.getCurrentPosition(function(position) {
                var positionInfo = "Your current position is (" + "Latitude: " + position.coords.latitude +
                ", " + "Longitude: " + position.coords.longitude + ")";
                document.getElementById("result").innerHTML = positionInfo;
            });
        } else {
            alert("Sorry, your browser does not support HTML5 geolocation.");
        }
    }
</script>
</head>
<body>
<div id="result">
<!--Position information will be inserted here-->
```

```
</div>
<button type="button" onclick="showPosition();">Show Position</button>
</body>
</html>
```

## **5.16 HTML5 Drag and Drop**

### **Drag and Drop an Element**

The HTML5 drag and drop feature allows the user to drag and drop an element to another location. The drop location may be a different application. While dragging an element a translucent representation of the element follows the mouse pointer.

Let's try out the following example to understand how it basically works:

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>Using Drag and Drop</title>
<script>
    function dragStart(e) {
        // Sets the operation allowed for a drag source
        e.dataTransfer.effectAllowed = "move";

        // Sets the value and type of the dragged data
        e.dataTransfer.setData("Text", e.target.getAttribute("id"));

    }
    function dragOver(e) {
        // Prevent the browser default handling of the data
        e.preventDefault();
        e.stopPropagation();

    }
    function drop(e) {
        // Cancel this event for everyone else
        e.stopPropagation();
        e.preventDefault();

        // Retrieve the dragged data by type
        var data = e.dataTransfer.getData("Text");

        // Append image to the drop box
        e.target.appendChild(document.getElementById(data));

    }

```

```

</script>
<style>
#dropBox {
    width: 300px;
    height: 300px;
    border: 5px dashed gray;
    background: lightyellow;
    text-align: center;
    margin: 20px 0;
    color: orange;
}
#dropBoximg {
    margin: 25px;
}
</style>
</head>
<body>
<h2>Drag and Drop Demo</h2>
<p>Drag and drop the image into the drop box:</p>
<div id="dropBox" ondragover="dragOver(event);"
     ondrop="drop(event);">
    <!--Dropped image will be inserted here-->
</div>

</body>
</html>

```

### Drag and Drop Events

A number of events are fired during the various stages of the drag and drop operation. But mouse events such as mousemove are not fired during a drag operation.

The following table provides you a brief overview of all the drag and drop events.

#### Event Description

ondragstart	Fires when the user starts dragging an element.
ondragenter	Fires when a draggable element is first moved into a drop listener.
ondragover	Fires when the user drags an element over a drop listener.
ondragleave	Fires when the user drags an element out of drop listener.
ondrag	Fires when the user drags an element anywhere; fires constantly but can give X and Y coordinates of the mouse cursor.
ondrop	Fires when the user drops an element into a drop listener successfully.
ondragend	Fires when the drag action is complete, whether it was successful or not. This event is not fired when dragging a file to the browser from the desktop.

## **5.17 HTML5 Web Storage**

### **What is Web Storage?**

The HTML5's web storage feature lets you store some information locally on the user's computer, similar to cookies, but it is faster and much better than cookies. However, web storage is no more secure than cookies. Please check out the tutorial on PHP cookies to learn more about cookies.

The information stored in the web storage isn't sent to the web server as opposed to the cookies where data sent to the server with every request. Also, where cookies let you store a small amount of data (nearly 4KB), the web storage allows you to store up to 5MB of data.

There are two types of web storage, which differ in scope and lifetime:

**Local storage** – The local storage uses the `localStorage` object to store data for your entire website on a permanent basis. That means the stored local data will be available on the next day, the next week, or the next year unless you remove it.

**Session storage** – The session storage uses the `sessionStorage` object to store data on a temporary basis, for a single browser window or tab. The data disappears when session ends i.e. when the user closes that browser window or tab.

#### **The `localStorage` Object**

As stated earlier, the `localStorage` object stores the data with no expiration date. Each piece of data is stored in a key/value pair. The key identifies the name of the information (like '`first_name`'), and the value is the value associated with that key (say '`Peter`'). Here's an example:

```
<script>
// Check if the localStorage object exists
if(localStorage) {
    // Store data
    localStorage.setItem("first_name", "Peter");

    // Retrieve data
    alert("Hi, " + localStorage.getItem("first_name"));
} else {
    alert("Sorry, your browser do not support local storage.");
}
</script>
```

Example explained:

The above JavaScript code has the following meaning:

**`localStorage.setItem(key, value)` stores the value associated with a key.**

**`localStorage.getItem(key)` retrieves the value associated with the key.**

You can also remove a particular item from the storage if it exists, by passing the key name to the `removeItem()` method, like `localStorage.removeItem("first_name")`.

However, if you want to remove the complete storage use the `clear()` method, like `localStorage.clear()`. The `clear()` method takes no arguments, and simply clears all key/value pairs from `localStorage` at once, so think carefully before you using it.

#### **The `sessionStorage` Object**

The `sessionStorage` object work in the same way as `localStorage`, except that it stores the data only for one session i.e. the data remains until the user closes that window or tab.

Let's try out the following example to understand how it basically works:

```

<script>
// Check if the sessionStorage object exists
if(sessionStorage) {
    // Store data
    sessionStorage.setItem("last_name", "Parker");

    // Retrieve data
    alert("Hi, " + localStorage.getItem("first_name") + " " + sessionStorage.getItem("last_name"));
} else {
    alert("Sorry, your browser do not support session storage.");
}
</script>

```

## **Summary**

- HTML is a markup language that is used for creating attractive web pages with the help of styling, and which looks in a nice format on a web browser.
- HTML tags are like keywords define that how a web browser will format and display the content.
- If You have used an open tag <tag>, then you must use a close tag </tag>.
- HTML attributes are special words that provide additional information about the elements or attributes that are the modifier of the HTML element.
- Each element or tag can have attributes, which define the behavior of that element.
- <canvas> gives you an easy and powerful way to draw graphics using JavaScript. It can be used to draw graphs, make photo compositions, or do simple (and not so simple) animations.
- SVG stands for Scalable Vector Graphics and it is a language for describing 2D graphics and graphical applications in XML and the XML is then rendered by an SVG viewer.
- SVG is mostly useful for vector type diagrams like Pie charts, Two-dimensional graphs in an X, Y coordinate system, etc.

## **Keywords**

- **DOCTYPE:** tellsthe web browser which version of HTML the page is written in. In this class, we will be using 'XHTML Transitional', which allows us a little flexibility.
- The <head> element surrounds all the special "behind the scenes" elements of a web document. Most of these elements do not get displayed directly on the webpage.
- The <body> element surrounds all the actual content (text, images, videos, links, etc.) that will be displayed on our webpage.
- **H1** is displayed on the browser in bold format and the size of the text depends on the number of headings.<sup>1</sup>
- **P Tag defines a paragraph<sup>1</sup>**

## **Self Assessment**

1. Which of the following tag is used to define options in a drop-down selection list?
  - A. <select>
  - B. <list>
  - C. <dropdown>
  - D. <option>

2. HTML tags are enclosed in-
  - A. # and #
  - B. { and }
  - C. ! and ?
  - D. < and >
3. Which of the following tag is used to add rows in the table?
  - A. <td> and </td>
  - B. <th> and </th>
  - C. <tr> and </tr>
  - D. None of the above
4. The <hr> tag in HTML is used for -
  - A. new line
  - B. vertical ruler
  - C. new paragraph
  - D. horizontal ruler
5. Which of the following attribute is used to provide a unique name to an element?
  - A. class
  - B. id
  - C. type
  - D. None of the above
6. Which of the following HTML tag is used to display the text with scrolling effect?
  - A. <marquee>
  - B. <scroll>
  - C. <div>
  - D. None of the above
7. Which of the following HTML tag is the special formatting tag?
  - A. <p>
  - B. <b>
  - C. <pre>
  - D. None of the above
8. How to insert a background image in HTML?
  - A. <body background = "img.png">
  - B. <img background = "img.png">
  - C. <bg-image = "img.png">
  - D. None of the above
9. Which of the following is the correct way to create a list using the lowercase letters?
  - A. <ol alpha = "a" >
  - B. <ol type = "a">
  - C. <ol letter = "a">
  - D. None of the above
10. Which of the following HTML attribute is used to define inline styles?
  - A. style
  - B. type
  - C. class

- D. None of the above
11. Which of the following is the paragraph tag in HTML?
- A. <p>
  - B. <b>
  - C. <pre>
  - D. None of the above
12. An HTML program is saved by using the \_\_\_\_ extension.
- A. ht
  - B. .html
  - C. .hml
  - D. None of the above
13. A program in HTML can be rendered and read by -
- A. Web browser
  - B. Server
  - C. Interpreter
  - D. None of the above
14. What are the types of unordered or bulleted list in HTML?
- A. disc, square, triangle
  - B. polygon, triangle, circle
  - C. disc, circle, square
  - D. All of the above
15. Which of the following is the correct way to create a list using the lowercase letters?
- A. <ol alpha = "a" >
  - B. <ol type = "a">
  - C. <ol letter = "a">
  - D. None of the above

### **Answers for Self Assessment**

- |       |       |       |       |       |
|-------|-------|-------|-------|-------|
| 1. D  | 2. D  | 3. C  | 4. D  | 5. B  |
| 6. A  | 7. C  | 8. A  | 9. B  | 10. A |
| 11. A | 12. B | 13. A | 14. C | 15. B |

### **Review Questions**

1. What are some of the key new features in HTML5?
2. What are the different new form element types in HTML 5?
3. Explain the layout of HTML?
4. What were some of the key goals and motivations for the HTML5 specification?
5. How to create a hyperlink in HTML

6. What is the canvas element in HTML5?
7. What's the difference between the <svg> and <canvas> elements?
8. Give a simple implementation of the <video> tag to embed a video stored at [http://www.example.com/amazing\\_video.mp4](http://www.example.com/amazing_video.mp4). Give the video width of 640 pixels by 360 pixels.



## **Further Readings**

- HTML5 Black Book (Covers CSS3, JavaScript, XML, XHTML, AJAX, PHP, j Query) 2Ed.
- Web Enabled Commercial Application Development Using HTML, Java Script, DHTML and PHP (4<sup>th</sup> Revised Edition)
- Trevor Burnham. *Async JavaScript*. The Pragmatic Bookshelf, Raleigh, NC and Dallas, TX, 2012.

## **UNIT 6: Introduction of CSS, box model and advanced CSS**

### **Objectives**

After studying this unit, you will be able to:

- Introduction to CSS3
- Discuss the selectors present in the syntax of a CSS file
- How to work CSS3
- Learn about how to insert CSS in an HTML document.

### **Introduction**

CSS is used to control the style of a web document in a simple and easy way.

CSS is the acronym for "Cascading Style Sheet".

Cascading Style Sheets fondly referred to as CSS, is a simple design language intended to simplify the process of making web pages presentable.

CSS is a MUST for students and working professionals to become great Software Engineers especially when they are working in Web Development Domain.

### **6.1 Benefits of CSS3**

CSS or cascading sheet is a text-based coding language that specifies the website formats and the way of communicating with web browsers. The language allows web developers to regulate various style elements and functionalities, like layout, color, fonts, and therefore the formatting and display of HTML documents.

The main goal (as a method sheet language) was to separate document content from document presentation, which incorporates style elements, like color, layout, and fonts. CSS handles the design and feel a part of an internet page. Using CSS, you will control the color of the text, the design of fonts, the spacing between paragraphs, how columns are sized and laid out, etc.

- CSS saves time – You can write CSS once and then reuse the same sheet in multiple HTML pages. You can define a style for each HTML element and apply it to as many Web pages as you want.
- Easy maintenance – To make a global change, simply change the style, and all elements in all the web pages will be updated automatically.
- Global web standards – Now HTML attributes are being deprecated and it is being recommended to use CSS. So, it's a good idea to start using CSS in all the HTML pages to make them compatible with future browsers.
- Platform Independence – The Script offers consistent platform independence and can support the latest browsers as well.

### **Why learn CSS?**

**Create a Stunning Website** - CSS handles the look and feel part of a web page. Using CSS, you can control the color of the text, the style of fonts, the spacing between paragraphs, how columns are sized and laid out, what background images or colors are used, layout designs, variations in display for different devices and screen sizes as well as a variety of other effects.

**Become a web designer** - If you want to start a career as a professional web designer, HTML and CSS designing is a must skill.

**Control web** - CSS is easy to learn and understand but it provides powerful control over the presentation of an HTML document. Most commonly, CSS is combined with the markup languages HTML or XHTML.

**Learn other languages** - Once you understand the basics of HTML and CSS then other related technologies like JavaScript, PHP, or angular have become easier to understand.

### **Applications of CSS**

**CSS saves time** - You can write CSS once and then reuse the same sheet in multiple HTML pages. You can define a style for each HTML element and apply it to as many Web pages as you want.

**Pages load faster** - If you are using CSS, you do not need to write HTML tag attributes every time. Just write one CSS rule of a tag and apply it to all the occurrences of that tag. So less code means faster download times.

**Easy maintenance** - To make a global change, simply change the style, and all elements in all the web pages will be updated automatically.

**Superior styles to HTML** - CSS has a much wider array of attributes than HTML, so you can give a far better look to your HTML page in comparison to HTML attributes.

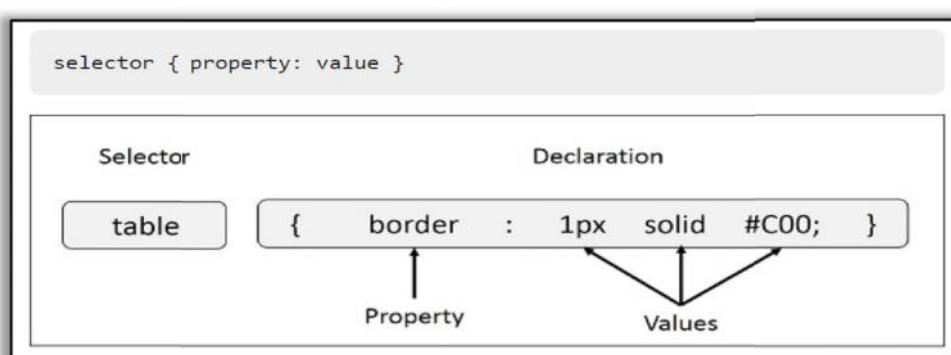
**Multiple Device Compatibility** - Style sheets allow content to be optimized for more than one type of device. By using the same HTML document, different versions of a website can be presented for handheld devices such as PDAs and cell phones or for printing.

**Global web standards** - Now HTML attributes are being deprecated and it is being recommended to use CSS. So, it's a good idea to start using CSS in all the HTML pages to make them compatible with future browsers.

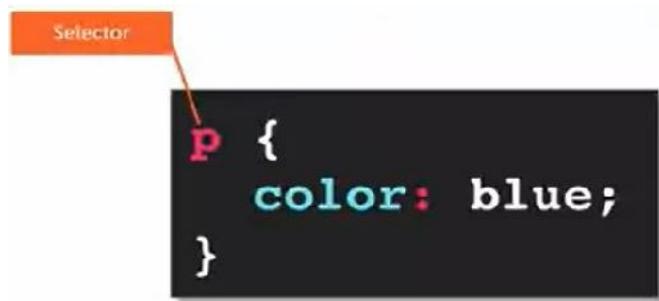
## **6.2 Understanding the syntax of CSS**

The syntax can be defined as a rule that defines the structure or the order of the statements used in a programming language. It also specifies how words and symbols are put together to form statements and expressions. CSS also uses syntax to apply CSS rules in an HTML document.

A CSS comprises style rules that are interpreted by the browser and then applied to the corresponding elements in your document. A style rule is made of three parts- Selector, property, and Value.

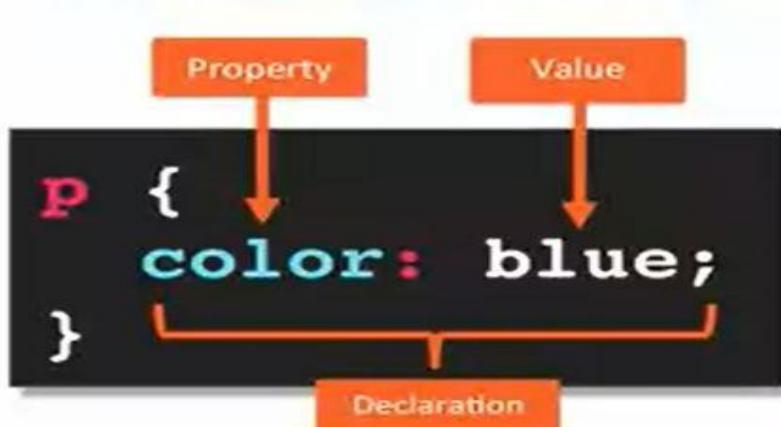


**Selector** - A selector is an HTML tag at which a style will be applied. This could be any tag like `<h1>` or `<table>` etc.



**Property** – A property is a type of attribute of an HTML tag. Put simply, all the HTML attributes are converted into CSS properties. They could be color, border etc.

**Value** – Values are assigned to properties. For example, color property can have value either red or #F1F1F1 etc.



#### Example,

- You can define a table border as follows –
- table{ border :1px solid #C00; }
- Here table is a selector and border is a property and given value 1px solid #C00 is the value of that property.

### 6.3 Define selectors in various simple ways

The following code to understand the selectors.

The Type Selectors

```
h1 {
    color: #36CFFF;
}
```

The Universal Selectors

Rather than selecting elements of a specific type, the universal selector quite simply matches the name of any element type –

```
* {
    color: #000000;
}
```

This rule renders the content of every element in our document in black.

### The Descendant Selectors

Suppose you want to apply a style rule to a particular element only when it lies inside a particular element. As given in the following example, the style rule will apply to the `<em>` element only when it lies inside `<ul>` tag.

```
ul em {  
    color: #000000;  
}
```

### The Class Selectors

You can define style rules based on the class attribute of the elements. All the elements having that class will be formatted according to the defined rule.

```
.black {  
    color: #000000;  
}
```

This rule renders the content in black for every element with a class attribute set to black in our document.



#### Example

You can make it a bit more particular. For example –

```
h1.black {  
    color: #000000;  
}
```

This rule renders the content in black for only `<h1>` elements with class attribute set to black.

You can apply more than one class selector to given element. Consider the following example –

```
<p class = "center bold">
```

This para will be styled by the classes center and bold.

```
</p>
```

## 6.4 Inserting CSS in an HTML Document

There are four ways to associate styles with your HTML document.

The most commonly used methods are

- Inline CSS and
- External CSS.
- Embedded CSS - The `<style>` Element

You can put your CSS rules into an HTML document using the `<style>` element. This tag is placed inside the `<head>...</head>` tags. Rules defined using this syntax will be applied to all the elements available in the document.

```
<!DOCTYPE html>  
<html><head>  
<style type = "text/css" media = "all">  
    body {
```

```

background-color: linen;
}
h1 { color: maroon;
margin-left: 40px; }

</style>
</head>
<body>
<h1>This is a heading</h1>
<p>This is a paragraph.</p>
</body>
</html>

```



### Attributes

Attributes associated with <style> elements are -

Attribute	Value	Description
type	text/css	Specifies the style sheet language as a content-type (MIME type). This is required attribute.
media	Screen tvprojection Handheld print braille aural all	Specifies the device the document will be displayed on. The default value is <i>all</i> .

### Inline CSS - The style Attribute

- You can use the style attribute of any HTML element to define style rules. These rules will be applied to that element only. Here is the generic syntax -
- <element style = "...style rules....">

Attribute	Value	Description
style	style rules	The value of the <i>style</i> attribute is a combination of style declarations separated by semicolons ( ; ).



#### Example

<html>

```
<head>
</head>
<body>
<h1 style = "color:#36C;">
This is inline CSS
</h1>
</body>
</html>
```

**This is inline CSS**

#### External CSS - The <link> Element

- The <link> element can be used to include an external stylesheet file in your HTML document.
- An external style sheet is a separate text file with .CSS extension. You define all the Style rules within this text file and then you can include this file in any HTML document using <link> element.
- Here is the generic syntax of including external CSS file –
- <head>
- <link type = "text/css" href = "..." media = "..." />
- </head>

#### Attributes

Attribute	Value	Description
type	text CSS	Specifies the style sheet language as a content-type (MIME type). This attribute is required.
href	URL	Specifies the style sheet file having Style rules. This attribute is required.
media	Screentvprojection Handheldprint Brailleauralall	Specifies the device the document will be displayed on. The default value is <i>all</i> . This is an optional attribute



#### Example,

Consider a simple style sheet file with a name mystyle.css having the following rules –

```
h1, h2, h3 {
color: #36C;
font-weight: normal;
```

```
letter-spacing: .4em;  
margin-bottom: 1em;  
text-transform: lowercase;  
}
```

Now you can include this file mystyle.css in any HTML document as follows –

```
<head>  
<link type = "text/css" href = "mystyle.css" media = " all" />  
</head>
```

### **Imported CSS - @import Rule**

- @Import is used to import an external stylesheet in a manner similar to the <link> element.  
Here is the generic syntax of @import rule.
- <head>
- @import "URL";
- </head>
- Here URL is the URL of the style sheet file having style rules. You can use another syntax as well –
  - <head>
  - @importurl("URL");

```
</head>
```

### **CSS Colors**

Colors in CSS can be specified by the following methods:

Hexadecimal colors

Hexadecimal colors with transparency

RGB colors

RGBA colors

HSL colors

HSLA colors

Predefined/Cross-browser color names

With the current color keyword

- **Hexadecimal Colors**

A hexadecimal color is specified with: #RRGGBB, where the RR (red), GG (green) and BB (blue) hexadecimal integers specify the components of the color. All values must be between 00 and FF.

 For example, the #0000ff value is rendered as blue, because the blue component is set to its highest value (ff) and the others are set to 00.

 **Example,**

different HEX colors:

```
#p1 {background-color: #ff0000;} /* red */
```

```
#p2 {background-color: #00ff00;} /* green */  
#p3 {background-color: #0000ff;} /* blue */
```

- **RGB Colors**

An RGB color value is specified with the `rgb()` function, which has the following syntax:

`rgb(red, green, blue)`

Each parameter (red, green, and blue) defines the intensity of the color and can be an integer between 0 and 255 or a percentage value (from 0% to 100%).

- For example, the `rgb(0,0,255)` value is rendered as blue, because the blue parameter is set to its highest value (255) and the others are set to 0.
- Also, the following values define equal color: `rgb(0,0,255)` and `rgb(0%,0%,100%)`

- **RGBA Colors**

RGBA color values are an extension of RGB color values with an alpha channel - which specifies the opacity of the object.

An RGBA color is specified with the `rgba()` function, which has the following syntax:

`rgba(red, green, blue, alpha)`

- **HSL Colors**

HSL stands for hue, saturation, and lightness - and represents a cylindrical-coordinate representation of colors.

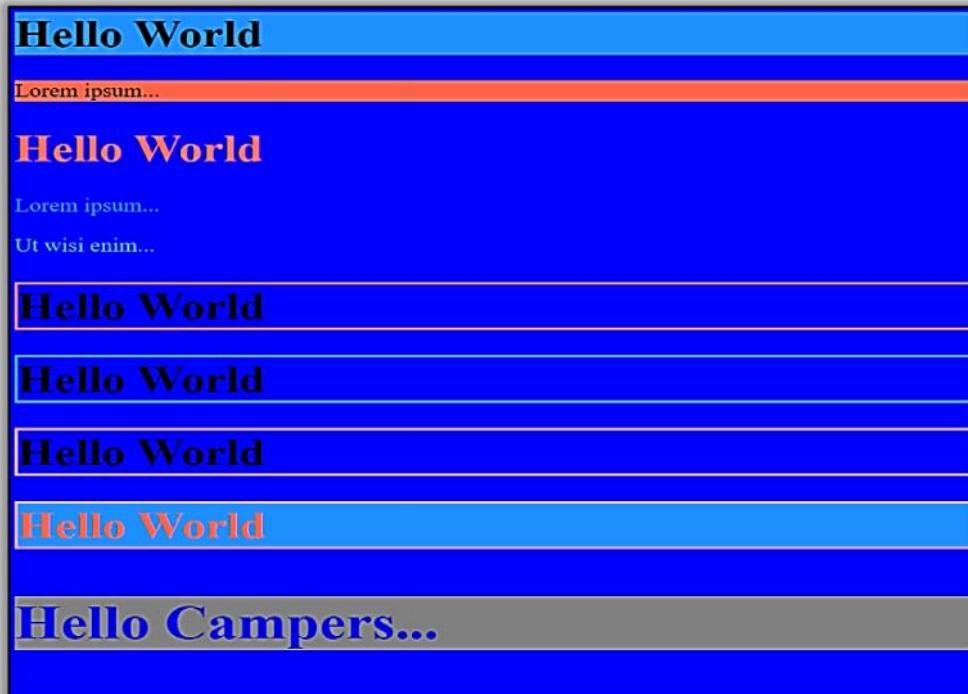
An HSL color value is specified with the `hsl()` function, which has the following syntax:

`hsl(hue, saturation, lightness)`



**Example**

```
<html>  
<body style = "background-color:blue";>  
  
<h1 style="background-color:DodgerBlue;">Hello World</h1>  
<p style="background-color:Tomato;">Lorem ipsum...</p>  
  
<h1 style="color:Tomato;">Hello World</h1>  
<p style="color:DodgerBlue;">Lorem ipsum...</p>  
<p style="color:MediumSeaGreen;">Ut wisi enim...</p>  
  
<h1 style="border:2px solid Tomato;">Hello World</h1>  
<h1 style="border:2px solid DodgerBlue;">Hello World</h1>  
<h1 style="border:2px solid Violet;">Hello World</h1>  
  
<h1 style="background-color:DodgerBlue;color:Tomato; border:2px solid Violet; ">Hello World</h1>  
  
<p style="background-color: grey; color: blue ; font-weight: bold ;font-size: 40px" >  
    Hello Campers...  
</p>  
</html>  
</body>
```



```
<html>
<body bgcolor=yellow>
<h1 style="text-decoration:underline;color:#30F;" align="center">
Example for Inline CSS</h1>
<p style="font-family:Arial, Helvetica, sans-serif; align:left; color:#F00;">
Cascading Style Sheet is a style language </p>
<h2 align="center" style="color:#C0C;text-decoration:underline;">
Image affected with styles</h2>


</body>
</html>
```



## Summary

- **Cascading Style Sheets (CSS)** is a stylesheet language used to describe the presentation of a document written in HTML.
- You declare a CSS class by using a dot (.) followed by the class name. You make the class name yourself. After the class name, and then enter the properties/values that you want to assign to your class.
- It supports more colors and a wide range of color definitions.
- It allows you to change the style of the same HTML file in which you are working.
- Cascading Style Sheets fondly referred to as CSS, is a simple design language intended to simplify the process of making web pages presentable
- Allows Multiple backgrounds on a web page.
- Its display shadow with the text.
- Provides an easy and quick to add a style sheet to a web page. You do not need to create the whole pager or edit a new element in the head of your document.
- Displays the Web page only after the entire style sheet is loaded.

## Keywords

**CSS:** stands for Cascading Style Sheets.

**Selectors:** CSS selectors are used to "find" (or select) the HTML elements.

**Style:** element contains style information for a document or part of a document.

**Href:** Specifies the location of the linked document.

**@Import** is used to import an external stylesheet in a manner similar to the `<link>` element. Here is the generic syntax of the `@import` rule.

## Self Assessment

1. Which of the following property is used as the shorthand property for the padding properties?  
A. padding-left

- B. padding-right  
C. padding  
D. All of the above
2. The CSS property used to make the text bold is -  
A. font-weight : bold  
B. weight: bold  
C. font: bold  
D. style: bold
3. The CSS property used to specify the transparency of an element is -  
A. Opacity1  
B. filter  
C. visibility  
D. overlay
4. What is a CSS selector?  
A. A CSS selector is the CSS class name  
B. A CSS selector is the set of properties that are going to be applied on HTML elements  
C. A CSS selector is name of CSS file.  
D. A CSS selector is the first part of a CSS Rule. It may an HTML element or pattern of elements.
5. A CSS \_\_\_\_\_ are used to selecting HTML elements based on their element name, id, attributes, etc  
A. Selectors  
B. Syntax  
C. Value  
D. None of these
6. A CSS \_\_\_\_\_ consists of a selector, property, and its value.  
A. Syntax rule  
B. Plus rule  
C. Declare rule  
D. None of these
7. CSS stands for -  
A. Cascade style sheets  
B. Color and style sheets  
C. Cascading style sheets  
D. None of the above
8. Which of the following is the correct syntax for referring the external style sheet?  
A. <style src = example.css>  
B. <style src = "example.css" >  
C. <stylesheet> example.css </stylesheet>  
D. <link rel="stylesheet" type="text/css" href="example.css">
9. The property in CSS used to change the background color of an element is -  
A. bgcolor  
B. color  
C. background-color  
D. All of the above
10. The property in CSS used to change the text color of an element is -  
A. bgcolor

B. color

C. background-color

D. All of the above

11. The CSS property used to control the element's font-size is -

A. text-style

B. text-size

C. font-size

D. None of the above

12. The HTML attribute used to define the inline styles is -

A. style

B. styles

C. class

D. None of the above

13. The HTML attribute used to define the internal stylesheet is -

A. <style>

B. style

C. <link>

D. <script>

14. Which of the following CSS property is used to set the background image of an element?

A. background-attachment

B. background-image

C. background-color

D. None of the above

15. Which of the following is the correct syntax to make the background-color of all paragraph elements to yellow?

A. p {background-color : yellow;}

B. p {background-color : #yellow;}

C. all {background-color : yellow;}

D. all p {background-color : #yellow;}

### **Answers for Self Assessment**

1. C            2. A            3. A            4. D            5. A

6. A            7. C            8. D            9. C            10. B

11. C            12. A            13. A            14. B            15. A

### **Review Questions**

1. What are the different ways you could integrate CSS into your HTML page?
2. What is the difference between the usage of an ID and a Class?
3. Elaborate the function of inline CSS with an example?

4. What are the different units used in CSS?
5. Name all the modules which are used in the current version of CSS?
6. What are CSS Selectors?
7. How to align images vertically in a division that spans vertically on the whole webpage?



## **Further Readings**

- HTML5 Black Book (Covers CSS3, JavaScript, XML, XHTML, AJAX, PHP, jQuery)  
2Ed.
- Web Enabled Commercial Application Development Using HTML, JavaScript, DHTML and PHP (4<sup>th</sup> Revised Edition)
- Trevor Burnham. *Async JavaScript*. The Pragmatic Bookshelf, Raleigh, NC and Dallas, TX, 2012.

## Unit 07 - Advanced CSS

### Objectives

After studying this unit, you will be able to:

- Discuss the Styling Text
- Explain The Box Model
- Learn about the Background Property
- Discuss Positioning elements

### Introduction

In this chapter, you learn about Styling Text, The Box Model, Outline Property, and Positioning elements. The box model and various aspects related to it. Such as margin, padding, width, and border. Next, we will learn about different backgrounds, colors, and gradient effects on your web pages. You learn about different background properties, such as background-position and background-repeat, and how to set multiple background values on a single web page. Next will discuss positioning elements, In CSS helps you to place your HTML element. You can put any HTML element at whatever location you like. You can require whether you want the element positioned relative to its natural position on the page or absolute based on its parent element.

### 7.1 Styling Text

The font represents the style and size of the text that is displayed in a web browser. Apart from communicating a visual request to the content, fonts are also used to help users discriminate between different types of information.

The CSS properties used to style text generally fall into two categories.

- **Font styles:** Properties that affect a text's font, e.g., which font gets applied, its size, and whether it's bold, italic, etc.
- **Text layout styles:** Properties that affect the spacing and other layout features of the text, allowing manipulation of, for example, the space between lines and letters, and how the text is aligned within the content box.

CSS provides three values for setting the font styles of text: normal, oblique, and italic. The normal style displays the text in upright and straight letters, oblique displays the text in slanting or leaning letters, and italic displays the text in italic letters.

```
<style>
P{
Font-size:20px;
}
Span. normal {
Font-style: normal;
{
Span. italic{
Font-style:italic;
}
```

```
Span. oblique{  
Font-style: oblique;  
}  
</style>
```

The **text-align** property is used to control how text is aligned within its holding content box. The following possible values are used with text-align property.

- Centre-Changes the text in the middle of the containing element.
- Justify-Fits the text in the containing element.
- Left-Align the text on the left side of the containing element.
- Right-Align the text on the right side of the containing element.

## 7.2 Box Model

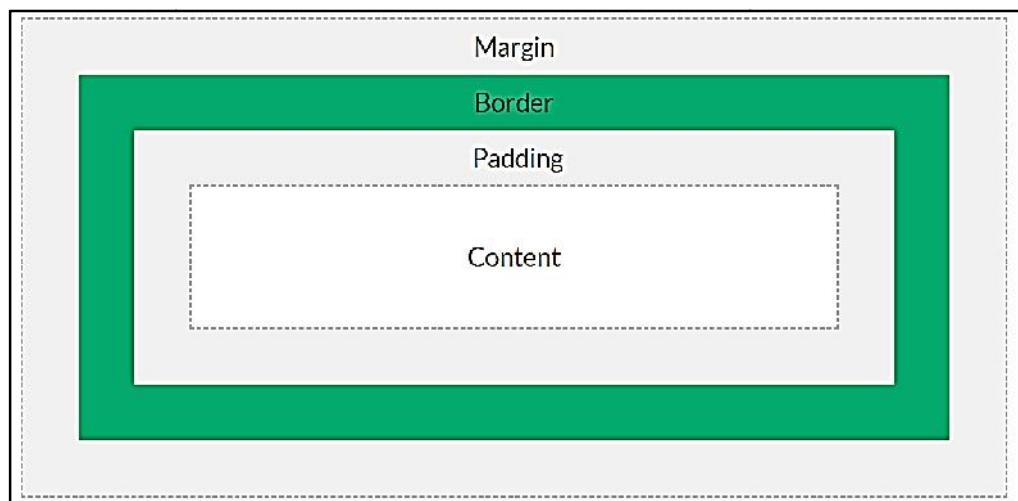
CSS treats an HTML document as an ordered tree of elements, where each element can have one or more child elements arranged in a well-organized way. The topmost element of this tree is called as the root element or the parent element. These elements display their content at a specific position, which is defined by using CSS properties. CSS converts the data of HTML elements into the form of rectangular boxes, by using these outlines is called the box model to set the design of HTML documents. The box model describes how HTML elements are displayed as boxes.

The CSS box model is essentially a box that wraps around every HTML element. It consists of: margins, borders, padding, and the actual content.

The CSS box model is essentially a box that wraps around every HTML element. It consists of:

- **Margins:** In CSS margin area specifies the area around the border area.
- **Borders:** In CSS Border area specifies the area around the padding area
- **Padding:** In CSS padding area specifies the area around the content area.
- **Content:** In CSS content area specifies the displays the content of a document, such as text and images. This is bounded by a rectangle, which is called the content edge.

**Note** Content area always appears inside the padding area.



A CSS box model is a compartment that includes numerous assets, such as edge, border, padding, and material. It is used to develop the design and structure of a web page. It can be used as a set of tools to mark the layout of different components. According to the CSS box model, the web browser supplies each element as a square prism.

**The various other features of the box model are in the following sections:**

- The padding properties
- The border properties
- The margin properties
- The width height properties

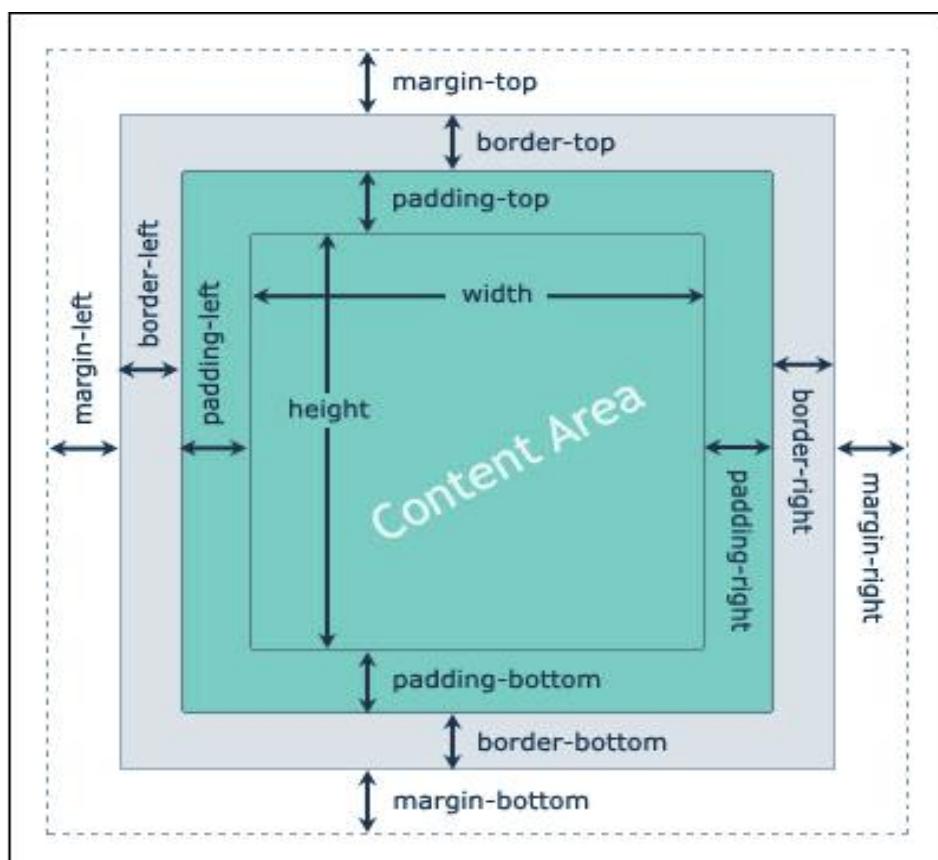
**Padding Properties-** is describing the distance between the border of an element and the content within it. The padding property is affected by the background color of an element; the value of the padding cannot be negative.

The padding property is used to change all the properties such as the padding-top, padding-bottom, padding-right, padding-left.

**Border Properties-** it specifies the space between the padding and converts it into a box model. It defines the width, colour, and style of the border area of the box.

**Margin Properties-** the blank area around the border of an element is called the margin. It is used to create extra space around an element. It is totally transparent and does not contain any background colour. margin is also used to fix the spacing around different elements. The margin property is used to set all the sides of an element, such as top, right, bottom, and left.

**Width-height properties**-specifies the width and height of the content area, padding area, or border area of a box.



#### Example,

```
div {
    width: 300px;
    border: 15px solid green;
    padding: 50px;
    margin: 20px;
}
```

- In order to set the width and height of an element correctly in all browsers, you need to know how the box model works.
- Important: When you set the width and height properties of an element with CSS, you just set the width and height of the content area. To calculate the full size of an element, you must also add padding, borders, and margins.



**Example:**

- This <div> element will have a total width of 350px:

```
div {  
    width: 320px;  
    padding: 10px;  
    border: 5px solid gray;  
    margin: 0;
```

How to calculate:

- 320px (width)
- + 20px (left + right padding)
- + 10px (left + right border)
- + 0px (left + right margin)
- = 350px

**How to calculate the width of an element**

The total width of an element should be calculated like this:

Total element width = width + left padding + right padding + left border + right border + left margin + right margin

The total height of an element should be calculated like this:

Total element height = height + top padding + bottom padding + top border + bottom border + top margin + bottom margin.

**CSS Outline**

- An outline is a line drawn outside the element's border.
- An outline is a line that is drawn around elements, OUTSIDE the borders, to make the element "stand out".

### **7.3 The Background Property**

The background of a web page is the area on which the content of the web page, such as text, tables, borders, and images, is displayed. A web page should have a background that expresses the axiom of the web page. For example, while constructing a web page for an organization, the background can have a logo that represents the organization.

The CSS background property is used to define the background effects on the element. The following CSS background properties affect the HTML elements:

1. background-color
2. background-image
3. background-repeat
4. background-attachment
5. background-position

The **background-color** property is used to set the color of the background area on which an element is displayed. It can be applied to any element. The background color property takes any of the following three values.

- Color name
- Hexadecimal equivalence of the color
- RGB color

 **Example,**

**h1**

```
{ background-color:# FFFFFF;}
```

The **Background-image** property is used to set an image in the background of an element. It is parallel to the background quality of the body element of HTML.

 **Example,**

**Body**

```
{background-image: url(abc.jpg")}
```

The **background-repeat** property allows you to piece the background images along the x-axis and y-axis of an element. The background-repeat property can take either of the following values:

- Repeat-x
- Repeat-y
- Repeat
- No-repeat

 **Example,**

```
{
```

```
<body style="background: url('abc.jpg'); background-repeat:repeat-y,">
```

The **background-attachment** property is used to fix or scroll the background image along with the text and other content displayed on it. The two values take this property fixed or scroll. Fixed means the background image does not move and scroll means the image scroll along with the text written over it.

 **Example,**

```
{
```

```
Body {background-image:url('abc.jpg'); background -attachement:scroll;}
```

```
}
```

 **Example,**

```
{
```

```
Body {background-image:url('abc.jpg'); background -position: right top;}
```

```
}
```

## 7.4 Exploring the Color Property

The RGB format uses three elementary colors, Red, Green, and blue, to specify the color of an element. CSS3 adds a new level, Alpha(A), the level of opacity of this RGB format.

### CSS Opacity / Transparency

The opacity property is used to produce a transparency effect in an HTML element. When you use the opacity property for an HTML element, it is also applied to its child elements.

- The opacity property specifies the opacity/transparency of an element.
- The opacity property can take a value from 0.0 - 1.0. The lower value, the more transparent:



### Transparent Hover Effect

- The opacity property is often used together with the: hover selector to change the opacity on mouse-over.

## 7.5 Positioning Elements

The **position** property in CSS talks about the technique of positioning for an element or an HTML entity.

The CSS position property is used to set the position for an element. It is also used to place an element behind another and is also useful for the scripted animation effect.

### Understand CSS Layout - float and clear property

The CSS float property specifies how an element should float.

The CSS clear property specifies what elements can float beside the cleared element and on which side.

Float Left

Float Right

### Float Properties

- The float property is used for positioning and formatting content e.g. let an image float left to the text in a container.
- The float property can have one of the following values:
  - left - The element floats to the left of its container
  - right - The element floats to the right of its container
  - none - The element does not float (will be displayed just where it occurs in the text). This is default
  - inherit - The element inherits the float value of its parent.



**Note:** In its simplest use, the float property can be used to wrap text around images.



**Example** of float right

```
img
```

```
{
```

```
    float: right;  
}  
}
```



### Example of Float: left:

```
img {  
float: left;  
}  
}
```



Lorem ipsum dolor sit amet, consectetur adipiscing elit. Phasellus imperdiet, nulla et dictum interdum, nisi lorem egestas odio, vitae scelerisque enim ligula venenatis dolor. Maecenas nisl est, ultrices nec congue eget, auctor vitae massa. Fusce luctus vestibulum augue ut aliquet. Mauris ante ligula, facilisis sed ornare eu, lobortis in odio. Praesent convallis urna a lacus interdum ut hendrerit risus congue. Nunc sagittis dictum nisi, sed ullamcorper ipsum dignissim ac...

## Example of No float

- In the following example the image will be displayed just where it occurs in the text (float: none ;):



**LOREM IPSUM DOLOR SIT AMET, CONSECTETUR ADIPISCING ELIT.** Phasellus imperdiet,

nulla et dictum interdum, nisi lorem egestas odio, vitae scelerisque enim ligula venenatis dolor. Maecenas **nisi** est, ultrices nec congue eget, auctor vitae massa. Fusce luctus vestibulum augue ut aliquet. Mauris ante ligula, facilisis sed ornare eu, lobortis in odio. Praesent convallis urna a lacus interdum ut hendrerit risus congue. Nunc sagittis dictum **nisi**, sed ullamcorper ipsum dignissim ac...



### Example - Float Next To Each Other

- Normally div elements will be displayed on top of each other. However, if we use float: left we can let elements float next to each other.

## Float Next To Each Other

In this example, the three divs will float next to each other.



### The Clear Property

- When we use the float property, and we want the next element below (not on right or left), we will have to use the clear property.
- The clear property specifies what should happen with the element that is next to a floating element

**Clear property can have one of the following values:**

- None - The element is not pushed below left or right floated elements. This is default
- left - The element is pushed below left floated elements
- right - The element is pushed below right floated elements
- both - The element is pushed below both left and right floated elements
- inherit - The element inherits the clear value from its parent

When clearing floats, you should match the clear to the float: If an element is floated to the left, then you should clear to the left. Your floated element will continue to float, but the cleared element will appear below it on the web page.



### Example,

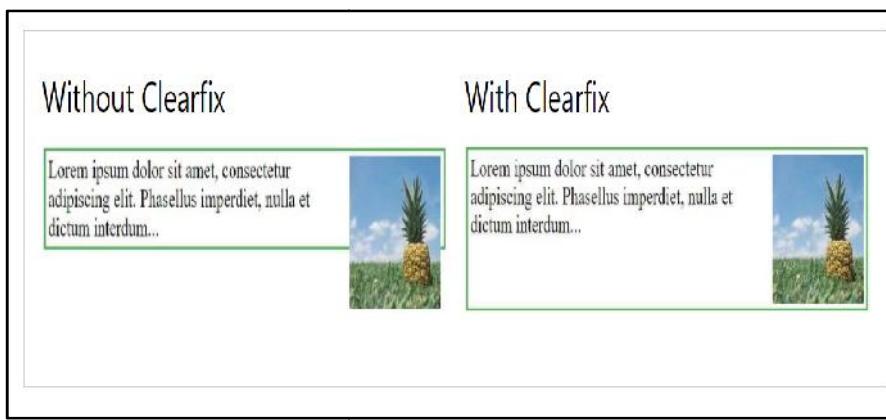
This example clears the float to the left. Here, it means that the <div2> element is pushed below the left floated <div1> element:

```
div1
{
    float: left;
}

div2
{
    clear: left;
}
```

### The Clearfix Hack

- If a floated element is taller than the containing element, it will "overflow" outside of its container. We can then add a clearfix hack to solve this problem:



## Relative Positioning

Relative Positioning changes the position of the HTML element relative to where it normally appears. So "left30" 30 pixels to the element LEFT position.

You can use two values top and left along with the position property to move an HTML element anywhere in the HTML document.

Move left

Move right

Move top

Move bottom

```
<html>
<head>
</head>
<body>

<div style = "position: relative; left 50px; top:3 px; background-color red">
```

This is a CSS3 document.

```
</div>
</body>
</html>
```



**Note:** You can use *bottom* or *right* values as well in the same way as *top* and *left*.

## Absolute Positioning

An element with position: absolute is positioned at the specified coordinates relative to your screen top-left corner.

You can use two values top and left along with the position property to move an HTML element anywhere in the HTML document.

Move left

Move right

Move top

Move bottom

```
<html>
<head>
</head>
<body>

<div style = "position: relative; left 50px; top:3px; background-color red">
```

This is a CSS3 document.

```
</div>
</body>
</html>
```

## **Summary**

- Font's properties help the person who reads to easily understand and identify important terms and information in a document.
- Box model is used when talking about design and layout. The CSS box model is essentially a box that wraps around every HTML element.
- Background properties at the same place in a style sheet, can be used to specify the values for the background color, background image, background-repeat, background-attachment, background position, and background-size properties.
- The opacity property is used to produce a transparency effect in an HTML element.
- Internet browser does not support the opacity property.
- Css3 enables to set of a gradient background for an element. Gradient background is a variation of colors, which are arranged from lightest to darkest or vice versa.
- Background color property can take the value in three ways: as the direct color name, in hexadecimal color format, or in RGB color format.
- Color properties in CSS enables you to add different types of colors to our web page using the different specification. these specifications are RGB,RGBA,HSL,HSLA and opacity.
- Positioning allows you to take elements out of normal document flow and make them behave differently.
- Absolute values are the fixed value that is used to specify the font size on a web page.
- Relative value is how to set the font size of the text using relative values.

## **Keywords**

**Font size** is used to change the size of the text.

**Absolute Value** Refers to the absolute size of the font.

**Font-style** property is used to specify the style of the font.

**Text-align** sets the horizontal alignment of the text.

**Padding** property is set the values for all four directions in a box layout model.

**Border** is specified between the padding and content in the box model.

**URL ()** value allows you to provide the file path to any image, and it will show the background for that element

## **Self Assessment**

1 Which of the following property adds padding to the top of an element?

- A. height
- B. padding-height
- C. top
- D. padding-top

2.A CSS \_\_\_\_\_ is a compartment that includes numerous assets, such as edge, border, padding and material.

- A. box model
- B. table
- C. Bootstrap
- D. None of these

3.Which of the following property defines the style for the bottom border of an element?

- A. border-bottom-style
- B. border-collapse
- C. border-style-bottom
- D. none of the mentioned

4.Which of the following property defines the style for the Top border of an element?

- A. border-top -style
- B. border-collapse
- C. border-style-top
- D. none of the mentioned

5. Which of the following property defines the style for the Right border of an element?

- A. border-right-style
- B. border-left -style
- C. Margin-right
- D. none of the mentioned

6. Which of the following property defines the style for the left border of an element?

- A. border-right-style
- B. border-left -style
- C. Margin-left
- D. none of the mentioned

7. What clears the area around content?

- A. Padding
- B. Border
- C. Margin
- D. None of these

8.A \_\_\_\_\_ is that goes around the padding and content

- A. Padding
- B. Border
- C. Margin
- D. None of these

9.Clears an area outside the border

- A. Padding
- B. Border
- C. Margin
- D. None of these

10.Which of the following visibility property value is described by The element is not visible, but the layout of surrounding elements is not affected?

- A. visible
- B. hidden
- C. collapse
- D. none of the mentioned

11. Which of the following property is used to control the behavior of floating elements?

- A. format
- B. clean
- C. clear
- D. remove

12. The \_\_\_\_\_ property is used for positioning and formatting content

- A. Float
- B. clean
- C. clear
- D. remove

13. The CSS\_\_\_\_\_ property specifies what elements can float beside the cleared element and on which side

- A. format
- B. clean
- C. clear
- D. remove

14. The default value of float is

- A. None
- B. Right
- C. Left
- D. Top

15. Which float property ensures the element floats to the right of its container
- None
  - Right
  - Left
  - Top

### Answers for Self Assessment

1.	D	2.	A	3.	A	4.	A	5.	A
6.	B	7.	A	8.	B	9.	B	10.	C
11.	C	12.	A	13.	C	14.	A	15.	B

### Review Questions

- How do assign absolute values using the font-size property?
- Explain font-size property using percentage values?
- What is the main property of the Box Model in CSS Explain with an example?
- What are the two properties that influence the dimensions of the content in the CSS Box Model?
- Tell us about the property used for image scroll controlling**
- How to use the margin property of a box Explain with an example?
- How to set the top margin property of an element? Negative values can be allowed in this property?
- Display a margin of 10cm from the right and 10 am from the top?
- Tell us about column-span and column fill properties?



### Further Readings

- HTML5 Black Book (Covers CSS3, JavaScript, XML, XHTML, AJAX, PHP, jQuery) 2nd Ed.
- Web Enabled Commercial Application Development Using HTML, JavaScript, DHTML and PHP (4th Revised Edition)
- Trevor Burnham. *Async JavaScript*. The Pragmatic Bookshelf, Raleigh, NC and Dallas, TX, 2012.

## **Unit 08: Images and Media types in Advanced CSS**

### **CONTENTS**

- Objectives
- Introduction
- 8.1     HTML Tables
- 8.2     Creating Tables
- 8.3     Tables and the Border Attribute
- 8.4     Linking Document
- 8.5     Inserting inline images
- 8.6     Creating Image Links
- 8.7     IMG Attributes
- 8.8     HTML Lists
- 8.9     HTML List Classes
- 8.10    HTML List Types
- 8.11    HTML Block and Inline Elements
- 8.12    HTML Layout Elements and Techniques
- 8.13    HTML Forms
- 8.14    I frames in HTML
- Summary
- Keywords
- Self Assessment
- Answers for Self Assessment
- Review Questions
- Further Readings

### **Objectives**

- Describe about HTML Tables
- Explain how HTML Tables are created
- Explain the steps involved in creating HTML Tables
- Describe about Linking Documents
- Describe about Creating Link Lists

### **Introduction**

In this unit, we describe the statements for creating and updating tables. We assume that the user knows about the data which has to be stored and what the structure of the data is, i.e., what tables are to be created and what the appropriate columns are.

### **8.1    HTML Tables**

Tables are defined with the <table> tag. A table is divided into rows (with the <tr> tag), and each

row is divided into data cells (with the <td> tag). The letters td stands for “table data,” which is the content of a data cell. A data cell can contain text, images, lists, paragraphs, forms, horizontal rules, tables, etc.


**Example:**

```
<table border="1">
<tr>
<th>Heading</th>
<th>Another Heading</th>
</tr>
<tr>
<td>row 1, cell 1</td>
<td>row 1, cell 2</td>
</tr>
<tr>
<td>row 2, cell 1</td>
<td>row 2, cell 2</td>
</tr>
</table>
```

**Output**

<u>Heading</u>	<u>Another Heading</u>
row 1, cell 1	row 1, cell 2
row 2, cell 1	row 2, cell 2

## 8.2 Creating Tables

The basic structure of an HTML table consists of the following tags:      Table tags:  
 <TABLE></TABLE>

- Row tags: <TR></TR>
- Cell tags: <TD></TD>

Constructing an HTML table consists of describing the table between the beginning table tag,

<TABLE>, and the ending table tag, </TABLE>. Between these tags, you then construct each row and each cell in the row. To do this, you would first start the row with the beginning row tag,

LOVELY PROFESSIONAL UNIVERSITY 47

Unit 5: Creating Tables

<TR>, and then build the row by creating each cell with the beginning cell tag, <TD>, adding the Notes

data for that cell, and then closing the cell with the ending cell tag, </TD>. When you finish all

of the cells for a row, you would then close the row with the ending row tag, </TR>.



**Notes:** For each new row, you would repeat the process of beginning the row, building each cell in the row, and closing the row.



#### Example:

The following table is an example of a basic table with three rows and two columns of data.

Data 1 Data 2

Data 3 Data 4

Data 5 Data 6

The codes that generated this table will look like this:

```
<TABLE>
<TR>
<TD>Data 1</TD>
<TD>Data 2</TD>
</TR>
<TR>
<TD>Data 3</TD>
<TD>Data 4</TD>
</TR>
<TR>
<TD>Data 5</TD>
<TD>Data 6</TD>
</TR>
</TABLE>
```

This table contains no border, title, or headings. If you wish to add any of these elements to your table, you need to include additional HTML codes.

### **8.3 Tables and the Border Attribute**

If you do not specify a border attribute the table will be displayed without any borders. Sometimes this can be useful, but most of the time, you want the borders to show.

To display a table with borders, you will have to use the border attribute:

```
<table border="1">
<tr>
<td>Row 1, cell 1</td>
<td>Row 1, cell 2</td>
</tr>
</table>
```

## Headings in a Table

Headings in a table are defined with the <th> tag.

```
<table border="1">
<tr>
<th>Heading</th>
<th>Another Heading</th>
</tr>
<tr>
<td>row 1, cell 1</td>
<td>row 1, cell 2</td>
</tr>
<tr>
<td>row 2, cell 1</td>
<td>row 2, cell 2</td>
</tr>
</table>
```

### Output

Heading	Another Heading
row 1, cell 1	row 1, cell 2
row 2, cell 1	row 2, cell 2

## Empty Cells in a Table

Table cells with no content are not displayed very well in most browsers.

```
<table border="1">
<tr>
<td>row 1, cell 1</td>
<td>row 1, cell 2</td>
</tr>
<tr>
<td>row 2, cell 1</td>
<td></td>
</tr>
</table>
```

### Output

row 1, cell 1	row 1, cell 2
row 2, cell 1	



**Note:** that the borders around the empty table cell are missing (NB! Mozilla Firefox displays the border).

To avoid this, add a non-breaking space (&nbsp;) to empty data cells, to make the borders visible:

```
<table border="1">
<tr>
<td>row 1, cell 1</td>
<td>row 1, cell 2</td>
</tr>
<tr>
<td>row 2, cell 1</td>
<td>&nbsp;</td>
</tr>
</table>
```

#### Output

row 1, cell 1	row 1, cell 2
row 2, cell 1	

## Table Tags

Table 1 shows different table tags that can be used in HTML

Tag	Description
<table>	Defines a table
<th>	Defines a table header
<tr>	Defines a table row
<td>	Defines a table cell
<caption>	Defines a table caption
<colgroup>	Defines groups of table columns
<col>	Defines the attribute values for one or more columns in a table
<thead>	Defines a table head
<tbody>	Defines a table body
<tfoot>	Defines a table footer

Table 1 Table tags in HTML



Task: Illustrate the use of various table tags.

## 8.4 Linking Document

Once you have the ability to create HTML pages, you'll want to learn how to create links between them, so that you can start building a site. Links are the essence of HTML – they are what makes it unique.

What makes the web so effective is the ability to define links from one page to another, and to follow links at the click of a button. A single click can take you right across the world!

## Links

Links are defined with the `<a>` tag. Let's create a link to the page defined in the file "example1.html": This link to

```
<a href="Example1.html">My first homepage</a>.
```

The text between the `<a>` and the `</a>` is used as the caption for the link. It is common for the caption to be in blue underlined text. It is by the way a good idea to not have any blank spaces in the names of your HTML files, as this might create problems with some web servers. You can use an underscore, `"_"`, to separate words in your file names. To link to a page on another web site you need to give the full web address (the URL). For instance, to link to "Google" you need to write: A link to `<a href="http://www.google.com">Google</a>`. If you want the user's browser to open a new window for the linked page, (that way the user finds back to your page as soon as he or she closes the new window), use the attribute `target`:

```
A link to <a href="http://www.google.com target='_blank'">Google</a>.
```

## 8.5 Inserting inline images

`<Img>` causes an "inline image" to be inserted into the output. The image will be retrieved and rendered as if it were just another part of the text. Inline images can occur within headings, or paragraphs, almost anywhere in fact, except body (in other words, you can have a 'free floating' `<img>` tag – it must be contained within some other element.)

Like `<hr>`, this is an empty element. That is, there must be no end-tag. You will sometimes see `<hr>` used with an end-tag (e.g. as a container around a caption), but this is obsolete usage.

`<Img>` has 1 required attribute `src` as well as 3 optional attributes.

**src** The `src` attribute is used to specify the URL of the image (i.e. the address or filename the browser uses to retrieve the image file),

e.g. ``

``

**alt** `alt` is used to provide an text alternative to the image for readers whose browsers do not support graphics (or for visually impaired readers using alternative display devices). Although not required, the use of the `alt` attribute is nearly always appropriate and is strongly recommended. The only exception might be cases where the image is strictly decorative or of generic character. In this case the default text chosen by the browser (typically something like "[IMAGE]" may be adequate).

**align** `align` can take one of three values:

- top
- middle
- bottom

and is used to indicate how the browser should align the image with the adjacent text.

**bottom**: align the bottom of the image with the bottom of text

**middle**: align the middle of the image with the middle of text

**top**: align the top of the image with the top of text.

Images can also be retrieved using by means of a hypertext link using the `<a>` tag. The key difference between an inline image and an image retrieved with the `<a>` tag is that an inline image requires no action on the part of the reader; it is retrieved with the page just as if it were part of the text. With the `<a>` tag, the reader has to make a special action (e.g. clicking on a button) to retrieve the image.



**Task:** Give examples of the attributes used with `<img>` tag.

## 8.6 Creating Image Links

Image links are constructed as you might expect, by embedding an `<img>` tag inside of an anchor element `<a>`. Like HTML text links, image links require opening and closing anchor tags, but instead of placing text between these opening and closing tags, the developer needs to place an image tag – with a valid source attribute value of course.



**Example:**

```
<!DOCTYPE html>
<html>
<body>
<p>Create a link of an image: <a href="default.html">
</a></p>
<p>No border around the image, but still a link: <a href="default.html">
</a>
</p>
</body>
</html>
```

### Output

Create a link of an image:

No border around the image, but still a link:

## 8.7 IMG Attributes

`<IMG>` indicates that an image – such as a photograph, icon, animation, cartoon, or other graphic – is to be displayed at that location. The `<IMG>` tag should contain within it further parameters as part of the command: `SRC= "URL/graphic.gif or .jpg"`: contains the URL (Uniform Resource Locator or web address) and name of the graphic image file, such as `graphic.gif` or `graphic.jpg`. Most commonly, the photograph, icon, or other graphic is a “`gif`” (Graphics Interchange Format image) or a “`jpg`” (Joint Photographic Experts Group image), both of which are recognized by most browsers. Some browsers also will recognize a “`bmp`” (Bitmap image) and a “`tif`” or “`tiff`” (Tag Image File Format image). Usually, the location source of the graphic file is in an adjacent directory such as “`graphics`,” or it possibly might be in the same directory. Assuming the image is a `.jpg` image, if the graphic is in an adjacent “`graphics`” directory, the tag would read: `<IMG SRC= "graphics/ graphic.jpg">`. If the image is located within the same directory as the document, the tag would read simply: `<IMG SRC="graphic.jpg">`. If the location of the image is somewhere else on the web, the tag might read something like this: `<IMG SRC= "http://www.tedmontgomery.com/tutorial/graphics/graphic.gif">`.

`ALIGN="LEFT" | "RIGHT" | "TOP" | "TEXTTOP" | "MIDDLE" | "ABSMIDDLE" | "BASELINE" | "BOTTOM" | "ABSBOTTOM"`: places the graphic image at a specified position, in relation either to the page margins or to the text. (Some browsers will not recognize all of these parameters.) “`LEFT`” aligns the image with the left margin of the page and allows text to wrap around the right side of the image. “`RIGHT`” aligns the image with the right margin of the page and allows the text to wrap around the left side of the image. Note: The only way to center a graphic horizontally on a page is to use `<CENTER>&</ CENTER>` tags around the `<IMG SRC>` tag. However, centering a graphic in this manner will prevent text from being

wrapped around either side of it. Also, any ALIGN="RIGHT" or ALIGN="LEFT" parameter within the <IMG> tag will override the effect of the centering tags. "TOP" aligns the top of the image with the top of the tallest item in the line. "TEXTTOP" aligns the top of the image with the top of the tallest text in the line; usually, but not always, the same as the "TOP" parameter. "MIDDLE" aligns the middle of the image with the baseline of the current line. "ABSMIDDLE" aligns the middle of the image with the middle of the current line. "BASELINE" aligns the bottom of the image with the baseline of the current line. "BOTTOM" is the same as the "BASELINE" parameter. "ABSBOTTOM" aligns the bottom of the image with the bottom of the current line; usually, but not always, the same as the "BASELINE" or "BOTTOM" parameter.

WIDTH="W": defines the width "W" of the image in pixels.

HEIGHT="H": defines the height "H" of the image in pixels.

BORDER="B": creates a border around the image, with a uniform width of "B" in pixels. (In case the image is incorporated as a hyperlink, the unvisited, active, and visited colors of the border will be the same as that of the other text hyperlinks on the page.) The default setting is BORDER= "2" (that is, a border 2 pixels wide).

HSPACE="H": creates a space, with width "H" in pixels, between the image and any text immediately to the right and/or left of it. (HSPACE means "horizontal space.")

VSPACE="V": creates a space, with height "V" in pixels, between the image and any text immediately above and/or below it. (VSPACE means "vertical space.")

ALT="alternate description": supplies a description of the image, which will be displayed instead of the image on non-graphical browsers. On typical graphical browsers, this description will appear before the image has loaded and also when the arrow is placed anywhere on the image.

TITLE="title": same function as the ALT tag, which is not recognized by some browsers.

ISMAP: indicates a server-side image map.

USEMAP: indicates a client-side image map

## **8.8 HTML Lists**

Lists commonly are found in documents, including web pages. They are an easy and effective way to itemize such things as elements, components, or ingredients. Words or phrases which need to be set apart from the rest of the body of text can be emphasized with a "bullet" (a heavy dot used for calling attention to a particular section of text). An empty tag called a "list" tag is used to do this: <LI>: creates a bullet in front of text which is to be set apart for emphasis and causes all text after it to be indented, either until another list tag is detected or until the end of the list is reached. It is used to itemize elements of "unordered" and "ordered" lists.



**Notes:** A <BR> tag is not inserted at the end of an item beginning with a <LI> tag, as a line break automatically occurs at that point.

## **8.9 HTML List Classes**

The HTML List classes allow you to easily create lists within your HTML pages. These classes provide methods to get and set various attributes of the lists and the items within the lists. In particular, the parent class HTML List provides a method to produce a compact list that displays items in as small a vertical space as possible.

**Methods for HTML List include:**

Compact the list

Add and remove items from the list

Add and remove lists from the list (making it possible to nest lists)

**Methods for HTML List Item include:**

Get and set the contents of the item

Get and set the direction of the text interpretation

Get and set the language of the input element Use the subclasses of HTML List and HTML List Item to create your HTML lists:

OrderedList and OrderedListItem

UnorderedList and UnorderedListItem

### **OrderedList and OrderedListItem**

Use the `OrderedList` and `OrderedListItem` classes to create ordered lists in your HTML pages.

**Methods for `OrderedList` include:**

Get and set the starting number for the first item in the list

Get and set the type (or style) for the item numbers

Methods for `OrderedListItem` include:

Get and set the number for the item

Get and set the type (or style) for the item number

### **UnorderedList and UnorderedListItem**

Use the `UnorderedList` and `UnorderedListItem` classes to create unordered lists in your HTML pages.

**Methods for `UnorderedList` include:**

Get and set the type (or style) for the items

**Methods for `UnorderedListItem` include:**

Get and set the type (or style) for the item

## **8.10 HTML List Types**

HTML provides three different types of lists to choose from when building a page, including unordered, ordered, and definition lists. Unordered lists are for lists of items where order isn't of important. While ordered lists place strong importance on the order of items. In the case where there is a list of terms and descriptions, perhaps for a glossary, definition lists are available.

### **Unordered List**

Unordered lists are purely a list of related items, in which their order does not matter nor do they have a numbered or alphabetical list element. Creating an unordered list in HTML is accomplished using the unordered list, `ul`, block level element. Each list item within an unordered list is individually marked up using the list item, `li`, block level element. By default most browsers represent each list item with a solid dot.

#### **Example**

```
<ul>
<li>Tamilnadu</li>
<li>Uttar Pradesh</li>
<li>West-Bengal</li>
</ul>
```

### **Ordered List**

The ordered list element, ol, works just like the unordered list element, including how each individual list item is created. The main difference between an ordered list and an unordered list is that with an ordered list the order of which items are represented is important. Instead of showing a dot as the default list item element, an ordered list uses numbers. Using CSS, these numbers can then be changed to letters, Roman numerals, and so on.

### Example

```
<ol type="1">
<li>Tamilnadu</li>
<li>Uttar Pradesh</li>
<li>West-Bengal</li>
</ol>
```

### Description List

Creating a definition list in HTML is accomplished using the dl element. Instead of using the li element to mark up list items, the definition list actually requires two elements: the definition term element, dt, and the definition description element, dd. A definition list may contain numerous terms and descriptions, one after the other. Additionally, a definition list may have multiple terms per description as well as multiple descriptions per term. A single term may have multiple meanings and warrant multiple definitions. In comparison, a single description may be suitable for multiple terms.

```
<dl>
<dt>study</dt>
<dd>the devotion of time and attention to acquiring knowledge on an academic subject, esp. by means of books</dd>
<dt>design</dt>
<dd>a plan or drawing produced to show the look and function or workings of a building, garment, or other object before it is built or made</dd>
<dd>purpose, planning, or intention that exists or is thought to exist behind an action, fact, or material object</dd>
<dt>business</dt>
<dt>work</dt>
<dd>a person's regular occupation, profession, or trade</dd>
</dl>
```

## **8.11 HTML Block and Inline Elements**

Every HTML element has a default display value, depending on what type of element it is.

There are two display values: block and inline.

### Block-level Elements

A block-level element always starts on a new line, and the browsers automatically add some space (a margin) before and after the element.

A block-level element always takes up the full width available (stretches out to the left and right as far as it can).

Two commonly used block elements are: <p> and <div>.

- The <p> element defines a paragraph in an HTML document.
- The <div> element defines a division or a section in an HTML document.

The <p> element is a block-level element.

The <div> element is a block-level element.

### Inline Elements

An inline element does not start on a new line. An inline element only takes up as much width as necessary.

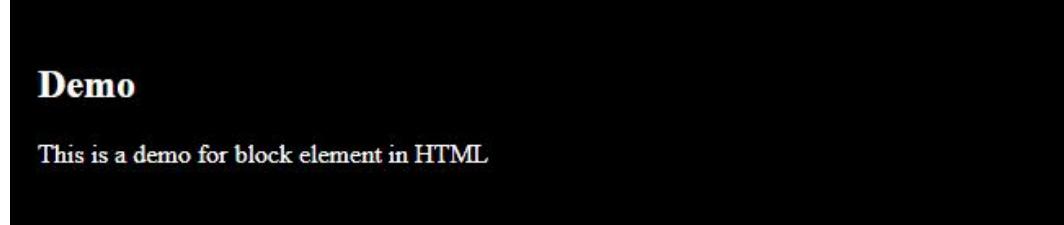
This is a <span> element inside a paragraph

### The <div> Element

The <div> element is often used as a container for other HTML elements. The <div> element has no required attributes, but style, class and id are common.

```
<!DOCTYPE html>
<html>
<body>
<div style="background-color:black;color:white;padding:20px;">
<h2>Demo</h2>
<p>This is a demo for block element in HTML</p>
</div>
</body>
</html>
```

#### Output



Demo

This is a demo for block element in HTML

### The <span> Element

The <span> element is an inline container used to mark up a part of a text, or a part of a document.

The <span> element has no required attributes, but style, class and id are common.

When used together with CSS, the <span> element can be used to style parts of the text:

#### Example

```
<!DOCTYPE html>
<html>
<body>
<h1>The span element</h1>
<p>The sky is <span style="color:blue;font-weight:bold">blue</span> in color and the trees are
<span style="color:darkolivegreen;font-weight:bold">dark green</span> in color. </p>
</body>
```

```
</html>
```

**Output**

## The span element

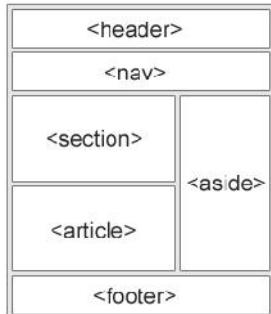
The sky is **blue** in color and the trees are **dark green** in color.

### 8.12 HTML Layout Elements and Techniques

Websites often display content in multiple columns (like a magazine or a newspaper).

#### HTML Layout Elements

HTML has several semantic elements that define the different parts of a web page:



- **<header>** - Defines a header for a document or a section
- **<nav>** - Defines a set of navigation links
- **<section>** - Defines a section in a document
- **<article>** - Defines an independent, self-contained content
- **<aside>** - Defines content aside from the content (like a sidebar)
- **<footer>** - Defines a footer for a document or a section
- **<details>** - Defines additional details that the user can open and close on demand
- **<summary>** - Defines a heading for the **<details>** element

#### Page Layout Information:

**Header:** The part of a front end which is used at the top of the page. **<header>** tag is used to add header section in web pages.

**Navigation bar:** The navigation bar is same as menu list. It is used to display the content information using hyperlink.

**Index / Sidebar:** It holds additional information or advertisements and is not always necessary to be added into the page.

**Content Section:** The content section is the main part where content is displayed.

**Footer:** The footer section contains the contact information and other query related to web pages. The footer section always put on the bottom of the web pages. The **<footer>** tag is used to set the footer in web pages.



#### Example

```
<!DOCTYPE html>
<html lang="en">
<head>
<title>CSS Template</title>
<meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<style>
* {
  box-sizing: border-box;
```

```
}
```

```
body {  
    font-family: Arial, Helvetica, sans-serif;  
}  
  
/* Style the header */  


```
header {  
    background-color: #666;  
    padding: 30px;  
    text-align: center;  
    font-size: 35px;  
    color: white;  
}  
  
/* Create two columns/boxes that floats next to each other */  


```
nav {  
    float: left;  
    width: 30%;  
    height: 300px; /* only for demonstration, should be removed */  
    background: #ccc;  
    padding: 20px;  
}  
  
/* Style the list inside the menu */  


```
nav ul {  
    list-style-type: none;  
    padding: 0;  
}  
  
article {  
    float: left;  
    padding: 20px;  
    width: 70%;  
    background-color: #f1f1f1;  
    height: 300px; /* only for demonstration, should be removed */  
}  
  
/* Clear floats after the columns */  


```
section::after {
```


```


```


```


```

```

content: "";
display: table;
clear: both;
}

/* Style the footer */
footer {
background-color: #777;
padding: 10px;
text-align: center;
color: white;
}

/* Responsive layout - makes the two columns/boxes stack on top of each other instead of next to
each other, on small screens */
@media (max-width: 600px) {
nav, article {
width: 100%;
height: auto;
}
}

</style>
</head>
<body>

<h2>Layout in HTML</h2>


In this example, we have created a header, two columns/boxes and a footer. On smaller screens, the columns will stack on top of each other.</p>
<header>
<h2>HTML</h2>
</header>

<section>
<nav>
<ul>
<li><a href="#">Home</a></li>
<li><a href="#">Gallery</a></li>
<li><a href="#">Contact us</a></li>
</ul>
</nav>


```

```

<article>
<h1>Layout</h1>
<p>Page layout is the part of graphic design that deals with the arrangement of visual elements on a page. Page layout is used to make the web pages look better. It establishes the overall appearance, relative importance, and relationships between the graphic elements to achieve a smooth flow of information and eye movement for maximum effectiveness or impact</p>
</article>
</section>

<footer>
<p>Footer</p>
</footer>

</body>
</html>

```

#### **Layout In HTML**

In this example, we have created a header, two columns/sidemenu and a footer. On smaller screens, the columns will stack on top of each other.



### **8.13 HTML Forms**

HTML stands for HyperText Markup Language. It is used to design web pages using a markup language. It is a combination of Hypertext and Markup language. HTML uses predefined tags and elements that tell the browser how to properly display the content on the screen. And form is one of them. So, in this article, we will learn what is exactly HTML form what are the elements of forms and how can we use HTML form in our webpage.

#### **What is HTML <form>?**

<form> is a HTML element to collect input data with containing interactive controls. It provides facilities to input text, number, values, email, password, and control fields such as checkboxes, radio buttons, submits buttons, etc. or in other words, form is a container that contains input elements, like text, email, number, radio buttons, checkboxes, submit buttons, etc. Forms are generally used when you want to collect data from the user. For example, a user wants to buy a bag online, so he/she has to first enter their shipping address in the address form and then add their payment detail in the payment form to place an order.

Forms are created by placing input fields within paragraphs, preformatted text, lists and tables. This gives considerable flexibility in designing the layout of forms.

#### **Syntax:**

```
<form>
<!--form elements-->
</form>
```

#### Form elements

These are the following HTML <form> elements:

- <label>: It defines label for <form> elements.
- <input>: It is used to get input data from the form in various type such as text, password, email, etc by changing its type.
- <button>: It defines a clickable button to control other elements or execute a functionality.
- <select>: It is used to create a drop-down list.
- <textarea>: It is used to get input long text content.
- <fieldset>: It is used to draw a box around other form elements and group the related data.
- <legend>: It defines caption for fieldset elements.
- <datalist>: It is used to specify pre-defined list options for input controls.
- <output>: It displays the output of performed calculations.
- <option>: It is used to define option in drop-down list.
- <optgroup>: It is used to define group related options in a drop down list.

### Textbox in HTML Form

In an HTML form, we use <input> tag by assigning type attribute value to text to input single line input. To define type attribute see the below syntax.

#### Syntax:

```
<input type="text" />
```

Or shorthand for "text" type:

```
<input />
```

### Password in an HTML Form

We can change type value text to password to get the input password



#### Example

```
<!DOCTYPE html>
<html>
<head>
<title>Page Title</title>
</head>
<body>
<h2>Welcome To HTML Forms</h2>
<form>
<p>
<label>Username : <input type="text" /></label>
</p>
<p>
<label>Password : <input type="password" /></label>
</p>
<p>
<button type="submit">Submit</button>

```

```

</p>
</form>
</body>
</html>
```

## Welcome To HTML Forms

Username :

Password :

### Radio Button in an HTML Form

To create a radio button, we use the `<input>` tag following by radio type to provide users to choose a limited number of choices.

Syntax:

```
<input type="radio" name="radio_button_name" value="radio_button_value" />
```

**Note:** The radio button must have shared the same name to be treated as a group.

**Note:** The value attribute defines the unique value associated with each radio button. The value is not shown to the user, but is the value that is sent to the server on “submit” to identify which radio button that was selected.



Example:

```

<!DOCTYPE html>
<html>
<head>
<title>Page Title</title>
</head>
<body>
<h2>Select your gender</h2>
<form>
    <label>Male<input type="radio" name="gender" value="male" /></label>
    <label>Female<input type="radio" name="gender" value="female" /></label>
</form>
</body>
</html>
```

### Select your gender

Male  Female

In this example, we will create a radio button to choose your gender.

### Checkbox in an HTML Form

To create a checkbox in an HTML form, we use the <input> tag following by the input type checkbox. It is a square box to ticked to activate this. It used to choose more options at a time.

#### Syntax:

```
<input type="checkbox" name="select_box_name" value="select_box_value" />
```

Note: the "name" and "value" attributes are used to send the checkbox data to the server.



#### Example:

In this example, we use checkboxes to select language.

```
<!DOCTYPE html>
<html>
<head>
<title>Page Title</title>
</head>
<body>
<h2>Choose Language</h2>
<form>
<ul style="list-style-type:none;">
<li><input type="checkbox" name="language" value="hindi" />Hindi</li>
<li><input type="checkbox" name="language" value="english" />English</li>
<li><input type="checkbox" name="language" value="sanskrit" />Sanskrit</li>
</ul>
</form>
</body>
</html>
```

### Choose Language

- Hindi
- English
- Sanskrit

## Combobox in an HTML Form

Combobox is used to create a drop-down menu in your form which contains multiple options. So, to create an Combobox in an HTML form, we use the `<select>` tag with `<option>` tag. It is also known as a drop-down menu.

Syntax:

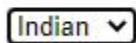
```
<select name="select_box_name">
<option value="value1">option1</option>
<option value="value2">option2</option>
<option value="value3">option3</option>
</select>
```



**Example:**

```
<!DOCTYPE html>
<html>
<head>
<title>Page Title</title>
</head>
<body>
<h2>Select Your Nationality</h2>
<form>
<select name="language">
    <option value="indian">Indian</option>
    <option value="nepali">Nepali</option>
    <option value="others">Others</option>
</select>
</form>
</body>
</html>
```

## Select Your Nationality



## Submit button in an HTML Form

In the HTML form, submit button is used to submit the details of the form to the form handler. A form handler is a file on the server with a script that is used to process input data.

Syntax:

```
<button type="submit">submit</button>
<!DOCTYPE html>
<html>
<head>
<title>Page Title</title>
```

```

</head>
<body>
<h2>Welcome To HTML forms</h2>
<form>
<p>
    <label>Username: <input type="text" /></label>
<p>
    <label>Password: <input type="password" /></label>
</p>
<p>
    <button type="submit">submit</button>
</p>
</form>
</body>
</html>

```

## Welcome To HTML forms

Username:

Password:

### TextArea in an HTML Form

In the HTML form, a text area is used to add comments or reviews, or addresses in the form. Or in other words, the text area is a multi-line text input control. It contains an unlimited number of characters, and the text renders in a fixed-width font and the size of the text area is given by the `<rows>` and `<cols>` attributes. To create a text area in the form use the `<textarea>` tag.

Syntax:

```

<textarea name="textarea_name">content</textarea>
<!DOCTYPE html>
<html>
<head>
<title>Page Title</title>
</head>
<body>
<h2>Welcome To HTML Forms</h2>
<form>
    <textarea    name="welcomeMessage"    rows="3"    cols="40">Kindly    give    your
feedback</textarea>
</form>

```

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

## Welcome To HTML Forms

Kindly give your feedback

### 8.14 Iframes in HTML

HTML Iframe is used to display a nested webpage (a webpage within a webpage). The HTML <iframe> tag defines an inline frame, hence it is also called as an Inline frame.

An HTML iframe embeds another document within the current HTML document in the rectangular region.

The webpage content and iframe contents can interact with each other using JavaScript.

#### Iframe Syntax

An HTML iframe is defined with the <iframe> tag:

```
<iframesrc="URL"></iframe>
```

Here, "src" attribute specifies the web address (URL) of the inline frame page.

#### Set Width and Height of iframe

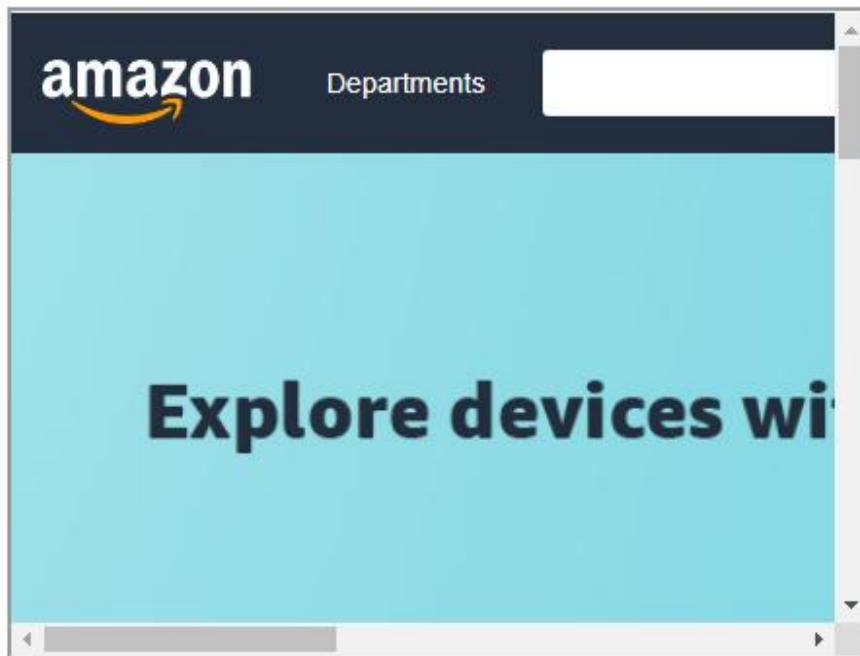
You can set the width and height of iframe by using "width" and "height" attributes. By default, the attributes values are specified in pixels but you can also set them in percent. i.e. 50%, 60% etc.

#### Example: (Pixels)

```
<!DOCTYPE html>  
<html>  
<body>  
<h2>HTML Iframes example</h2>  
<p>Use the height and width attributes to specify the size of the iframe:</p>  
<iframesrc="https://www.amazon.com/" height="300" width="400"></iframe>  
</body>  
</html>
```

## HTML Iframes example

Use the height and width attributes to specify the size of the iframe:



You can also change the size, color, style of the iframe's border.

### Embed YouTube video using iframe

You can also add a YouTube video on your webpage using the <iframe> tag. The attached video will be played at your webpage and you can also set height, width, autoplay, and many more properties for the video.

Following are some steps to add YouTube video on your webpage:

- Goto YouTube video which you want to embed.
- Click on SHARE ➔ under the video.
- Click on Embed <> option.
- Copy HTML code.
- Paste the code in your HTML file
- Change height, width, and other properties (as per requirement).



### Example

```
<!DOCTYPE html>
<html>
<head>

</head>
<body style="background-color: #f0f8ff">
    <h3>Play videos using iframe</h3>
```

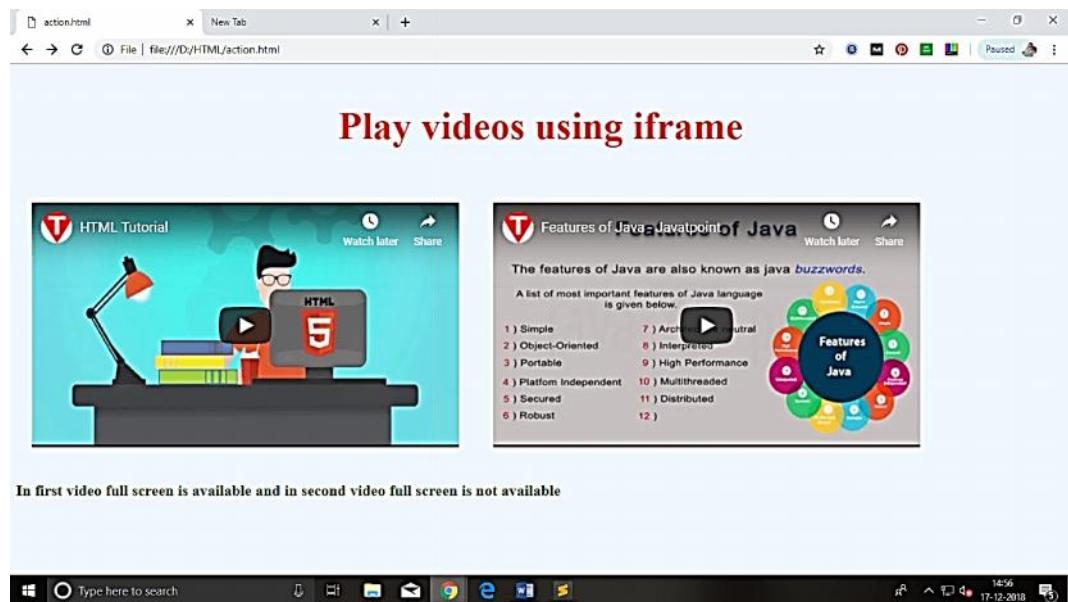
```
<iframe width="550" height="315"
src="https://www.youtube.com/embed/JHq3pL4cdy4" frameborder="0" allow="accelerometer;
autoplay; encrypted-media; gyroscope; picture-in-picture" allowfullscreen
style="padding:20px;"></iframe>
```

```
<iframe width="550" height="315" src="https://www.youtube.com/embed/O5hShUO6wxs"
frameborder="0" allow="accelerometer; autoplay; encrypted-media; gyroscope; picture-in-picture"
style="padding:20px;"></iframe>
```

<p>In first video full screen is available and in second video full screen is not available</p>

</body>

</html>



## Summary

- Lists commonly are found in documents, including web pages.
- The HTML List classes allow you to easily create lists within your HTML pages.
- HTML List provides a method to produce a compact list that displays items in as small a vertical space as possible.
- HTML provides three different types of lists to choose from when building a page, including unordered, ordered, and definition lists.
- Ordered lists place strong importance on the order of items.
- Unordered lists are purely a list of related items, in which their order does not matter nor do they have a numbered or alphabetical list element.
- A definition list may contain numerous terms and descriptions, one after the other.
- One of the best ways to create an impact with your web page is to add some graphics.
- Tables are defined with the <table> tag.
- Constructing an HTML table consists of describing the table between the beginning table tag, <TABLE>, and the ending table tag, </TABLE>.
- Links are the essence of HTML — they are what makes it unique. Links are defined with the <a> tag.

- If you do not specify a border attribute the table will be displayed without any borders.
- <Img> causes an “inline image” to be inserted into the output.
- Image maps are images with clickable areas (sometimes referred to as “hotspots”) that usually link to another page.
- Email links are done much the same as links to other pages, using the <a href> tag.
- The key difference between an inline image and an image retrieved with the <a> tag is that an inline image requires no action on the part of the reader

## **Keywords**

- <IMG>: <IMG> indicates that an image—such as a photograph, icon, animation, cartoon, or other graphic—is to be displayed at that location.
- DL: Definition lists, created using the DL element, generally consist of a series of term/definition Pairs. HTML List classes: It allows you to easily create lists within your
- HTML pages. Inline image: An image which is displayed on a web browser is referred to as an “inline image”.
- alt: alt is used to provide an text alternative to the image for readers whose browsers do not support graphics.
- Cell tags: <TD></TD> Image links: Image links are constructed as you might expect, by embedding an <img> tag inside of an anchor element <a>.
- Links: Links are the essence of HTML — they are what makes it unique
- Row tags: <TR></TR>
- List tag: An empty tag called a “list” tag is used to do itemize elements of “unordered” and “ordered” lists.
- OL: An ordered list, created using the OL element, should contain information where order should be emphasized.
- Ordered List Item: It allows you to override the numbering and type for a specific item in the list. Unordered lists: Unordered lists are for lists of items where order isn’t of important.

## **Self Assessment**

1. Which HTML tag is used to define a table?
  - A. <tb>
  - B. <tl>
  - C. <table>
  - D. <tab>
  
2. With the help of which tag, is a row defined in HTML?
  - A. <row>
  - B. <table-row>
  - C. <tablerow>
  - D. <tr>

3. Choose the correct option.

- A. <th> is used for defining the heading of a table.
- B. By default, contents written between <th> and </th> are bold and centered.
- C. Both A and B
- D. None Of these

4. Fill in the blanks with the help of options given below in order to get the following table when the below code is executed.

```
<!DOCTYPE html>
<html>
<body>
<table border="2">
<tr>
<th>Name</th>
<th>Phone no</th>
</tr>
<tr>
<td _____>John</td>
<td>9898989898</td>
</tr>
<tr>
<td>9876543210</td>
</tr>
</body>
</html>
```

Name	Phone no
John	9898989898
	9876543210

- A. colspan= "1"
- B. colspan= "2"
- C. rowspan= "1"
- D. rowspan= "2"

5. Which one of the following tags is used to add caption to a table?

- A. <table-caption>
- B. <tcaption>
- C. <caption>
- D. <tc>

6. Each cell of the table can be represented by using \_\_\_\_\_

- A. <tr>
- B. <td>
- C. <th>
- D. <thead>

7. The \_\_\_\_\_ tag defines an image in an HTML page.

- A <image>
- B <pic>

C <imge>

D <img>

8. Which of the following pair of attribute is required for img tag ?

- A.src and a
- B. img and src
- C. img and alt
- D. src and alt

9. If the image cannot be displayed then \_\_\_\_\_ specifies an alternate text for an image.

- A. text attribute
- B. value attribute
- C. alt attribute
- D. caption attribute

10. Alt Attribute is more useful in the situation where user have \_\_\_\_\_.

- A. High Speed Internet Connection
- B. Broadband Connection
- C. None of these
- D. Slow Internet Connection

11. Which one of the following is a type of lists that HTML supports?

- A. Ordered lists.
- B. Unordered lists.
- C. Description lists.
- D. All of the above

12. By which tag, an unordered list is represented?

- A. <u>
- B. <I>
- C. <ul>
- D. <ol>

13. By which tag, an ordered list is represented?

- A. <u>
- B. <I>
- C. <ul>
- D. <ol>

14. A HTML code is given below

Fill in the blanks with appropriate option to get the output as below:

```
<!DOCTYPE html>
<html>
<body>
<dl>
    _____
Mathematics
    _____
Calculus
</dl>
</body>
</html>
```

- A <dd>,</dd>,<dt>,</dt>
- B. <dt>,</dt>,<dd>,</dd>
- C. <li>,</li>,<dd>,</dd>
- D. <dt>,</dt>,<li>,</li>

15. What is the default item marker in unordered lists of HTML?

- A. Circle
- B. Marker
- C. disc
- D. None of the above

### **Answers for Self Assessment**

- |       |       |       |       |       |
|-------|-------|-------|-------|-------|
| 1. C  | 2. D  | 3. C  | 4. D  | 5. C  |
| 6. B  | 7. C  | 8. B  | 9. C  | 10. D |
| 11. D | 12. C | 13. D | 14. B | 15. C |

### **Review Questions**

1. On what information we should emphasize while preparing an order list? ‘
2. Discuss the methods for HTML List.
3. Discuss the methods to create ordered lists, unordered lists, and nested lists.
4. Discuss IMG Attributes.
5. Explain with examples about graphic image alignment parameters in the HTML.
6. Explain the three different types of HTML lists.
7. Explain how HTML Tables are created?
8. Define and Explain the steps involved in creating HTML Tables.
9. Discuss in brief about Linking Document.
10. What are the three attributes that can be specified with the <BODY> tag? Explain each of them.

11. Define and explain Hyperlinks and their types.
12. . Justify the use of link list creation in html document.
13. Explain how to display a table with borders.
14. . Discuss how to link to a page on another web site. Illustrate with example.
15. Explain the steps used to move to a specific location on a Web page



### **Further Readings**

Hall, 2009, Core Web Programming, 2/E, Pearson Education India. Jon Duckett, 2011, Beginning Web Programming with HTML, XHTML and CSS,

John Wiley & Sons. Robert F. Breedlove, 1996, Web Programming Unleashed, Sams.net.

Tim Downey, 2012, Guide to Web Development with Java: Understanding Website Creation, Springer.



### **Web Links**

<http://www.temple.edu/cs/web/tables.html>

[http://www.w3schools.com/html/tryit.asp?filename=tryhtml\\_imglink](http://www.w3schools.com/html/tryit.asp?filename=tryhtml_imglink)

<http://www.tizag.com/htmlT/htmlimagelinks.php>

[http://www.quackit.com/html/tutorial/html\\_image\\_maps.cfm](http://www.quackit.com/html/tutorial/html_image_maps.cfm)

<http://www.echoecho.com/htmllinks11.htm>

<http://interestingwebs.blogspot.in/2008/12/how-to-create-scrollable-linklist.html>

<http://www-sul.stanford.edu/tools/tutorials/html2.0/img.html>

## **UNIT 9: Introduction of JavaScript, basic elements and JavaScript objects**

### **Objectives**

After studying this unit, you will be able to:

- Introduction to JavaScript
- Adjusting Development Environment for JavaScript Development
- Where to place JavaScript Code

### **Introduction**

A lightweight programming language ("scripting language") used to make web pages interactive insert dynamic text into HTML (ex: user name), react to events (ex: page load user click), get information about a user's computer (ex: browser type), perform calculations on user's computer (ex: form validation).

JavaScript was first known as LiveScript, but Netscape changed its name to JavaScript, possibly because of the excitement being generated by Java. JavaScript made its first appearance in Netscape 2.0 in 1995 with the name LiveScript. The general-purpose core of the language has been embedded in Netscape, Internet Explorer, and other web browsers.

### **Applications of Javascript Programming**

- **Manipulating HTML Pages** - Javascript helps in manipulating HTML page on the fly.
- Adding and deleting any HTML tag very easily using javascript and modify your HTML to change its look and feel based on different devices and requirements.
- **User Notifications** - You can use Javascript to raise dynamic pop-ups on the webpages to give different types of notifications to your website visitors
- **Back-end Data Loading** - Javascript provides Ajax library which helps in loading back-end data while you are doing some other processing. This really gives an amazing experience to your website visitors.
- **Presentations** - JavaScript also provides the facility of creating presentations which gives website look and feel. JavaScript provides RevealJS and BespokeJS libraries to build a web-based slide presentations.
- Client Side validation - This is really important to verify any user input before submitting it to the server and Javascript plays an important role in validating those inputs at front-end itself.



**Example:** To "make web pages alive," JavaScript was initially developed. Scripts are what the programmes in this language are known as. They can be directly written in the HTML of a web page and executed when the page loads. Plain text scripts are delivered and run. They can run without any additional setup or compilation.

## **Why to Learn JavaScript**

Javascript is everywhere, it comes installed on every modern web browser and so to learn Javascript you really do not need any special environment setup. For example Chrome, Mozilla Firefox , Safari and every browser you know as of today, supports Javascript. Javascript helps you create really beautiful and crazy fast websites. You can develop your website with a console like look and feel and give your users the best Graphical User Experience.

Great thing about Javascript is that you will find tons of frameworks and Libraries already developed which can be used directly in your software development to reduce your time to market.

## **Applications of JavaScript Programming**

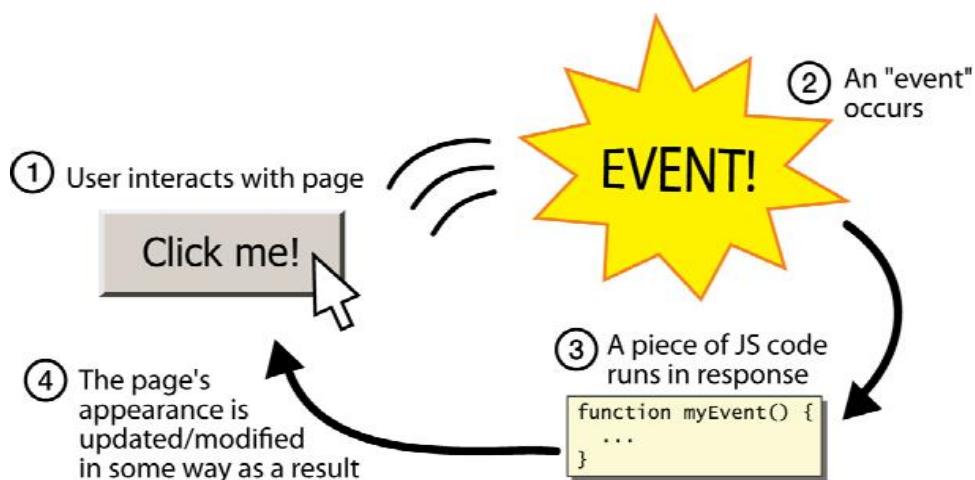
- Manipulating HTML Pages - Javascript helps in manipulating HTML page on the fly.
- This helps in adding and deleting any HTML tag very easily using javascript and modify your HTML to change its look and feel based on different devices and requirements.
- User Notifications - You can use Javascript to raise dynamic pop-ups on the webpages to give different types of notifications to your website visitors.
- Back-end Data Loading - Javascript provides Ajax library which helps in loading back-end data while you are doing some other processing. This really gives an amazing experience to your website visitors.
- Presentations - JavaScript also provides the facility of creating presentations which gives website look and feel. JavaScript provides RevealJS and BespokeJS libraries to build a web-based slide presentation.
- Client side validation - This is really important to verify any user input before submitting it to the server and Javascript plays an important role in validating those inputs at front-end itself.

## **Linking JavaScript File**

JavaScript file can be linked by script tag that must be placed in HTML page's head. Code is stored in a separate .js file.JS code can be placed directly in the HTML file's body or head (like CSS), but this is bad style (should separate content, presentation, and behavior).

```
<script src="filename" type="text/javascript"></script>
```

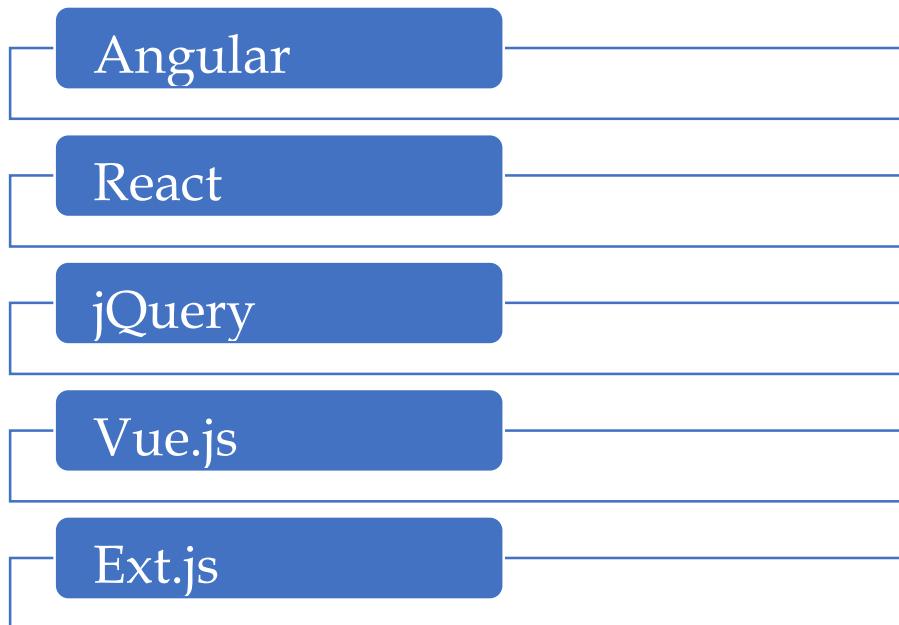
## **Event-driven Programming**



Split breaks apart a string into an array using a delimiter. It can also be used with regular expressions (seen later). Join merges an array into a single string, placing a delimiter between them

### **JavaScript Frameworks and Libraries**

There are many useful Javascript frameworks and libraries available:



JavaScript can be implemented using JavaScript statements that are placed within the <script>...</script> HTML tags in a web page. You can place the <script> tags, containing your JavaScript, anywhere within your web page, but it is normally recommended that you should keep it within the <head> tags. The <script> tag alerts the browser program to start interpreting all the text between these tags as a script

### **JavaScript syntax**

A simple syntax of your JavaScript will appear as follows.

<script ...>

    JavaScript code

</script>

Script tag

The script tag takes two important attributes –

Language – This attribute specifies what scripting language you are using. Typically, its value will be javaScript. Although recent versions of HTML (and XHTML, its successor) have phased out the use of this attribute.

Type – This attribute is what is now recommended to indicate the scripting language in use and its value should be set to "text/javaScript".

Script Tag

<script language = "javaScript" type = "text/javaScript">

    JavaScript code

</script>

### **First JavaScript Code**

Let us take a sample example to print out "Hello World"

```
. <html>
<body>
<script language = "javascript" type = "text/javascript">
document.write("Hello World!")
</script>
</body>
</html>
```

### **Advantages of JavaScript**

- Speed. Client-side JavaScript is very fast because it can be run immediately within the client-side browser. Unless outside resources are required, JavaScript is unhindered by network calls to a backend server.
- Simplicity. JavaScript is relatively simple to learn and implement.
- Popularity. JavaScript is used everywhere on the web.
- Interoperability. JavaScript plays nicely with other languages and can be used in a huge variety of applications.
- Server Load. Being client-side reduces the demand on the website server.
- Gives the ability to create rich interfaces.

### ***JavaScript offers procedural programming features***

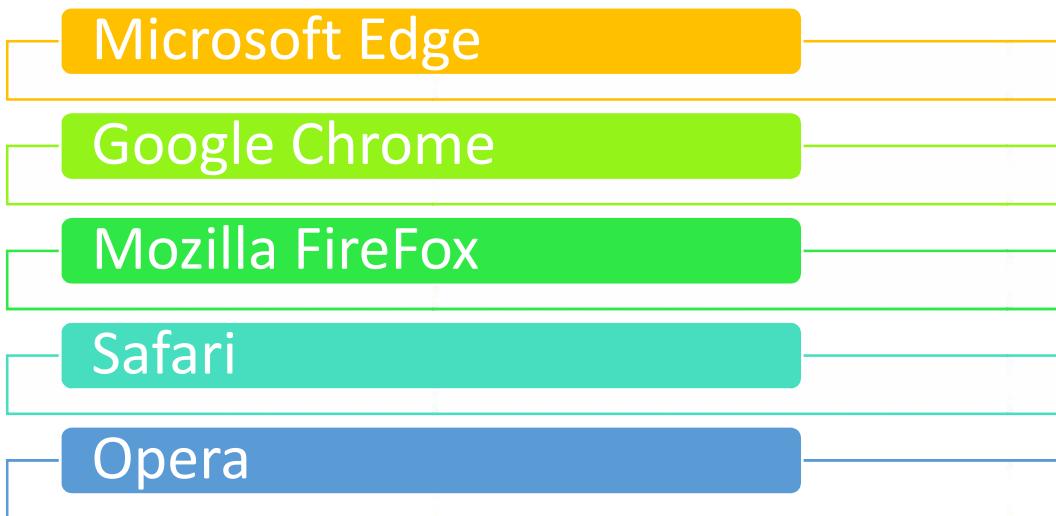
Even though the language is simple to discover, it offers all the procedure-based features that make it a well-liked and influential programming language. With JavaScript, if you have options to generate branches, initiate conditional checking, loops and much more, which will make your website much more exciting to use.

### ***Setup JavaScript Development Environment***

To use JavaScript in developing a web application you must have at least two things, a browser, and an editor to write the JavaScript code.

Browser

You can also install the following browser as per your preference:



#### ***IDEs for JavaScript Application Development***

You can write JavaScript code using a simple editor like Notepad. However, you can install any open-sourced or licensed IDE (Integrated Development Environment) to get the advantage of IntelliSense support for JavaScript and syntax error/warning highlighter for rapid development.

JavaScript Editors:

The followings are some of the well-known JavaScript editors:

- Visual Studio Code (Free, cross-platform)
- Eclipse (Free, cross-platform)
- Atom (Free, cross-platform)
- Notepad++ (Free, Windows)
- Code Lobster (Free, cross-platform)
- WebStorm (Paid, cross-platform)

#### ***Where can I download JavaScript?***

- You don't have to get or download JavaScript.
- JavaScript is already running in your browser on your computer, on your tablet, and on your smart-phone.
- JavaScript is free to use for everyone.

#### **JavaScript importance**

JavaScript is one of the 3 languages all web developers must learn:

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

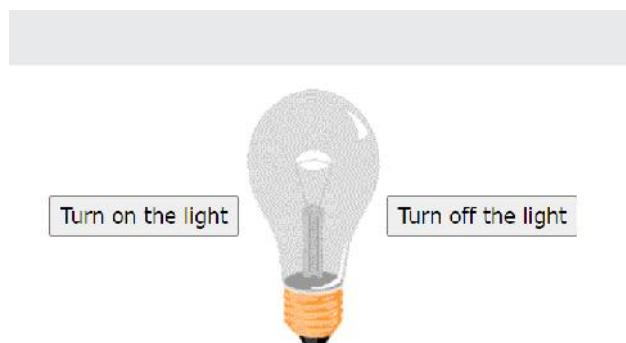
#### ***JavaScript Can Change HTML Content***

- One of many JavaScript HTML methods is getElementById().
- The example below "finds" an HTML element (with id="demo"), and changes the element content (innerHTML) to "Hello JavaScript":

- Example
- `document.getElementById("demo").innerHTML = "Hello JavaScript";`

JavaScript Can Change HTML  
Attribute Values

JavaScript changes the value of the src (source) attribute of an <img> tag



JavaScript changes the value of the src (source) attribute of an <img> tag



Changing the style of an HTML element, is a variant of changing an HTML attribute:

Example

```
document.getElementById("demo").style.fontSize = "35px";
```

Hiding HTML elements can be done by changing the display style:

Example

```
document.getElementById("demo").style.display = "none";
```

## The <script> Tag

In HTML, JavaScript code is inserted between <script> and </script> tags.

Example

```
<script>
document.getElementById("demo").innerHTML = "My First JavaScript";
</script>
```

### *JavaScript Functions and Events*

A JavaScript function is a block of JavaScript code, that can be executed when "called" for.

For example, a function can be called when an event occurs, like when the user clicks a button.

### *JavaScript in <head> or <body>*

- You can place any number of scripts in an HTML document.
- Scripts can be placed in the <body>, or in the <head> section of an HTML page, or in both.

- In following example (next slide), a JavaScript function is placed in the <head> section of an HTML page.
- The function is invoked (called) when a button is clicked:

```
<!DOCTYPE html>
<html>
<head>
<script>
function myFunction() {
  document.getElementById("demo").innerHTML = "Paragraph changed.";
}
</script>
</head>
<body><h2>Demo JavaScript in Head</h2>

<p id="demo">A Paragraph</p>
<button type="button" onclick="myFunction()">Try it</button>

</body>
</html>
```

### ***JavaScript in <body>***

- In this example (next slide), a JavaScript function is placed in the <body> section of an HTML page.
- The function is invoked (called) when a button is clicked:

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

<h2>Demo JavaScript in Body</h2>

<p id="demo">A Paragraph</p>

<button type="button" onclick="myFunction()">Try it</button>

<script>
function myFunction() {
  document.getElementById("demo").innerHTML = "Paragraph changed.";
}
</script>

</body>
</html>
```

### **External JavaScript**

- External scripts are practical when the same code is used in many different web pages.
- JavaScript files have the file extension .js.
- To use an external script, put the name of the script file in the src (source) attribute of a <script> tag:



Example:

```
<script src="myScript.js"></script>
```

You can place an external script reference in <head> or <body> as you like. The script will behave as if it was located exactly where the <script> tag is located.

### ***External JavaScript Advantages***

Placing scripts in external files has some advantages:

- It separates HTML and code
- It makes HTML and JavaScript easier to read and maintain
- Cached JavaScript files can speed up page loads

### ***JavaScript Output***

- JavaScript Display Possibilities
- JavaScript can "display" data in different ways:
  - Writing into an HTML element, using innerHTML.
  - Writing into the HTML output using document.write().
  - Writing into an alert box, using window.alert().
  - Writing into the browser console, using console.log().

#### ***Using innerHTML***

- To access an HTML element, JavaScript can use the document.getElementById(id) method.
- The id attribute defines the HTML element. The innerHTML property defines the HTML content

#### ***Using document.write()***

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

<h1>My First Web Page</h1>
<p>My first paragraph.</p>

<script>
document.write(5 + 6);
</script>

</body>
</html>
```

#### ***Using window.alert()***

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

<h1>My First Web Page</h1>
<p>My first paragraph.</p>

<script>
window.alert(5 + 6);
</script>

</body>
</html>
```

### ***JavaScript Comments***

- JavaScript comments can be used to explain JavaScript code, and to make it more readable.
- JavaScript comments can also be used to prevent execution, when testing alternative code.

#### ***Single Line Comments***

- Single line comments start with //.
- Any text between // and the end of the line will be ignored by JavaScript (will not be executed).

#### ***Multi-line Comments***

- Multi-line comments start with /\* and end with \*/.
- Any text between /\* and \*/ will be ignored by JavaScript.

## **Summary**

- A lightweight programming language ("scripting language") used to make web pages interactive insert dynamic text into HTML.
- Scripts are what the programme's in this language are known as. They can be directly written in the HTML of a web page and executed when the page loads.
- A JavaScript function is a block of JavaScript code, that can be executed when "called" for
- Placing scripts in external files has some advantages: It separates HTML and code
- Split breaks apart a string into an array using a delimiter. It can also be used with regular expressions (seen later). Join merges an array into a single string, placing a delimiter between them
- JavaScript comments can be used to explain JavaScript code, and to make it more readable.
- External scripts are practical when the same code is used in many different web pages. JavaScript files have the file extension .js.
- One of many JavaScript HTML methods is getElementById().

## **Keywords**

**Javascript:** Bootstrap is a free front-end framework for faster and easier web development. It also give you the ability to easily create responsive designs.

**JavaScript Developer-** To use JavaScript in developing a web application you must have at least two things, a browser, and an editor to write the JavaScript code.

**External scripts-** The are practical when the same code is used in many different web pages. JavaScript files have the file extension .js.

**Inner HTML-** To access an HTML element, JavaScript can use the document.getElementById(id) method.

**JavaScript Function-** A JavaScript function is a block of JavaScript code, that can be executed when "called" for

**IDEs for JavaScript-** JavaScript code using a simple editor like Notepad. However, you can install any open-sourced or licensed IDE (Integrated Development Environment)

## **Self Assessment**

1. JavaScript is an important\_\_\_\_\_ scripting language and widely used in dynamic websites.  
A. client-side  
B. Server side

- C. Both of above
  - D. None of above
2. It is a technique used in web development in which scripts run on the client's browser
- A. client-side scripting
  - B. Server side
  - C. Both of above
  - D. None of above
3. It is a technique that uses scripts on the webserver to produce a response that is customized for each client's request.
- A. client-side scripting
  - B. Server side Scripting
  - C. Both of above
  - D. None of above
4. In which element we put javascript?
- A. <javascript>
  - B. <scripting>
  - C. <script>
  - D. None of These
5. How we write "Hello World" in an alert box?
- A. alertBox("Hello World");
  - B. msgBox("Hello World");
  - C. msg("Hello World");
  - D. alert("Hello World");
6. What are advantages of JavaScript
- A. Speed
  - B. Interoperability
  - C. Simplicity
  - D. All of the above
7. Which type of JavaScript language is \_\_\_\_?
- A. Object-oriented
  - B. Object-based
  - C. Functional programming
  - D. All of the above
8. Which symbol is used separate JavaScript statements?
- A. Comma (,)
  - B. Colon (:)
  - C. Hyphen (-)
  - D. Semicolon (;)
9. JavaScript ignores?
- A. newlines
  - B. tabs
  - C. spaces

- D. All of the above
10. Which JavaScript method is used to access an HTML element by id?
- A. getElementById()
  - B. getElement(id)
  - C. getElementById(id)
  - D. elementById(id)
11. Which property is used to define the HTML content to an HTML element with a specific id?
- A. innerText
  - B. innerContent
  - C. elementText
  - D. innerHTML
12. Which JavaScript method is used to write HTML output?
- A. document.write()
  - B. document.output()
  - C. console.log()
  - D. document.writeHTML()
13. Which JavaScript method is used to write on browser's console?
- A. console.write()
  - B. console.output()
  - C. console.log()
  - D. console.writeHTML()
14. How many keywords are there in JavaScript to declare variables or constants?
- A. 1
  - B. 2
  - C. 3
  - D. 4
15. What is the main difference between var and let keywords in JavaScript?
- A. var defines a variable while let defines a constant
  - B. var defined function scoped variable while let define block scoped variable
  - C. The value of a variable declared with var can be changed while the value of a variable declared with let cannot be changed
  - D. All of the above

### **Answers for Self Assessment**

- |       |       |       |       |       |
|-------|-------|-------|-------|-------|
| 1. A  | 2. A  | 3. B  | 4. C  | 5. D  |
| 6. A  | 7. B  | 8. D  | 9. D  | 10. C |
| 11. D | 12. A | 13. C | 14. C | 15. B |

## **Review Questions**

1. Explain JavaScript? Explain its application in HTML?
2. Which IDE can be used implementing JavaScript?
3. What is equality in JavaScript
4. What's the difference between Host objects and Native objects?
5. How to compare two objects in JavaScript?
6. How JavaScript can be used in inner HTML?
7. Explain Prototype Inheritance in JavaScript?
8. What is the new keyword in JavaScript?
9. Is JavaScript a pass-by-reference or pass-by-value language?
10. Why Script Tag is Used in JavaScript?



## **Further Readings**

HTML, JavaScript , DHTML and PHP By. Ivan Bayross .

Beginning JavaScript 2<sup>nd</sup> Edition By. Paul Wilton

Html 5 Black Book, Covers Css 3, Javascript, Xml, Xhtml, Ajax, Php And Jquery, Second Edition



## **Web Links**

<https://www.javascripttutorial.net/javascript-function/>

[www.webopedia.com](http://www.webopedia.com)

[www.web-source.net](http://www.web-source.net)

## **UNIT 10: Functions and Arrays in Java Script**

### **Objectives**

After studying this unit, you will be able to:

- Discuss introduction regarding Variables
- Explain Function and Scope
- What are JavaScript Types and Common Language Constructs
- Handling of default value in JavaScript

### **Introduction**

To work with JavaScript as other programming languages, JavaScript has variables for the storage of some value. Variables can be assumed of as named containers. You can place data into these containers and then refer to the data simply by naming the container.

Before we use a variable in a JavaScript program, it must be declared with the var keyword as follows.

```
<script type = "text/javascript">  
<!--  
    var price;  
    var name;  
//-->  
</script>
```

You can also declare multiple variables with the same var keyword as follows –

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

Storing or placing a value in a variable is called variable initialization. Variable initialization at the time of variable creation or when you need that variable.

For instance, we might create a variable named price and assign the value 2000 to it later. For another variable, you can assign a value at the time of initialization as follows.

```
<script type = "text/javascript">  
<!--  
    var name = "hardisk";  
    var price;  
price = 2000;  
//-->  
</script>
```



Example: The var keyword is used only for declaration or initialization, only once variable name is declared, and variable name cannot be same in document.

JavaScript is an untyped language which means JavaScript variable can hold a value of any data type. Unlike many other languages, you don't need to tell JavaScript during variable declaration what type of value the variable will hold. The value type of a variable can change during the execution of a program and JavaScript takes care of it automatically.

### JavaScript Variable Scope

The scope of a variable is the section of a program in which it is defined. JavaScript variables have only two scopes.

Global Variables – A global variable has global scope which means it can be defined anywhere in your JavaScript code.

Local Variables – A local variable will be visible only within a function where it is defined. Function parameters are always local to that function.

Within the body of a function, a local variable takes priority over a global variable with the same name. If we have declared a local variable or function parameter with the same name as a global variable, we effectively abstract the global variable. For example.

```
<html>
<body onload = vscope();>
<script type = "text/javascript">
<!--
    var myVar = "global";    // Declare a global variable
    function vscope() {
        var myVar = "local";  // Declare a local variable
        document.write(myVar);
    }
//-->
</script>
</body>
</html>
```

This produces the following result –

local

#### JavaScript Variable Names

While naming our variables in JavaScript, following rules must be kept in mind.

We must not use any type of JavaScript reserved keywords as a variable name. These keywords are mentioned in the table below. For example, break or boolean variable names are not valid.

JavaScript variable names should not start with a numeral (0-9). They must begin with a letter or an underscore character. For example, 12test is an invalid variable name but \_12test is a valid one.

JavaScript variable names are case-sensitive. For example, Name and name are two different variables.

#### JavaScript Reserved Words

## 10.1 Keywords used in JavaScript

abstract	else	instanceof	switch
boolean	enum	int	synchronized
break	export	interface	this
byte	extends	long	throw
case	false	native	throws
catch	final	new	transient
char	finally	null	true
class	float	package	try
const	for	private	typeof
continue	function	protected	var
debugger	goto	public	void
default	if	return	volatile
delete	implements	short	while
do	import	static	with
double	in	super	

## 10.2 Functions and Scope

Function is a set of reusable code which may be called anywhere in your program. This removes the need of writing the identical code multiple times. It allows programmers in writing modular codes. Functions permit a programmer to divide a massive software into some small and manageable functions. Like every other superior programming language, JavaScript additionally helps all of the functions essential to jot down modular code with the usage of functions. Some more functions like `alert()` and `write()` are also used. We are using functions again and again, however, they were written in core JavaScript only once. JavaScript lets us to write our own functions as well. The section shows how to write our known JavaScript function.

Before using a function, we need to define it. The most common way to define a function in JavaScript is by using the function keyword, followed by a unique function name, a list of parameters (that might be empty), and a statement block surrounded by curly braces.

Basic syntax of function:

```
<script type="text/javascript">  
!--  
function functionname(parameter-list)  
{  
    statements  
}  
//-->  
</script>
```

For example. It defines a function called Helloworld that takes no parameters:

```
<script type="text/javascript">  
!--  
function start()  
{  
    alert("Hello World");  
}  
//-->  
</script>
```

### **10.3 Calling a Function**

To invoke a function we would simply need to write the name of that function as shown in the following code.

```
<html>  
<head>  
<script type="text/javascript">  
function test()  
{  
    document.write("Hello World!");  
}  
</script>  
</head>  
  
<body>  
<p>Click to call the function</p>  
<form>  
<input type="button" onclick="test()" value="Hello World" >
```

```
</form>

<p>Use some different text and try again...</p>
</body>
</html>
```

**Output:**

**Click to call the function**

Hello World

#### **10.4 Function Parameters**

As we have seen in the above section functions without parameters. But there is an ability to pass different parameters while calling a function. These passed parameters can be captured inside the function and any manipulation can be done over those parameters. A function can take multiple parameters separated by a comma.

For example we have modified our test function here. Now it takes two parameters.

```
<html>
<head>
<script type="text/javascript">functiontest(name,age)
{
    document.write(name+"is"+age+"yearsold.");
}
</script>
</head>

<body>
<p>Click to call the function</p>
<form>
<input type="button" onclick="test('Aseem',7)" value="Hello world">
</form>

<p>Usedifferentparametersinsidethefunctionandthentry...</p>
</body>
</html>
```

**Output**

Click to call the function

**HELLO WORLD**

Use different parameters inside the function and then try...

## **10.5 The return Statement**

A JavaScript function can have an optional return statement. This is required if you want to return a value from a function. This statement should be the last statement in a function.

For example, you can pass two numbers in a function, and then you can expect the function to return their multiplication in your calling program.

```
<html>
<head>
<script
type="text/javascript">functioncon
catenate(first,last)
{
    varfull;

    full=first+last;re
    turnfull;
}
functionsecondFunction()
{
    varresult;
    result=concatenate('Zara','Ali');document.write(re
sult);
}
</script>
</head>
<body>
<p>Click the following button to call the function</p>
<form>
<input type="button" onclick="secondFunction()" value="CallFunction">
</form>

<p>Use different parameters inside the function and then try...</p>
</body>
</html>
```

Click the following button to call the function

**Call Function**

Use different parameters inside the function and then try...

## **10.6 JavaScript Types**

JavaScript provides different data types to hold different types of values. There are two types of data types in JavaScript.

- Primitive data type
- Non-primitive (reference) data type

JavaScript is a dynamic type language; means there is no need to specify type of the variable because it is dynamically used by JavaScript engine. Only the need is to use var here to specify the data type. It can hold any type of values such as numbers, strings etc. For example:

```
var a=20;//holding number  
var b="AseemKhana";//holding string
```

### **JavaScript primitive data types**

There are five types of primitive data types in JavaScript. They are as follows:

Data Type	Description
String	represents sequence of characters e.g. "aseem"
Number	represents numeric values e.g. 50
Boolean	represents boolean value either false or true
Undefined	represents undefined value
Null	represents null i.e. no value at all

### **JavaScript non-primitive data types**

The non-primitive data types are as follows:

Data Type	Description
Object	represents instance through which we can access members
Array	represents group of similar values

RegExp	represents regular expression
--------	-------------------------------

## **10.7 JavaScript Default Parameters**

Default function parameters are the parameters in JavaScript that provide a default value while declaring the JavaScript function. This is useful, as we can provide default values to the parameters, which can be used if a value is not provided when the function is called.

So default function parameters are used to initialize the named parameter with a default value if no value or undefined is passed during the function call.

For example, if we define a function in which we provide two variables, it divides the first parameter with the second parameter and returns the result of the division operator. Now if the user provides only one parameter, then the function call will fail. For avoiding this, you can provide a default value to the second parameter as 1, which will divide the single parameter passed during the function call with the default value 1, rather than giving an error.

By default, if a function parameter is not provided with a value during the function call, undefined is set in it.

JavaScript Default Parameters Syntax:

```
function function_name([param1[ = defaultValue1][, ..., paramS[ = defaultValueS ]]])  
{  
    statements  
}
```

One important point to note that if you have multiple parameters in a function, say 2, and you provide a default value to the first one and leave the second parameter without a default value, then during the call, if that function is called with a single argument, it will get assigned to first parameter( one with default parameter value) and the second will get the undefined value, this is because parameter assignment happens from left to right. So you should not have parameters without default values after default parameters.

```
function printValue(a=2, b) {  
    console.log("a = " + a + " and b = " + b);  
}  
  
printValue(); // Logs: a = 2 and b = undefined  
printValue(8); // Logs: a = 8 and b = undefined  
printValue(8, 5); // Logs: a = 8 and b = 5
```

The above, when we provided no argument while calling the function then, a used its default value while b got undefined. So provided one argument while calling the function, and that argument got assigned to the first parameter which is a, as argument assignment to parameters happens in order, hence b again got undefined.

Default values for parameters and calling it without arguments are explained in below example.

```
// default function parameters
```

```
function add(a=10, b=20)
{
    return a+b;
}

console.log(" Sum is : " + add()); // No argument
console.log(" Sum is : " + add(1)); // with one argument
console.log(" Sum is : " + add(5,6)); // with both argument
```

#### Output

```
Sum is : 30
Sum is : 21
Sum is : 11
```

## **Summary**

- JavaScript supports a number of types of data, Such as numbers, text, and Booleans.
- Variables are Java Scripts's way of storing data, such as numbers and text, in memory so that they can be used again and again in your code.
- Variable names must not include certain illegal characters, like % and &, or be reserved word, like with.
- Use a function keyword to declare a function.
- FunctionName() is used to call a function.
- All functions implicitly return undefined if they don't explicitly return a value.
- the return statement to return a value from a function explicitly can be used.
- The arguments variable is an array-like object inside a function, representing function arguments.
- The function hoisting allows you to call a function before declaring it.

## **Keywords**

**Variables:** Variables can be assumed as named containers. You can place data into these containers and then refer to the data simply by naming the container.

**Functions:** A function is a set of reusable codesthat may be called anywhere in your program. This removes the need of writing the identical code multiple times.

**JavaScript Datatype:** JavaScript is a dynamic type language; means there is no need to specify type of the variable because it is dynamically used by JavaScript engine.

**Number:** Number variables are just that—numbers. JavaScript does not require numbers to be declared as integers, floating-point (decimal) numbers, or any other number type.

**Strings:**String variables are variables that represent a string of text. The string may contain letters, words, spaces, numbers, symbols, or most anything you like

**Parameters:**Parameters are used to allow a function to import one or more values from somewhere outside the function

**Global Variables:**They are defined outside any functions; they can be changed anywhere in the script—inside or outside of functions

**Local Variables:**A local variable can be used only within the function in which it is declared. It does not exist outside that function, unless you pass it along to another function by using a parameter.

## **Self Assessment**

1. The const keyword is used to define a \_\_\_\_\_.
  - A. Function scopes variable
  - B. Block scoped variable
  - C. Constant
  - D. Constant with no initial value
  
2. Which is the correct syntax to declare a constant in JavaScript?
  - A. const constant\_name;
  - B. constant\_name const;
  - C. constant\_name const = value;
  - D. const constant\_name = value;
  
3. What will be the value of VALUE?

```
<script>
    const VALUE = 10;
    VALUE = 20;
</script>
```

  - A. 10
  - B. 20
  - C. ValueError
  - D. TypeError
  
4. What is the default value of an uninitialized variable?
  - A. 0
  - B. undefined
  - C. null
  - D. NaN
  
5. Can be redeclare a variable that is declared with var keyword?
  - A. Yes
  - B. No
  - C. Both A and B
  - D. None of above
  
6. \_\_\_\_\_ are containers for storing data
  - A. Variables
  - B. Arrays
  - C. Functions
  - D. All of the above
  
7. JavaScript arrays are written with \_\_\_\_\_.
  - A. round brackets ()
  - B. curly brackets {}
  - C. double quotes ""
  - D. square brackets []
  
8. JavaScript objects are written with \_\_\_\_\_.
  - A. round brackets ()
  - B. curly brackets {}
  - C. double quotes ""
  - D. square brackets []
  
9. Which JavaScript operator is used to determine the type of a variable?
  - A. typeof

- B. sizeOf
- C. typOf
- D. sizeof

10. Which keyword is used to define a JavaScript function?

- A. module
- B. fun
- C. func
- D. function

11. Can we use a function as a variable value?

- A. True
- B. False

12. In JavaScript a variable contains one value while an object may contain \_\_\_\_.

- A. One value
- B. Two values
- C. Three values
- D. Many values

13. Which is the correct syntax to access an object property in JavaScript?

- A. objectName:propertyName
- B. propertyName
- C. objectName["propertyName"]
- D. Both B. and C.

14. Which property is used to get the length of a string in JavaScript?

- A. strlen
- B. len
- C. length
- D. Lent

15. Which character is used to break up a code line within a text string in JavaScript?

- A. Single quote ()
- B. Single backslash (\)
- C. Double quote ("")
- D. Tipple single quote ("")

### **Answers for Self Assessment**

- |       |       |       |       |       |
|-------|-------|-------|-------|-------|
| 1. C  | 2. D  | 3. D  | 4. B  | 5. A  |
| 6. A  | 7. D  | 8. B  | 9. A  | 10. D |
| 11. A | 12. D | 13. D | 14. C | 15. B |

### **Review Questions**

1. Write a JavaScript code block using arrays and generate the current date in words, this should include the day, the month and the year. The output should be as follows, Saturday, October 09,2021.

2. What if I put a function into my script but decide not to call it in the script? Will it matter?
  3. What happens if I decide to remove a function from my script later?
  4. What is the best way to determine when to use a function and when to just code what I want right into the script?
  5. Write a JavaScript function which accepts an argument and returns the type
- Note : There are six possible values that typeof returns: object, boolean, function, number, string, and undefined.



### **Further Readings**

HTML, JavaScript , DHTML and PHP By. Ivan Bayross.

Beginning JavaScript 2<sup>nd</sup> Edition By. Paul Wilton

Html 5 Black Book, Covers Css 3, Javascript, Xml, Xhtml, Ajax, Php And Jquery, Second Edition



### **Web Links**

<https://www.javascripttutorial.net/javascript-function/>

[www.webopedia.com](http://www.webopedia.com)

[www.web-source.net](http://www.web-source.net)

## Unit 11: Java Script Events and Validations

### CONTENTS

- Objectives
- Introduction
- 11.1 Anatomy of an HTML document
- 11.2 HTML Editors
- 11.3 Marking up text
- 11.4 HTML Attributes
- Summary
- Keywords
- Self Assessment
- Answers for Self Assessment
- Review Questions
- Further Readings

### Objectives

- Learn about what is HTML?
- How HTML will help in Web development
- Understand about different tags

### Introduction

HTML (HyperText Markup Language) is the code that is used to structure a web page and its content. For example, content could be structured within a set of paragraphs, a list of bulleted points, or using images and data tables. HTML was first created by Tim Berners-Lee, Robert Cailliau, and others starting in 1989. It stands for Hyper Text Markup Language.

Hypertext means that the document contains links that allow the reader to jump to other places in the document or to another document altogether. The latest version is known as HTML5.

A Markup Language is a way that computers speak to each other to control how text is processed and presented. To do this HTML uses two things: tags and attributes.

### So what is HTML?

HTML is a markup language that defines the structure of your content. HTML consists of a series of elements, which you use to enclose, or wrap, different parts of the content to make it appear a certain way, or act a certain way. The enclosing tags can make a word or image hyperlink to somewhere else, can italicize words, can make the font bigger or smaller, and so on. For example, take the following line of content:

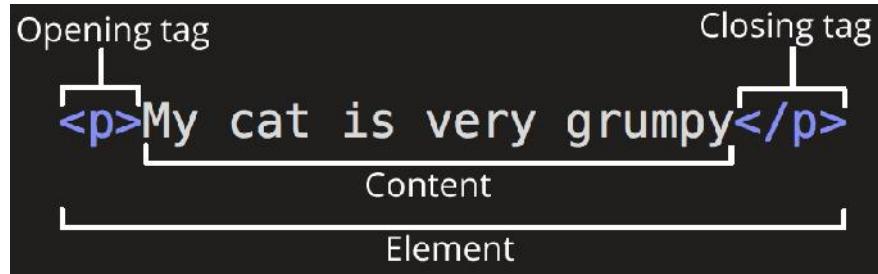
**My cat is very grumpy**

If we wanted the line to stand by itself, we could specify that it is a paragraph by enclosing it in paragraph tags:

```
<p>My cat is very grumpy</p>
```

### Anatomy of an HTML element

Let's explore this paragraph element a bit further.



*Figure 1 Anatomy of an HTML element*

The main parts of our element are as follows:

- **The opening tag:** This consists of the name of the element (in this case, p), wrapped in opening and closing angle brackets. This states where the element begins or starts to take effect – in this case where the paragraph begins.
- **The closing tag:** This is the same as the opening tag, except that it includes a forward slash before the element name. This states where the element ends – in this case where the paragraph ends. Failing to add a closing tag is one of the standard beginner errors and can lead to strange results.
- **The content:** This is the content of the element, which in this case, is just text.
- **The element:** The opening tag, the closing tag, and the content together comprise the element.

Elements can also have attributes that look like the following:

```
Attribute
<p class="editor-note">My cat is very grumpy</p>
```

Attributes contain extra information about the element that you don't want to appear in the actual content. Here, class is the attribute name and editor-note is the attribute value. The class attribute allows you to give the element a non-unique identifier that can be used to target it (and any other elements with the same class value) with style information and other things.

An attribute should always have the following:

- A space between it and the element name (or the previous attribute, if the element already has one or more attributes).
- The attribute name followed by an equal sign.
- The attribute value wrapped by opening and closing quotation marks.



**Notes:** Simple attribute values that don't contain ASCII whitespace (or any of the characters " ' ` = <> ) can remain unquoted, but it is recommended that you quote all attribute values, as it makes the code more consistent and understandable.

## Nested Elements

You can put elements inside other elements too — this is called nesting. If we wanted to state that our cat is very grumpy, we could wrap the word "very" in a `<strong>` element, which means that the word is to be strongly emphasized:

```
<p>My cat is <strong>very</strong> grumpy. </p>
```

You do however need to make sure that your elements are properly nested. In the example above, we opened the `<p>` element first, then the `<strong>` element; therefore, we have to close the `<strong>` element first, then the `<p>` element. The following is incorrect:

```
<p>My cat is <strong>very grumpy.</p></strong> ×
```

The elements have to open and close correctly so that they are clearly inside or outside one another. If they overlap as shown above, then your web browser will try to make the best guess at what you were trying to say, which can lead to unexpected results. So don't do it!

## Empty Elements

Some elements have no content and are called empty elements. Take the `<img>` element that we already have in our HTML page:

```
<imgsrc="images/firefox-icon.png" alt="My test image">
```

This contains two attributes, but there is no closing `</img>` tag and no inner content. This is because an image element doesn't wrap content to affect it. Its purpose is to embed an image in the HTML page in the place it appears.

## 11.1 Anatomy of an HTML document

HTML elements are the building blocks of HTML pages. With HTML constructs, images and other objects such as interactive forms may be embedded into the rendered page. HTML provides a means to create structured documents by denoting structural semantics for text such as headings, paragraphs, lists, links, quotes and other items. HTML elements are delineated by tags, written using angle brackets. Tags such as `<img />` and `<input />` directly introduce content into the page. Other tags such as `<p>` surround and provide information about document text and may include other tags as sub-elements. Browsers do not display the HTML tags but use them to interpret the content of the page.

HTML can embed programs written in a scripting language such as JavaScript, which affects the behavior and content of web pages. Inclusion of CSS defines the look and layout of content. The World Wide Web Consortium (W3C), former maintainer of the HTML and current maintainer of the CSS standards, has encouraged the use of CSS over explicit presentational HTML since 1997.[ A form of HTML, known as HTML5, is used to display video and audio, primarily using the `<canvas>` element, in collaboration with javascript.

All HTML documents must start with a document type declaration: `<!DOCTYPE html>`.

The HTML document itself begins with `<html>` and ends with `</html>`.

The visible part of the HTML document is between `<body>` and `</body>`.

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>My test page</title>
</head>
<body>
<imgsrc="images/firefox-icon.png" alt="My test image">
```

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

Here, we have the following:

- **<!DOCTYPE html>** – doctype. It is a required preamble. In the mists of time, when HTML was young (around 1991/92), doctypes were meant to act as links to a set of rules that the HTML page had to follow to be considered good HTML, which could mean automatic error checking and other useful things. However these days, they don't do much and are basically just needed to make sure your document behaves correctly. That's all you need to know for now.
- **<html></html>** – the **<html>** element. This element wraps all the content on the entire page and is sometimes known as the root element.
- **<head></head>** – the **<head>** element. This element acts as a container for all the stuff you want to include on the HTML page that isn't the content you are showing to your page's viewers. This includes things like keywords and a page description that you want to appear in search results, CSS to style our content, character set declarations, and more.
- **<meta charset="utf-8">** – This element sets the character set your document should use to UTF-8 which includes most characters from the vast majority of written languages. Essentially, it can now handle any textual content you might put on it. There is no reason not to set this and it can help avoid some problems later on.
- **<title></title>** – the **<title>** element. This sets the title of your page, which is the title that appears in the browser tab the page is loaded in. It is also used to describe the page when you bookmark/favorite it.
- **<body></body>** – the **<body>** element. This contains all the content that you want to show to web users when they visit your page, whether that's text, images, videos, games, playable audio tracks, or whatever else.

## **11.2 HTML Editors**

Web pages can be created and modified by using professional HTML editors.

However, for learning HTML we recommend a simple text editor like Notepad (PC) orTextEdit (Mac). We believe in that using a simple text editor is a good way to learn HTML. Text editors intended for use with HTML usually provide at least syntax highlighting. Some editors additionally feature templates, toolbars and keyboard shortcuts to quickly insert common HTML elements and structures. Wizards, tooltip prompts and autocompletion may help with common tasks.

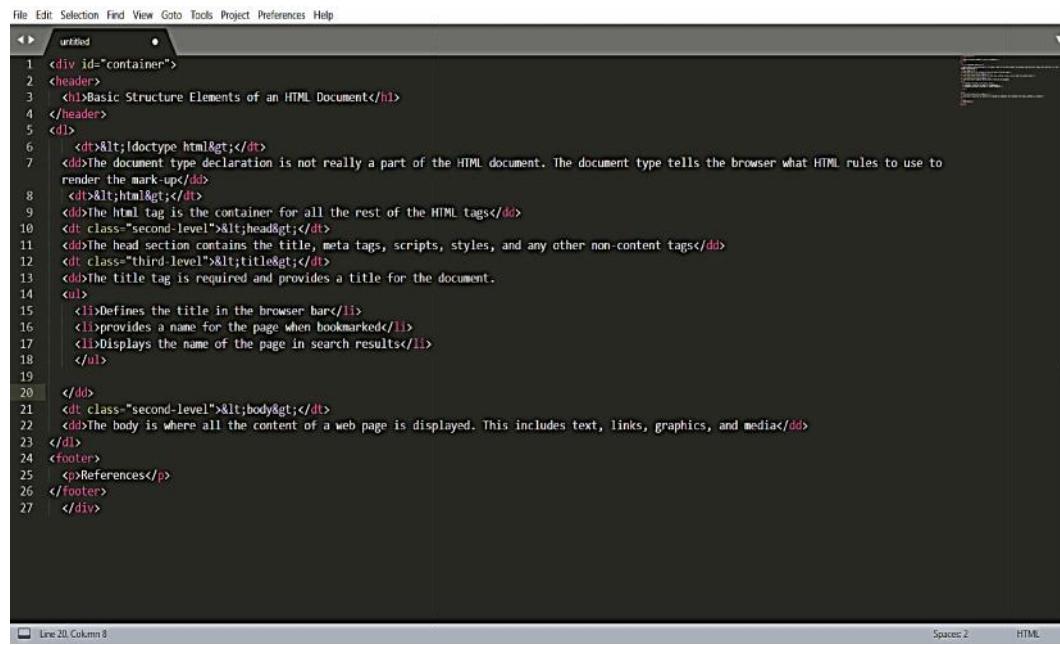
Text editors commonly used for HTML typically include either built-in functions or integration with external tools for such tasks as version control, link-checking and validation, code cleanup and formatting, spell-checking, uploading by FTP or WebDAV, and structuring as a project. Some functions, such as link checking or validation may use online tools, requiring a network connection.

Text editors require user understanding of HTML and any other web technologies the designer wishes to use like CSS, JavaScript and server-side scripting languages.

To ease this requirement, some editors allow editing of the markup in more visually organized modes than simple color highlighting, but in modes not considered WYSIWYG. These editors typically include the option of using palette windows or dialog boxes to edit the text-based parameters of selected objects. These palettes allow editing parameters in individual fields, or inserting new tags by filling out an onscreen form, and may include additional widgets to present and select options when editing parameters (such as previewing an image or text styles) or an outline editor to expand and collapse HTML objects and properties. Below are some of the text editors that can be used for HTML.

### Sublime Text 3

Sublime Text 3 as it is free and also offers cross-platform support for Windows, Mac, and Linux users.



```

File Edit Selection Find View Goto Tools Project Preferences Help
untitled
1 <div id="container">
2 <header>
3   <h1>Basic Structure Elements of an HTML Document</h1>
4 </header>
5 <dd>
6   <dt>&lt;!DOCTYPE HTML&gt;</dt>
7   <dd>The document type declaration is not really a part of the HTML document. The document type tells the browser what HTML rules to use to render the mark-up</dd>
8   <dt>&lt;html&gt;</dt>
9   <dd>The html tag is the container for all the rest of the HTML tags</dd>
10  <dt class="second-level">&lt;head&gt;</dt>
11  <dd>The head section contains the title, meta tags, scripts, styles, and any other non-content tags</dd>
12  <dt class="third-level">&lt;title&gt;</dt>
13  <dd>The title tag is required and provides a title for the document.
14 <ul>
15   <li>Defines the title in the browser bar</li>
16   <li>Provides a name for the page when bookmarked</li>
17   <li>Displays the name of the page in search results</li>
18 </ul>
19
20 </dd>
21 <dt class="second-level">&lt;body&gt;</dt>
22 <dd>The body is where all the content of a web page is displayed. This includes text, links, graphics, and media</dd>
23 </dl>
24 <footer>
25   <p>References</p>
26 </footer>
27 | </div>

```

Line 20, Column 8      Spaces: 2      HTML

#### Pros

- Easily customizable
- Beginner-friendly
- Pleasant color schemes to choose from.

#### Cons

- Can't print documents or code
- No toolbar or dashboard available.

#### Notepad ++

Another common choice for HTML and other language coders is Notepad++. It is a tiny program to download and perform the functions you need for writing clean code.

By Eli, Scott, You, Specia, Lengyel, Sjogren, Tech, Meier, Ben Shahn, Winkler, L

HyperText Markup Language file

```
1 <div><strong><a href="#"></a></strong></div><h1>Headings</h1><h2>Headings</h2><h3>Headings</h3><h4>Headings</h4><h5>Headings</h5><h6>Headings</h6>
```

2 <div>Headings are a way to make text stand out by breaking up the page.</div>

```
3 <strong><a href="#">Paragraphs</a></strong>
```

4 <div>Paragraphs determine line spacing.</div>

```
5 <strong><a href="#">Tables</a></strong>
```

6 <div>Tables create <tr><td> content just like in a word processor.</div>

```
7 <strong><a href="#">Bold</a></strong>
```

8 <div>Strong bold <strong>emphasizes</strong> keywords.</div>

```
9 <strong><a href="#">Italics</a></strong>
```

```
10 <strong><a href="#">Underlined</a></strong>
```

11 <div>The anchor tag is not commonly used to create links in combination with the <a style="background-color: #ffffcc; href="#button1-a-keef;">congratulations!</a> attribute.</div>

```
12 <strong><a href="#">Unordered List</a></strong>
```

13 <div>Unnumbered lists of bullet points use the Unordered List tag.</div>

```
14 <strong><a href="#">List Item</a></strong>
```

15 <div>Each line on a list is enclosed by a List Item tag.</div>

```
16 <strong><a href="#">Blockquote</a></strong>
```

17 <div>blockquote tags are used to enclose quotations from people. This tag helps to differentiate the quote from the text around it.</div>

```
18 <strong><a href="#">Horizontal Rule</a></strong>
```

19 <div>Horizontal rule is a straight line commonly used for dividing areas of a webpage.</div>

```
20 <strong><a href="#">Image</a></strong>
```

21 <div>Images can be used to make your page look prettier. Add your page with:

```
22 <strong><a href="#">Division</a></strong>
```

```
23 <div>The division tag defines specific layout styles within CSS.</div></div>
```

Pros

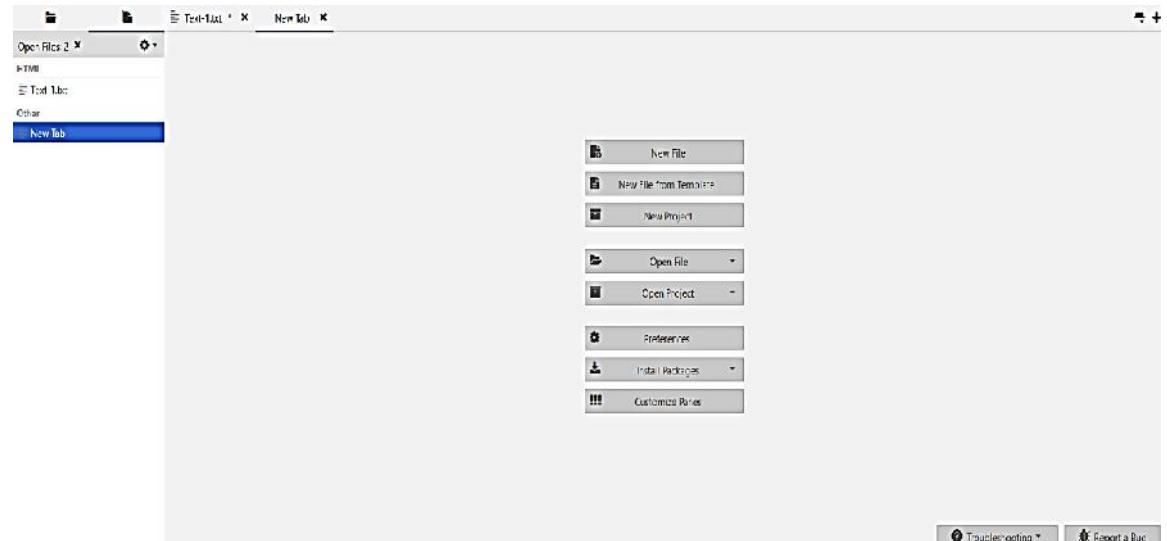
- Distraction-free interface
  - Auto-completion feature
  - Plugin options for extended functionalities

Cons

- Can be difficult to get used to for beginners
  - No support for Mac.

Komodo Edit

Komodo Edit is one of two editors released by the same label. They offer a simple, open-source editor with a variety of extensions and language support.



Pros

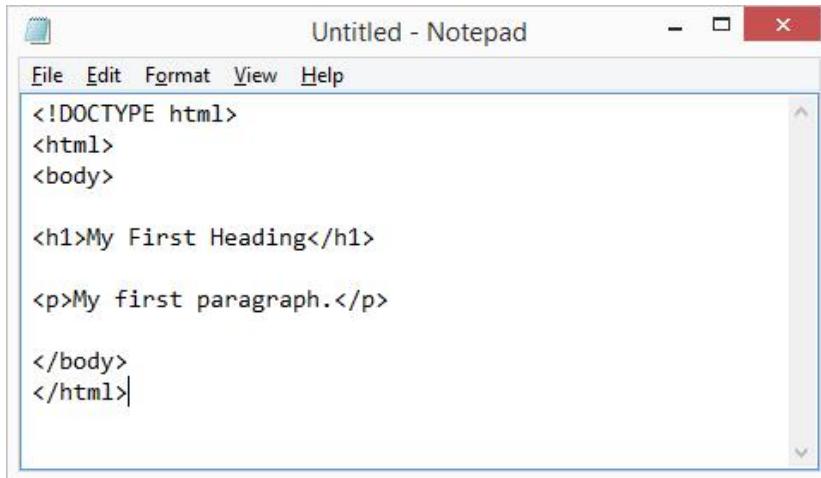
- Easy-to-grasp coding interface
  - Available for Mac, Windows, and Linux
  - Impressive language support.

**Cons**

- No autocompletion by default
- Visual settings are difficult to find and change.

**Notepad**

The best and easiest HTML editor that can be used in Notepad



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

<h1>My First Heading</h1>

<p>My first paragraph.</p>

</body>
</html>|
```

While creating a document just save the document with .HTML extension it will open in the web browser.

### **11.3 Marking up text**

This section will cover some of the essential HTML elements you'll use for marking up the text.

**Headings**

Heading elements allow you to specify that certain parts of your content are headings — or subheadings. In the same way that a book has the main title, chapter titles, and subtitles, an HTML document can too. HTML contains 6 heading levels, `<h1>` - `<h6>`, although you'll commonly only use 3 to 4 at most:

```
<!-- 4 heading levels: -->
<h1>My main title</h1>
<h2>My top level heading</h2>
<h3>My subheading</h3>
<h4>My sub-subheading</h4>
```

Note: Anything in HTML between `<!--` and `-->` is an HTML comment. The browser ignores comments as it renders the code. In other words, they are not visible on the page - just in the code. HTML comments are a way for you to write helpful notes about your code or logic.

**Paragraphs**

As explained above, `<p>` elements are for containing paragraphs of text; you'll use these frequently when marking up regular text content:

```
<p>This is a single paragraph</p>
```

Add your sample text (you should have it from What will your website look like?) into one or a few paragraphs, placed directly below your <img> element.

### Links

Links are very important — they are what makes the web a web! To add a link, we need to use a simple element — <a> — "a" being the short form for "anchor". To make text within your paragraph into a link, follow these steps:

- Choose some text. We choose the text "Google".
- Wrap the text in an <a> element, as shown below:

<a> Google </a>

- Give the <a> element an href attribute, as shown below:

<a href=""> Google </a>

Fill in the value of this attribute with the web address that you want the link to link to:

<a href=" https://www.google.com/">Google</a>

You might get unexpected results if you omit the https:// or http:// part, called the protocol, at the beginning of the web address. After making a link, click it to make sure it is sending you where you wanted it to.

### **11.4 HTML Attributes**

HTML attributes provide additional information about HTML elements.

#### Properties of HTML Attributes

- All HTML elements can have attributes
- Attributes provide additional information about elements
- Attributes are always specified in the start tag
- Attributes usually come in name/value pairs like: name="value"

#### The href Attribute

The <a> tag defines a hyperlink. The href attribute specifies the URL of the page the link goes to:

*Syntax:*

<a href=" https://www.google.com/">Google</a>

#### src Attribute

The <img> tag is used to embed an image in an HTML page. The src attribute specifies the path to the image to be displayed:



#### The width and height Attributes

The <img> tag should also contain the width and height attributes, which specifies the width and height of the image (in pixels):



## The alt Attribute

The required **alt** attribute for the **<img>** tag specifies an alternate text for an image, if the image for some reason cannot be displayed. This can be due to slow connection, or an error in the **src** attribute, or if the user uses a screen reader.

```
<imgsrc="img_girl.jpg" alt="this image is about">
```

## Style Attribute

The style attribute is used to add styles to an element, such as color, font, size, and more.

```
<p style="color:red;">This is a red paragraph.</p>
```

## The title Attribute

The title attribute defines some extra information about an element.

The value of the title attribute will be displayed as a tooltip when you mouse over the element:

```
<p title="I'm a tooltip">This is a paragraph.</p>
```



**Task:** Write down the code to create your first HTML page by using all the tags discussed above

## Summary

- All HTML elements can have attributes
- The href attribute of **<a>** specifies the URL of the page the link goes to
- The src attribute of **<img>** specifies the path to the image to be displayed
- The width and height attributes of **<img>** provide size information for images
- The alt attribute of **<img>** provides an alternate text for an image
- The style attribute is used to add styles to an element, such as color, font, size, and more
- The lang attribute of the **<html>** tag declares the language of the Web page
- The title attribute defines some extra information about an element

## Keywords

- **Hyper Text markup language:** HTML (HyperText Markup Language) is the most basic building block of the Web. It defines the meaning and structure of web content.
- **HTML tag:** An HTML tag is a piece of markup language used to indicate the beginning and end of an HTML element in an HTML document.
- **Title:** the **<title>** element. This sets the title of your page, which is the title that appears in the browser tab the page is loaded in. It is also used to describe the page when you bookmark/favorite it.
- **Attribute:** An HTML attribute is a piece of markup language used to adjust the behavior or display of an HTML element. For example, attributes can be used to change the color, size, or functionality of HTML elements.
- **HTML Editors:** HTML editor is a software used for writing code in HTML which is used for structuring and creating websites. Even though codes can be written from scratch using a normal text editor, HTML editors provide a great deal of ease to the developers by ensuring hassle-free coding.

## **SelfAssessment**

1. What is HTML?
  - A. HTML describes the structure of a webpage
  - B. HTML is the standard markup language mainly used to create web pages
  - C. HTML consists of a set of elements that helps the browser how to view the content
  - D. All of the mentioned
  
2. HTML stands for \_\_\_\_\_
  - A. HyperText Markup Language
  - B. HyperText Machine Language
  - C. HyperText Marking Language
  - D. HighText Marking Language
  
3. What is the correct syntax of doctype in HTML?
  - A. </doctype html>
  - B. <doctype html>
  - C. <doctype html!>
  - D. <!doctype html>
  
4. Which of the following tag is used for inserting the largest heading in HTML?
  - A. head
  - B. <h1>
  - C. <h6>
  - D. heading
  
5. In which part of the HTML metadata is contained?
  - A. head tag
  - B. title tag
  - C. html tag
  - D. body tag
  
6. How do we write comments in HTML?
  - A. </.....>
  - B. <!.....>
  - C. </...../>
  - D. <.....!>
  
7. The correct sequence of HTML tags for starting a webpage is -
  - A. Head, Title, HTML, body
  - B. HTML, Body, Title, Head

- C. HTML, Head, Title, Body
- D. HTML, Head, Title, Body

8. Which character is used to represent the closing of a tag in HTML?

- A. \
- B. !
- C. /
- D. .

9. ALL HTML tags are enclosed in what?

- A. # and #
- B. ? and !
- C. < and >
- D. { and }

10. To create HTML page, you need \_\_\_\_\_

- A. Web browser
- B. text editor
- C. Both [A] and [B]
- D. None of the above

11. The BODY tag is usually used after \_\_\_\_\_

- A. HTML tag
- B. EM tag
- C. TITLE tag
- D. HEAD tag

12. Which tag tells the browser where the page starts and stops?

- A. <html>
- B. <body>
- C. <head>
- D. <title>

13. Which program do you need to write HTML?

- A. A graphics program
- B. Any text editor
- C. HTML -development suite 4
- D. All of the above

14. HTML uses

- A. User defined tags

- B. Pre-specified tags
- C. Fixed tags defined by the language
- D. Tags only for linking

15. Fundamental HTML Block is known as \_\_\_\_\_.

- A. HTML Body
- B. HTML Tag
- C. HTML Attribute
- D. HTML Element

### **Answers for Self Assessment**

- |       |       |       |       |       |
|-------|-------|-------|-------|-------|
| 1. D  | 2. A  | 3. D  | 4. C  | 5. A  |
| 6. B  | 7. C  | 8. C  | 9. C  | 10. C |
| 11. D | 12. A | 13. B | 14. C | 15. B |

### **Review Questions**

1. What do you mean by HTML? Discuss the use of HTML
2. Discuss the difference between a Tag and an Attribute in HTML.
3. Write down the steps or code to create a simple HTML document and print Welcome to my new HTML page.
4. Explain all the tags that are used to create an HTML Page along with their functionality.
5. Discuss about some of the HTML editors that can be used for web development.



### **Further Readings**

<https://www.computer-pdf.com/web-programming/html/>



### **Web Links**

<https://matfuvit.github.io/UVIT/predavanja/literatura/TutorialsPoint%20HTML.pdf>

## Unit 12: JavaScript - Browser Object Model

### **CONTENTS**

Objectives

Introduction

12.1 HTML Tables

12.2 Creating Tables

12.3 Tables and the Border Attribute

12.4 Linking Document

12.5 Inserting inline images

12.6 Creating Image Links

12.7 IMG Attributes

12.8 HTML Lists

12.9 HTML List Classes

12.10 HTML List Types

12.11 HTML Block and Inline Elements

12.12 HTML Layout Elements and Techniques

12.13 HTML Forms

12.14 I frames in HTML

Summary

Keywords

Self Assessment

Answers for Self Assessment

Review Questions

Further Readings

### **Objectives**

- Describe about HTML Tables
- Explain how HTML Tables are created
- Explain the steps involved in creating HTML Tables
- Describe about Linking Documents
- Describe about Creating Link Lists

### **Introduction**

In this unit, we describe the statements for creating and updating tables. We assume that the user knows about the data which has to be stored and what the structure of the data is, i.e., what tables are to be created and what the appropriate columns are.

### **12.1 HTML Tables**

Tables are defined with the <table> tag. A table is divided into rows (with the <tr> tag), and each

row is divided into data cells (with the <td> tag). The letters td stands for “table data,” which is the content of a data cell. A data cell can contain text, images, lists, paragraphs, forms, horizontal rules, tables, etc.


**Example:**

```
<table border="1">
<tr>
<th>Heading</th>
<th>Another Heading</th>
</tr>
<tr>
<td>row 1, cell 1</td>
<td>row 1, cell 2</td>
</tr>
<tr>
<td>row 2, cell 1</td>
<td>row 2, cell 2</td>
</tr>
</table>
```

**Output**

<u>Heading</u>	<u>Another Heading</u>
row 1, cell 1	row 1, cell 2
row 2, cell 1	row 2, cell 2

## 12.2 Creating Tables

The basic structure of an HTML table consists of the following tags:      Table tags:  
 <TABLE></TABLE>

- Row tags: <TR></TR>
- Cell tags: <TD></TD>

Constructing an HTML table consists of describing the table between the beginning table tag,

<TABLE>, and the ending table tag, </TABLE>. Between these tags, you then construct each row and each cell in the row. To do this, you would first start the row with the beginning row tag,

LOVELY PROFESSIONAL UNIVERSITY 47

Unit 5: Creating Tables

<TR>, and then build the row by creating each cell with the beginning cell tag, <TD>, adding the Notes

data for that cell, and then closing the cell with the ending cell tag, </TD>. When you finish all

of the cells for a row, you would then close the row with the ending row tag, </TR>.



**Notes:** For each new row, you would repeat the process of beginning the row, building each cell in the row, and closing the row.



#### Example:

The following table is an example of a basic table with three rows and two columns of data.

Data 1 Data 2

Data 3 Data 4

Data 5 Data 6

The codes that generated this table will look like this:

```
<TABLE>
<TR>
<TD>Data 1</TD>
<TD>Data 2</TD>
</TR>
<TR>
<TD>Data 3</TD>
<TD>Data 4</TD>
</TR>
<TR>
<TD>Data 5</TD>
<TD>Data 6</TD>
</TR>
</TABLE>
```

This table contains no border, title, or headings. If you wish to add any of these elements to your table, you need to include additional HTML codes.

### **12.3 Tables and the Border Attribute**

If you do not specify a border attribute the table will be displayed without any borders. Sometimes this can be useful, but most of the time, you want the borders to show.

To display a table with borders, you will have to use the border attribute:

```
<table border="1">
<tr>
<td>Row 1, cell 1</td>
<td>Row 1, cell 2</td>
</tr>
</table>
```

## Headings in a Table

Headings in a table are defined with the <th> tag.

```
<table border="1">
<tr>
<th>Heading</th>
<th>Another Heading</th>
</tr>
<tr>
<td>row 1, cell 1</td>
<td>row 1, cell 2</td>
</tr>
<tr>
<td>row 2, cell 1</td>
<td>row 2, cell 2</td>
</tr>
</table>
```

### Output

Heading	Another Heading
row 1, cell 1	row 1, cell 2
row 2, cell 1	row 2, cell 2

## Empty Cells in a Table

Table cells with no content are not displayed very well in most browsers.

```
<table border="1">
<tr>
<td>row 1, cell 1</td>
<td>row 1, cell 2</td>
</tr>
<tr>
<td>row 2, cell 1</td>
<td></td>
</tr>
</table>
```

### Output

row 1, cell 1	row 1, cell 2
row 2, cell 1	



**Note:** that the borders around the empty table cell are missing (NB! Mozilla Firefox displays the border).

To avoid this, add a non-breaking space (&nbsp;) to empty data cells, to make the borders visible:

```
<table border="1">
<tr>
<td>row 1, cell 1</td>
<td>row 1, cell 2</td>
</tr>
<tr>
<td>row 2, cell 1</td>
<td>&nbsp;</td>
</tr>
</table>
```

#### Output

row 1, cell 1	row 1, cell 2
row 2, cell 1	

### Table Tags

Table 1 shows different table tags that can be used in HTML

Tag	Description
<table>	Defines a table
<th>	Defines a table header
<tr>	Defines a table row
<td>	Defines a table cell
<caption>	Defines a table caption
<colgroup>	Defines groups of table columns
<col>	Defines the attribute values for one or more columns in a table
<thead>	Defines a table head
<tbody>	Defines a table body
<tfoot>	Defines a table footer

Table 1 Table tags in HTML



Task: Illustrate the use of various table tags.

### 12.4 Linking Document

Once you have the ability to create HTML pages, you'll want to learn how to create links between them, so that you can start building a site. Links are the essence of HTML – they are what makes it unique.

What makes the web so effective is the ability to define links from one page to another, and to follow links at the click of a button. A single click can take you right across the world!

### Links

Links are defined with the `<a>` tag. Let's create a link to the page defined in the file "example1.html": This link to

```
<a href="Example1.html">My first homepage</a>.
```

The text between the `<a>` and the `</a>` is used as the caption for the link. It is common for the caption to be in blue underlined text. It is by the way a good idea to not have any blank spaces in the names of your HTML files, as this might create problems with some web servers. You can use an underscore, `_`, to separate words in your file names. To link to a page on another web site you need to give the full web address (the URL). For instance, to link to "Google" you need to write: A link to `<a href="http://www.google.com">Google</a>`. If you want the user's browser to open a new window for the linked page, (that way the user finds back to your page as soon as he or she closes the new window), use the attribute `target`:

```
A link to <a href="http://www.google.com target='_blank'">Google</a>.
```

## 12.5 Inserting inline images

`<Img>` causes an "inline image" to be inserted into the output. The image will be retrieved and rendered as if it were just another part of the text. Inline images can occur within headings, or paragraphs, almost anywhere in fact, except body (in other words, you can have a 'free floating' `<img>` tag – it must be contained within some other element.)

Like `<hr>`, this is an empty element. That is, there must be no end-tag. You will sometimes see `<hr>` used with an end-tag (e.g. as a container around a caption), but this is obsolete usage.

`<Img>` has 1 required attribute `src` as well as 3 optional attributes.

**src** The `src` attribute is used to specify the URL of the image (i.e. the address or filename the browser uses to retrieve the image file),

e.g. ``

``

**alt** `alt` is used to provide an text alternative to the image for readers whose browsers do not support graphics (or for visually impaired readers using alternative display devices). Although not required, the use of the `alt` attribute is nearly always appropriate and is strongly recommended. The only exception might be cases where the image is strictly decorative or of generic character. In this case the default text chosen by the browser (typically something like "[IMAGE]" may be adequate).

**align** `align` can take one of three values:

- top
- middle
- bottom

and is used to indicate how the browser should align the image with the adjacent text.

**bottom**: align the bottom of the image with the bottom of text

**middle**: align the middle of the image with the middle of text

**top**: align the top of the image with the top of text.

Images can also be retrieved using by means of a hypertext link using the `<a>` tag. The key difference between an inline image and an image retrieved with the `<a>` tag is that an inline image requires no action on the part of the reader; it is retrieved with the page just as if it were part of the text. With the `<a>` tag, the reader has to make a special action (e.g. clicking on a button) to retrieve the image.



**Task:** Give examples of the attributes used with `<img>` tag.

## 12.6 Creating Image Links

Image links are constructed as you might expect, by embedding an `<img>` tag inside of an anchor element `<a>`. Like HTML text links, image links require opening and closing anchor tags, but instead of placing text between these opening and closing tags, the developer needs to place an image tag – with a valid source attribute value of course.



**Example:**

```
<!DOCTYPE html>
<html>
<body>
<p>Create a link of an image: <a href="default.html">
</a></p>
<p>No border around the image, but still a link: <a href="default.html">
</a>
</p>
</body>
</html>
```

### Output

Create a link of an image:

No border around the image, but still a link:

## 12.7 IMG Attributes

`<IMG>` indicates that an image – such as a photograph, icon, animation, cartoon, or other graphic – is to be displayed at that location. The `<IMG>` tag should contain within it further parameters as part of the command: `SRC= "URL/graphic.gif or .jpg"`: contains the URL (Uniform Resource Locator or web address) and name of the graphic image file, such as `graphic.gif` or `graphic.jpg`. Most commonly, the photograph, icon, or other graphic is a “gif” (Graphics Interchange Format image) or a “jpg” (Joint Photographic Experts Group image), both of which are recognized by most browsers. Some browsers also will recognize a “bmp” (Bitmap image) and a “tif” or “tiff” (Tag Image File Format image). Usually, the location source of the graphic file is in an adjacent directory such as “graphics,” or it possibly might be in the same directory. Assuming the image is a `.jpg` image, if the graphic is in an adjacent “graphics” directory, the tag would read: `<IMG SRC= "graphics/ graphic.jpg">`. If the image is located within the same directory as the document, the tag would read simply: `<IMG SRC="graphic.jpg">`. If the location of the image is somewhere else on the web, the tag might read something like this: `<IMG SRC= "http://www.tedmontgomery.com/tutorial/graphics/graphic.gif">`.

`ALIGN="LEFT" | "RIGHT" | "TOP" | "TEXTTOP" | "MIDDLE" | "ABSMIDDLE" | "BASELINE" | "BOTTOM" | "ABSBOTTOM"`: places the graphic image at a specified position, in relation either to the page margins or to the text. (Some browsers will not recognize all of these parameters.) “LEFT” aligns the image with the left margin of the page and allows text to wrap around the right side of the image. “RIGHT” aligns the image with the right margin of the page and allows the text to wrap around the left side of the image. Note: The only way to center a graphic horizontally on a page is to use `<CENTER>&</ CENTER>` tags around the `<IMG SRC>` tag. However, centering a graphic in this manner will prevent text from being

wrapped around either side of it. Also, any ALIGN="RIGHT" or ALIGN="LEFT" parameter within the <IMG> tag will override the effect of the centering tags. "TOP" aligns the top of the image with the top of the tallest item in the line. "TEXTTOP" aligns the top of the image with the top of the tallest text in the line; usually, but not always, the same as the "TOP" parameter. "MIDDLE" aligns the middle of the image with the baseline of the current line. "ABSMIDDLE" aligns the middle of the image with the middle of the current line. "BASELINE" aligns the bottom of the image with the baseline of the current line. "BOTTOM" is the same as the "BASELINE" parameter. "ABSBOTTOM" aligns the bottom of the image with the bottom of the current line; usually, but not always, the same as the "BASELINE" or "BOTTOM" parameter.

WIDTH="W": defines the width "W" of the image in pixels.

HEIGHT="H": defines the height "H" of the image in pixels.

BORDER="B": creates a border around the image, with a uniform width of "B" in pixels. (In case the image is incorporated as a hyperlink, the unvisited, active, and visited colors of the border will be the same as that of the other text hyperlinks on the page.) The default setting is BORDER= "2" (that is, a border 2 pixels wide).

HSPACE="H": creates a space, with width "H" in pixels, between the image and any text immediately to the right and/or left of it. (HSPACE means "horizontal space.")

VSPACE="V": creates a space, with height "V" in pixels, between the image and any text immediately above and/or below it. (VSPACE means "vertical space.")

ALT="alternate description": supplies a description of the image, which will be displayed instead of the image on non-graphical browsers. On typical graphical browsers, this description will appear before the image has loaded and also when the arrow is placed anywhere on the image.

TITLE="title": same function as the ALT tag, which is not recognized by some browsers.

ISMAP: indicates a server-side image map.

USEMAP: indicates a client-side image map

## **12.8 HTML Lists**

Lists commonly are found in documents, including web pages. They are an easy and effective way to itemize such things as elements, components, or ingredients. Words or phrases which need to be set apart from the rest of the body of text can be emphasized with a "bullet" (a heavy dot used for calling attention to a particular section of text). An empty tag called a "list" tag is used to do this: <LI>: creates a bullet in front of text which is to be set apart for emphasis and causes all text after it to be indented, either until another list tag is detected or until the end of the list is reached. It is used to itemize elements of "unordered" and "ordered" lists.



**Notes:** A <BR> tag is not inserted at the end of an item beginning with a <LI> tag, as a line break automatically occurs at that point.

## **12.9 HTML List Classes**

The HTML List classes allow you to easily create lists within your HTML pages. These classes provide methods to get and set various attributes of the lists and the items within the lists. In particular, the parent class HTML List provides a method to produce a compact list that displays items in as small a vertical space as possible.

**Methods for HTML List include:**

Compact the list

Add and remove items from the list

Add and remove lists from the list (making it possible to nest lists)

**Methods for HTML List Item include:**

Get and set the contents of the item

Get and set the direction of the text interpretation

Get and set the language of the input element Use the subclasses of HTML List and HTML List Item to create your HTML lists:

OrderedList and OrderedListItem

UnorderedList and UnorderedListItem

### **OrderedList and OrderedListItem**

Use the `OrderedList` and `OrderedListItem` classes to create ordered lists in your HTML pages.

**Methods for `OrderedList` include:**

Get and set the starting number for the first item in the list

Get and set the type (or style) for the item numbers

Methods for `OrderedListItem` include:

Get and set the number for the item

Get and set the type (or style) for the item number

### **UnorderedList and UnorderedListItem**

Use the `UnorderedList` and `UnorderedListItem` classes to create unordered lists in your HTML pages.

**Methods for `UnorderedList` include:**

Get and set the type (or style) for the items

**Methods for `UnorderedListItem` include:**

Get and set the type (or style) for the item

## **12.10 HTML List Types**

HTML provides three different types of lists to choose from when building a page, including unordered, ordered, and definition lists. Unordered lists are for lists of items where order isn't of important. While ordered lists place strong importance on the order of items. In the case where there is a list of terms and descriptions, perhaps for a glossary, definition lists are available.

### **Unordered List**

Unordered lists are purely a list of related items, in which their order does not matter nor do they have a numbered or alphabetical list element. Creating an unordered list in HTML is accomplished using the unordered list, `ul`, block level element. Each list item within an unordered list is individually marked up using the list item, `li`, block level element. By default most browsers represent each list item with a solid dot.

#### **Example**

```
<ul>
<li>Tamilnadu</li>
<li>Uttar Pradesh</li>
<li>West-Bengal</li>
</ul>
```

### **Ordered List**

The ordered list element, ol, works just like the unordered list element, including how each individual list item is created. The main difference between an ordered list and an unordered list is that with an ordered list the order of which items are represented is important. Instead of showing a dot as the default list item element, an ordered list uses numbers. Using CSS, these numbers can then be changed to letters, Roman numerals, and so on.

#### Example

```
<ol type="1">
<li>Tamilnadu</li>
<li>Uttar Pradesh</li>
<li>West-Bengal</li>
</ol>
```

#### Description List

Creating a definition list in HTML is accomplished using the dl element. Instead of using the li element to mark up list items, the definition list actually requires two elements: the definition term element, dt, and the definition description element, dd. A definition list may contain numerous terms and descriptions, one after the other. Additionally, a definition list may have multiple terms per description as well as multiple descriptions per term. A single term may have multiple meanings and warrant multiple definitions. In comparison, a single description may be suitable for multiple terms.

```
<dl>
<dt>study</dt>
<dd>the devotion of time and attention to acquiring knowledge on an academic subject, esp. by means of books</dd>
<dt>design</dt>
<dd>a plan or drawing produced to show the look and function or workings of a building, garment, or other object before it is built or made</dd>
<dd>purpose, planning, or intention that exists or is thought to exist behind an action, fact, or material object</dd>
<dt>business</dt>
<dt>work</dt>
<dd>a person's regular occupation, profession, or trade</dd>
</dl>
```

### **12.11 HTML Block and Inline Elements**

Every HTML element has a default display value, depending on what type of element it is.

There are two display values: block and inline.

#### Block-level Elements

A block-level element always starts on a new line, and the browsers automatically add some space (a margin) before and after the element.

A block-level element always takes up the full width available (stretches out to the left and right as far as it can).

Two commonly used block elements are: <p> and <div>.

- The <p> element defines a paragraph in an HTML document.
- The <div> element defines a division or a section in an HTML document.

The <p> element is a block-level element.

The <div> element is a block-level element.

### Inline Elements

An inline element does not start on a new line. An inline element only takes up as much width as necessary.

This is a <span> element inside a paragraph

### The <div> Element

The <div> element is often used as a container for other HTML elements. The <div> element has no required attributes, but style, class and id are common.

```
<!DOCTYPE html>
<html>
<body>
<div style="background-color:black;color:white;padding:20px;">
<h2>Demo</h2>
<p>This is a demo for block element in HTML</p>
</div>
</body>
</html>
```

#### Output

### Demo

This is a demo for block element in HTML

### The <span> Element

The <span> element is an inline container used to mark up a part of a text, or a part of a document.

The <span> element has no required attributes, but style, class and id are common.

When used together with CSS, the <span> element can be used to style parts of the text:



#### Example

```
<!DOCTYPE html>
<html>
<body>
<h1>The span element</h1>
<p>The sky is <span style="color:blue;font-weight:bold">blue</span> in color and the trees are
<span style="color:darkolivegreen;font-weight:bold">dark green</span> in color. </p>
</body>
```

</html>

**Output**

## The span element

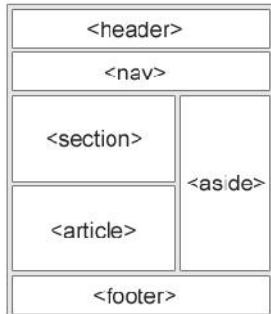
The sky is **blue** in color and the trees are **dark green** in color.

### **12.12 HTML Layout Elements and Techniques**

Websites often display content in multiple columns (like a magazine or a newspaper).

#### **HTML Layout Elements**

HTML has several semantic elements that define the different parts of a web page:



- **<header>** - Defines a header for a document or a section
- **<nav>** - Defines a set of navigation links
- **<section>** - Defines a section in a document
- **<article>** - Defines an independent, self-contained content
- **<aside>** - Defines content aside from the content (like a sidebar)
- **<footer>** - Defines a footer for a document or a section
- **<details>** - Defines additional details that the user can open and close on demand
- **<summary>** - Defines a heading for the **<details>** element

#### **Page Layout Information:**

**Header:** The part of a front end which is used at the top of the page. **<header>** tag is used to add header section in web pages.

**Navigation bar:** The navigation bar is same as menu list. It is used to display the content information using hyperlink.

**Index / Sidebar:** It holds additional information or advertisements and is not always necessary to be added into the page.

**Content Section:** The content section is the main part where content is displayed.

**Footer:** The footer section contains the contact information and other query related to web pages. The footer section always put on the bottom of the web pages. The **<footer>** tag is used to set the footer in web pages.



#### **Example**

```

<!DOCTYPE html>
<html lang="en">
<head>
<title>CSS Template</title>
<meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<style>
* {
  box-sizing: border-box;

```

```
}
```

```
body {  
    font-family: Arial, Helvetica, sans-serif;  
}  
  
/* Style the header */  
header {  
    background-color: #666;  
    padding: 30px;  
    text-align: center;  
    font-size: 35px;  
    color: white;  
}  
  
/* Create two columns/boxes that floats next to each other */  
nav {  
    float: left;  
    width: 30%;  
    height: 300px; /* only for demonstration, should be removed */  
    background: #ccc;  
    padding: 20px;  
}  
  
/* Style the list inside the menu */  
nav ul {  
    list-style-type: none;  
    padding: 0;  
}  
  
article {  
    float: left;  
    padding: 20px;  
    width: 70%;  
    background-color: #f1f1f1;  
    height: 300px; /* only for demonstration, should be removed */  
}  
  
/* Clear floats after the columns */  
section::after {
```

```
content: "";
display: table;
clear: both;
}

/* Style the footer */
footer {
background-color: #777;
padding: 10px;
text-align: center;
color: white;
}

/* Responsive layout - makes the two columns/boxes stack on top of each other instead of next to
each other, on small screens */
@media (max-width: 600px) {
nav, article {
width: 100%;
height: auto;
}
}

</style>
</head>
<body>

<h2>Layout in HTML</h2>


In this example, we have created a header, two columns/boxes and a footer. On smaller screens, the columns will stack on top of each other.</p>
<header>
<h2>HTML</h2>
</header>

<section>
<nav>
<ul>
<li><a href="#">Home</a></li>
<li><a href="#">Gallery</a></li>
<li><a href="#">Contact us</a></li>
</ul>
</nav>


```

```

<article>
<h1>Layout</h1>
<p>Page layout is the part of graphic design that deals with the arrangement of visual elements on a page. Page layout is used to make the web pages look better. It establishes the overall appearance, relative importance, and relationships between the graphic elements to achieve a smooth flow of information and eye movement for maximum effectiveness or impact</p>
</article>
</section>

<footer>
<p>Footer</p>
</footer>

</body>
</html>

```

#### **Layout In HTML**

In this example, we have created a header, two column sections and a footer. On smaller screens, the columns will stack on top of each other.



### **12.13 HTML Forms**

HTML stands for HyperText Markup Language. It is used to design web pages using a markup language. It is a combination of Hypertext and Markup language. HTML uses predefined tags and elements that tell the browser how to properly display the content on the screen. And form is one of them. So, in this article, we will learn what is exactly HTML form what are the elements of forms and how can we use HTML form in our webpage.

#### **What is HTML <form>?**

<form> is a HTML element to collect input data with containing interactive controls. It provides facilities to input text, number, values, email, password, and control fields such as checkboxes, radio buttons, submits buttons, etc. or in other words, form is a container that contains input elements, like text, email, number, radio buttons, checkboxes, submit buttons, etc. Forms are generally used when you want to collect data from the user. For example, a user wants to buy a bag online, so he/she has to first enter their shipping address in the address form and then add their payment detail in the payment form to place an order.

Forms are created by placing input fields within paragraphs, preformatted text, lists and tables. This gives considerable flexibility in designing the layout of forms.

#### **Syntax:**

```
<form>
<!--form elements-->
</form>
```

#### Form elements

These are the following HTML <form> elements:

- <label>: It defines label for <form> elements.
- <input>: It is used to get input data from the form in various type such as text, password, email, etc by changing its type.
- <button>: It defines a clickable button to control other elements or execute a functionality.
- <select>: It is used to create a drop-down list.
- <textarea>: It is used to get input long text content.
- <fieldset>: It is used to draw a box around other form elements and group the related data.
- <legend>: It defines caption for fieldset elements.
- <datalist>: It is used to specify pre-defined list options for input controls.
- <output>: It displays the output of performed calculations.
- <option>: It is used to define option in drop-down list.
- <optgroup>: It is used to define group related options in a drop down list.

### Textbox in HTML Form

In an HTML form, we use <input> tag by assigning type attribute value to text to input single line input. To define type attribute see the below syntax.

#### Syntax:

```
<input type="text" />
```

Or shorthand for "text" type:

```
<input />
```

### Password in an HTML Form

We can change type value text to password to get the input password



#### Example

```
<!DOCTYPE html>
<html>
<head>
<title>Page Title</title>
</head>
<body>
<h2>Welcome To HTML Forms</h2>
<form>
<p>
<label>Username : <input type="text" /></label>
</p>
<p>
<label>Password : <input type="password" /></label>
</p>
<p>
<button type="submit">Submit</button>
```

```

</p>
</form>
</body>
</html>

```

## Welcome To HTML Forms

Username :

Password :

### Radio Button in an HTML Form

To create a radio button, we use the `<input>` tag following by radio type to provide users to choose a limited number of choices.

Syntax:

```
<input type="radio" name="radio_button_name" value="radio_button_value" />
```

**Note:** The radio button must have shared the same name to be treated as a group.

**Note:** The value attribute defines the unique value associated with each radio button. The value is not shown to the user, but is the value that is sent to the server on “submit” to identify which radio button that was selected.



Example:

```

<!DOCTYPE html>
<html>
<head>
<title>Page Title</title>
</head>
<body>
<h2>Select your gender</h2>
<form>
    <label>Male<input type="radio" name="gender" value="male" /></label>
    <label>Female<input type="radio" name="gender" value="female" /></label>
</form>
</body>
</html>

```

**Select your gender**Male  Female 

In this example, we will create a radio button to choose your gender.

**Checkbox in an HTML Form**

To create a checkbox in an HTML form, we use the <input> tag following by the input type checkbox. It is a square box to ticked to activate this. It used to choose more options at a time.

**Syntax:**

```
<input type="checkbox" name="select_box_name" value="select_box_value" />
```

Note: the "name" and "value" attributes are used to send the checkbox data to the server.

**Example:**

In this example, we use checkboxes to select language.

```
<!DOCTYPE html>
<html>
<head>
<title>Page Title</title>
</head>
<body>
<h2>Choose Language</h2>
<form>
<ul style="list-style-type:none;">
<li><input type="checkbox" name="language" value="hindi" />Hindi</li>
<li><input type="checkbox" name="language" value="english" />English</li>
<li><input type="checkbox" name="language" value="sanskrit" />Sanskrit</li>
</ul>
</form>
</body>
</html>
```

**Choose Language**

- Hindi
- English
- Sanskrit

### Combobox in an HTML Form

Combobox is used to create a drop-down menu in your form which contains multiple options. So, to create an Combobox in an HTML form, we use the `<select>` tag with `<option>` tag. It is also known as a drop-down menu.

Syntax:

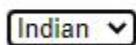
```
<select name="select_box_name">
<option value="value1">option1</option>
<option value="value2">option2</option>
<option value="value3">option3</option>
</select>
```



**Example:**

```
<!DOCTYPE html>
<html>
<head>
<title>Page Title</title>
</head>
<body>
<h2>Select Your Nationality</h2>
<form>
<select name="language">
    <option value="indian">Indian</option>
    <option value="nepali">Nepali</option>
    <option value="others">Others</option>
</select>
</form>
</body>
</html>
```

## Select Your Nationality



### Submit button in an HTML Form

In the HTML form, submit button is used to submit the details of the form to the form handler. A form handler is a file on the server with a script that is used to process input data.

Syntax:

```
<button type="submit">submit</button>
<!DOCTYPE html>
<html>
<head>
<title>Page Title</title>
```

```

</head>
<body>
<h2>Welcome To HTML forms</h2>
<form>
<p>
    <label>Username: <input type="text" /></label>
<p>
    <label>Password: <input type="password" /></label>
</p>
<p>
    <button type="submit">submit</button>
</p>
</form>
</body>
</html>

```

## Welcome To HTML forms

Username:

Password:

### TextArea in an HTML Form

In the HTML form, a text area is used to add comments or reviews, or addresses in the form. Or in other words, the text area is a multi-line text input control. It contains an unlimited number of characters, and the text renders in a fixed-width font and the size of the text area is given by the `<rows>` and `<cols>` attributes. To create a text area in the form use the `<textarea>` tag.

Syntax:

```

<textarea name="textarea_name">content</textarea>
<!DOCTYPE html>
<html>
<head>
<title>Page Title</title>
</head>
<body>
<h2>Welcome To HTML Forms</h2>
<form>
    <textarea    name="welcomeMessage"    rows="3"    cols="40">Kindly    give    your
feedback</textarea>
</form>

```

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

## Welcome To HTML Forms

Kindly give your feedback

### **12.14 Iframes in HTML**

HTML Iframe is used to display a nested webpage (a webpage within a webpage). The HTML <iframe> tag defines an inline frame, hence it is also called as an Inline frame.

An HTML iframe embeds another document within the current HTML document in the rectangular region.

The webpage content and iframe contents can interact with each other using JavaScript.

#### **Iframe Syntax**

An HTML iframe is defined with the <iframe> tag:

```
<iframesrc="URL"></iframe>
```

Here, "src" attribute specifies the web address (URL) of the inline frame page.

#### **Set Width and Height of iframe**

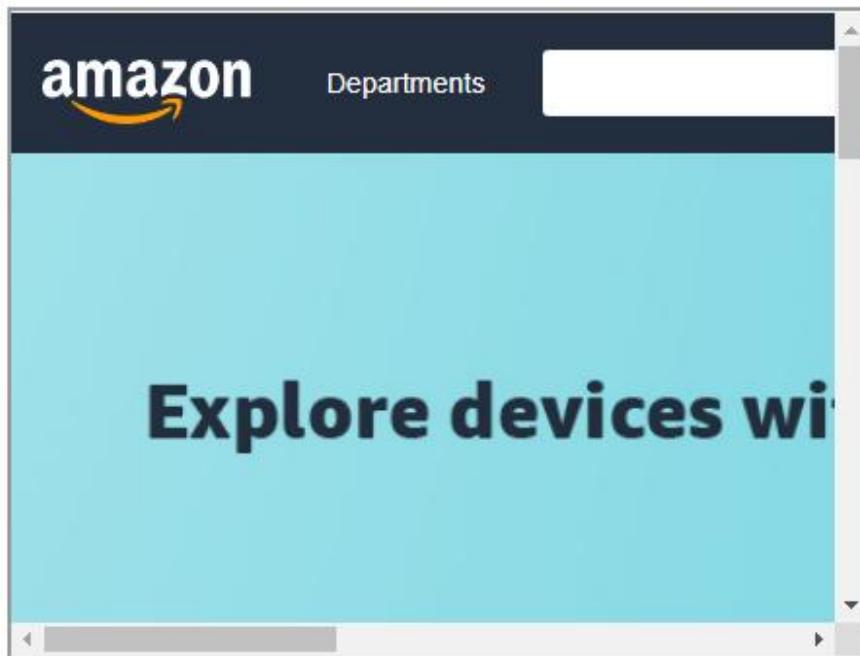
You can set the width and height of iframe by using "width" and "height" attributes. By default, the attributes values are specified in pixels but you can also set them in percent. i.e. 50%, 60% etc.

#### **Example: (Pixels)**

```
<!DOCTYPE html>  
<html>  
<body>  
<h2>HTML Iframes example</h2>  
<p>Use the height and width attributes to specify the size of the iframe:</p>  
<iframesrc="https://www.amazon.com/" height="300" width="400"></iframe>  
</body>  
</html>
```

## HTML Iframes example

Use the height and width attributes to specify the size of the iframe:



You can also change the size, color, style of the iframe's border.

### Embed YouTube video using iframe

You can also add a YouTube video on your webpage using the <iframe> tag. The attached video will be played at your webpage and you can also set height, width, autoplay, and many more properties for the video.

Following are some steps to add YouTube video on your webpage:

- Goto YouTube video which you want to embed.
- Click on SHARE ➔ under the video.
- Click on Embed <> option.
- Copy HTML code.
- Paste the code in your HTML file
- Change height, width, and other properties (as per requirement).



### Example

```
<!DOCTYPE html>
<html>
<head>

</head>
<body style="background-color: #f0f8ff">
    <h3>Play videos using iframe</h3>
```

Unit 12: Java Script- Browser Object Model

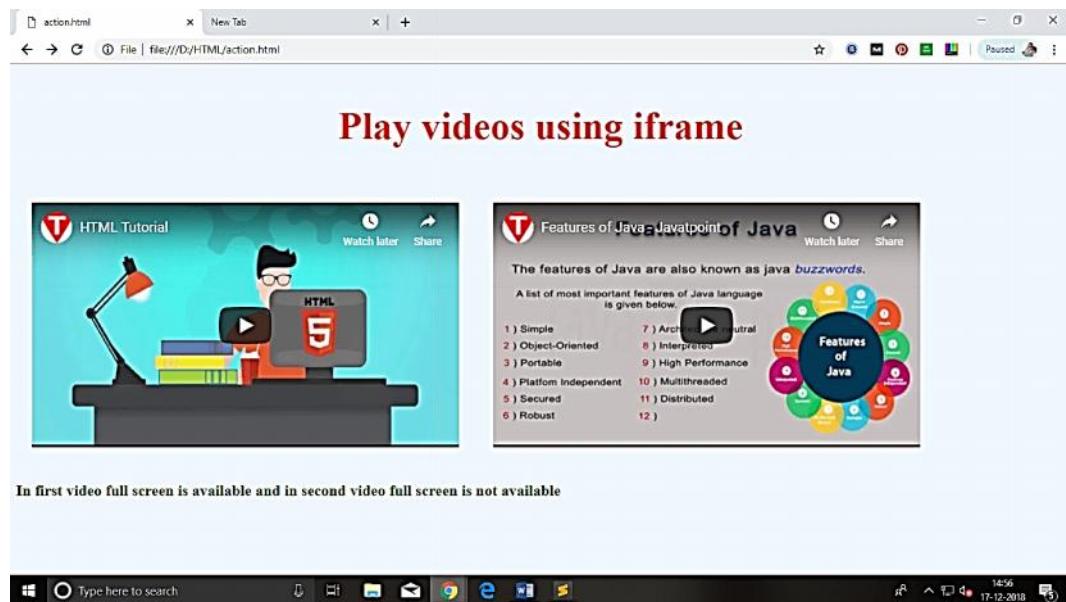
```
<iframe width="550" height="315"
src="https://www.youtube.com/embed/JHq3pL4cdy4" frameborder="0" allow="accelerometer;
autoplay; encrypted-media; gyroscope; picture-in-picture" allowfullscreen
style="padding:20px;"></iframe>
```

```
<iframe width="550" height="315" src="https://www.youtube.com/embed/O5hShUO6wxs"
frameborder="0" allow="accelerometer; autoplay; encrypted-media; gyroscope; picture-in-picture"
style="padding:20px;"></iframe>
```

<p>In first video full screen is available and in second video full screen is not available</p>

</body>

</html>

Summary

- Lists commonly are found in documents, including web pages.
- The HTML List classes allow you to easily create lists within your HTML pages.
- HTML List provides a method to produce a compact list that displays items in as small a vertical space as possible.
- HTML provides three different types of lists to choose from when building a page, including unordered, ordered, and definition lists.
- Ordered lists place strong importance on the order of items.
- Unordered lists are purely a list of related items, in which their order does not matter nor do they have a numbered or alphabetical list element.
- A definition list may contain numerous terms and descriptions, one after the other.
- One of the best ways to create an impact with your web page is to add some graphics.
- Tables are defined with the <table> tag.
- Constructing an HTML table consists of describing the table between the beginning table tag, <TABLE>, and the ending table tag, </TABLE>.
- Links are the essence of HTML — they are what makes it unique. Links are defined with the <a> tag.

- If you do not specify a border attribute the table will be displayed without any borders.
- <Img> causes an “inline image” to be inserted into the output.
- Image maps are images with clickable areas (sometimes referred to as “hotspots”) that usually link to another page.
- Email links are done much the same as links to other pages, using the <a href> tag.
- The key difference between an inline image and an image retrieved with the <a> tag is that an inline image requires no action on the part of the reader

## **Keywords**

- <IMG>: <IMG> indicates that an image—such as a photograph, icon, animation, cartoon, or other graphic—is to be displayed at that location.
- DL: Definition lists, created using the DL element, generally consist of a series of term/definition Pairs. HTML List classes: It allows you to easily create lists within your
- HTML pages. Inline image: An image which is displayed on a web browser is referred to as an “inline image”.
- alt: alt is used to provide an text alternative to the image for readers whose browsers do not support graphics.
- Cell tags: <TD></TD> Image links: Image links are constructed as you might expect, by embedding an <img> tag inside of an anchor element <a>.
- Links: Links are the essence of HTML — they are what makes it unique
- Row tags: <TR></TR>
- List tag: An empty tag called a “list” tag is used to do itemize elements of “unordered” and “ordered” lists.
- OL: An ordered list, created using the OL element, should contain information where order should be emphasized.
- Ordered List Item: It allows you to override the numbering and type for a specific item in the list. Unordered lists: Unordered lists are for lists of items where order isn’t of important.

## **Self Assessment**

1. Which HTML tag is used to define a table?
  - A. <tb>
  - B. <tl>
  - C. <table>
  - D. <tab>
  
2. With the help of which tag, is a row defined in HTML?
  - A. <row>
  - B. <table-row>
  - C. <tablerow>
  - D. <tr>

3. Choose the correct option.

- A. <th> is used for defining the heading of a table.
- B. By default, contents written between <th> and </th> are bold and centered.
- C. Both A and B
- D. None Of these

4. Fill in the blanks with the help of options given below in order to get the following table when the below code is executed.

```
<!DOCTYPE html>
<html>
<body>
<table border="2">
<tr>
<th>Name</th>
<th>Phone no</th>
</tr>
<tr>
<td _____>John</td>
<td>9898989898</td>
</tr>
<tr>
<td>9876543210</td>
</tr>
</body>
</html>
```

Name	Phone no
John	9898989898
	9876543210

- A. colspan= "1"
- B. colspan= "2"
- C. rowspan= "1"
- D. rowspan= "2"

5. Which one of the following tags is used to add caption to a table?

- A. <table-caption>
- B. <tcaption>
- C. <caption>
- D. <tc>

6. Each cell of the table can be represented by using \_\_\_\_\_

- A. <tr>
- B. <td>
- C. <th>
- D. <thead>

7. The \_\_\_\_\_ tag defines an image in an HTML page.

- A <image>
- B <pic>

C <imge>

D <img>

8. Which of the following pair of attribute is required for img tag ?

- A.src and a
- B. img and src
- C. img and alt
- D. src and alt

9. If the image cannot be displayed then \_\_\_\_\_ specifies an alternate text for an image.

- A. text attribute
- B. value attribute
- C. alt attribute
- D. caption attribute

10. Alt Attribute is more useful in the situation where user have \_\_\_\_\_.

- A. High Speed Internet Connection
- B. Broadband Connection
- C. None of these
- D. Slow Internet Connection

11. Which one of the following is a type of lists that HTML supports?

- A. Ordered lists.
- B. Unordered lists.
- C. Description lists.
- D. All of the above

12. By which tag, an unordered list is represented?

- A. <u>
- B. <I>
- C. <ul>
- D. <ol>

13. By which tag, an ordered list is represented?

- A. <u>
- B. <I>
- C. <ul>
- D. <ol>

14. A HTML code is given below

Fill in the blanks with appropriate option to get the output as below:

```
<!DOCTYPE html>
<html>
<body>
<dl>
    _____
Mathematics
    _____
Calculus
</dl>
</body>
</html>
```

- A <dd>,</dd>,<dt>,</dt>  
 B. <dt>,</dt>,<dd>,</dd>  
 C. <li>,</li>,<dd>,</dd>  
 D. <dt>,</dt>,<li>,</li>  
  
 15. What is the default item marker in unordered lists of HTML?  
 A. Circle  
 B. Marker  
 C. disc  
 D. None of the above

### **Answers for Self Assessment**

- |       |       |       |       |       |
|-------|-------|-------|-------|-------|
| 1. C  | 2. D  | 3. C  | 4. D  | 5. C  |
| 6. B  | 7. C  | 8. B  | 9. C  | 10. D |
| 11. D | 12. C | 13. D | 14. B | 15. C |

### **Review Questions**

1. On what information we should emphasize while preparing an order list? '
2. Discuss the methods for HTML List.
3. Discuss the methods to create ordered lists, unordered lists, and nested lists.
4. Discuss IMG Attributes.
5. Explain with examples about graphic image alignment parameters in the HTML.
6. Explain the three different types of HTML lists.
7. Explain how HTML Tables are created?
8. Define and Explain the steps involved in creating HTML Tables.
9. Discuss in brief about Linking Document.
10. What are the three attributes that can be specified with the <BODY> tag? Explain each of them.

11. Define and explain Hyperlinks and their types.
12. . Justify the use of link list creation in html document.
13. Explain how to display a table with borders.
14. . Discuss how to link to a page on another web site. Illustrate with example.
15. Explain the steps used to move to a specific location on a Web page



## **Further Readings**

Hall, 2009, Core Web Programming, 2/E, Pearson Education India. Jon Duckett, 2011, Beginning Web Programming with HTML, XHTML and CSS,

John Wiley & Sons. Robert F. Breedlove, 1996, Web Programming Unleashed, Sams.net.

Tim Downey, 2012, Guide to Web Development with Java: Understanding Website Creation, Springer.



## **Web Links**

<http://www.temple.edu/cs/web/tables.html>  
[http://www.w3schools.com/html/tryit.asp?filename=tryhtml\\_imglink](http://www.w3schools.com/html/tryit.asp?filename=tryhtml_imglink)  
<http://www.tizag.com/htmlT/htmlimagelinks.php>  
[http://www.quackit.com/html/tutorial/html\\_image\\_maps.cfm](http://www.quackit.com/html/tutorial/html_image_maps.cfm)  
<http://www.echoecho.com/htmllinks11.htm>  
<http://interestingwebs.blogspot.in/2008/12/how-to-create-scrollable-linklist.html>  
<http://www-sul.stanford.edu/tools/tutorials/html2.0/img.html>

## Unit 13: JavaScript-validation and Menu Builder

### Objectives

- Understand the concept of Form validation
- Discuss about different validations used in form
- Implementation of email validations
- Building of email menu builder

### Introduction

Form validation normally used to occur at the server, after the client had entered all the necessary data and then pressed the Submit button. If the data entered by a client was incorrect or was simply missing, the server would have to send all the data back to the client and request that the form be resubmitted with correct information. This was really a lengthy process which used to put a lot of burden on the server.

JavaScript provides a way to validate form's data on the client's computer before sending it to the web server. Form validation generally performs two functions.

Basic Validation – First of all, the form must be checked to make sure all the mandatory fields are filled in. It would require just a loop through each field in the form and check for data.

Data Format Validation – Secondly, the data that is entered must be checked for correct form and value. Your code must include appropriate logic to test correctness of data.

### Example

We will take an example to understand the process of validation. Here is a simple form in html format

```
<html>
<head>
<title>Form Validation</title>
<script type = "text/javascript">
<!--
    // Form validation code will come here.
    //-->
</script>
</head>

<body>
<form action = "/cgi-bin/test.cgi" name = "myForm" onsubmit = "return(validate());">
<table cellspacing = "2" cellpadding = "2" border = "1">

<tr>
<tdalign = "right">Name</td>
<td><input type = "text" name = "Name" /></td>
</tr>
```

```
<tr>
<tdalign = "right">EMail</td>
<td><input type = "text" name = "EMail" /></td>
</tr>

<tr>
<tdalign = "right">Zip Code</td>
<td><input type = "text" name = "Zip" /></td>
</tr>

<tr>
<tdalign = "right">Country</td>
<td>
<select name = "Country">
<option value = "-1" selected>[choose yours]</option>
<option value = "1">USA</option>
<option value = "2">UK</option>
<option value = "3">INDIA</option>
</select>
</td>
</tr>

<tr>
<tdalign = "right"></td>
<td><input type = "submit" value = "Submit" /></td>
</tr>

</table>
</form>
</body>
</html>
```

## Output

The form consists of five input fields arranged vertically. From top to bottom: 1. A text input field labeled 'Name'. 2. A text input field labeled 'EMail'. 3. A text input field labeled 'Zip Code'. 4. A dropdown menu labeled 'Country' containing the placeholder '[choose yours]'. 5. A submit button labeled 'Submit'.

### **13.1 Basic Form Validation**

First let us see how to do a basic form validation. In the above form, we are calling validate() to validate data when onsubmit event is occurring. The following code shows the implementation of this validate() function.

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

// Form validation code will come here.

function validate() {

if( document.myForm.Name.value == "" ) {
alert( "Please provide your name!" );
document.myForm.Name.focus() ;
return false;
}

if( document.myForm.EMail.value == "" ) {
alert( "Please provide your Email!" );
document.myForm.EMail.focus() ;
return false;
}

if( document.myForm.Zip.value == "" || isNaN( document.myForm.Zip.value ) ||
document.myForm.Zip.value.length != 5 ) {

alert( "Please provide a zip in the format #####." );
document.myForm.Zip.focus() ;
return false;
}

if( document.myForm.Country.value == "-1" ) {
alert( "Please provide your country!" );
return false;
}

return( true );
}
-->
```

```
//-->
</script>
```

## **13.2 Data Format Validation**

Now we will see how we can validate our entered form data before submitting it to the web server.

The following example shows how to validate an entered email address. An email address must contain at least a '@' sign and a dot (.). Also, the '@' must not be the first character of the email address, and the last dot must at least be one character after the '@' sign.

### **Example**

Try the following code for email validation.

```
<script type = "text/javascript">
<!--
function validateEmail() {
    var emailID = document.myForm.EMail.value;
    atpos = emailID.indexOf("@");
    dotpos = emailID.lastIndexOf(".");
    if (atpos< 1 || ( dotpos - atpos< 2 )) {
        alert("Please enter correct email ID")
        document.myForm.EMail.focus();
        return false;
    }
    return( true );
}
//-->
</script>
```

## **13.3 JavaScript Retype Password Validation**

```
<script type="text/javascript">
function matchpass(){
var firstpassword=document.f1.password.value;
var secondpassword=document.f1.password2.value;

if(firstpassword==secondpassword){
return true;
}
else{
alert("password must be same!");
return false;
}
}
```

```
</script>

<form name="f1" action="register.jsp" onsubmit="return matchpass()">
    Password:<input type="password" name="password" /><br/>
    Re-enter Password:<input type="password" name="password2"/><br/>
    <input type="submit">
</form>
```

### **13.4 JavaScript Number Validation**

Let's validate the textfield for numeric value only. Here, we are using isNaN() function.

```
<script>
function validate(){
var num=document.myform.num.value;
if (isNaN(num)){
document.getElementById("numloc").innerHTML="Enter Numeric value only";
return false;
} else{
    return true;
}
}

</script>
<form name="myform" onsubmit="return validate()">
    Number: <input type="text" name="num"><span id="numloc"></span><br/>
    <input type="submit" value="submit">
</form>
```

The image shows a simple HTML form. It contains a label "Number:" followed by a text input field. Below the input field is a submit button. The entire form is enclosed in a light gray border.

### **13.5 JavaScript email validation**

We can validate the email by the help of JavaScript.

There are many criteria that need to be follow to validate the email id such as:

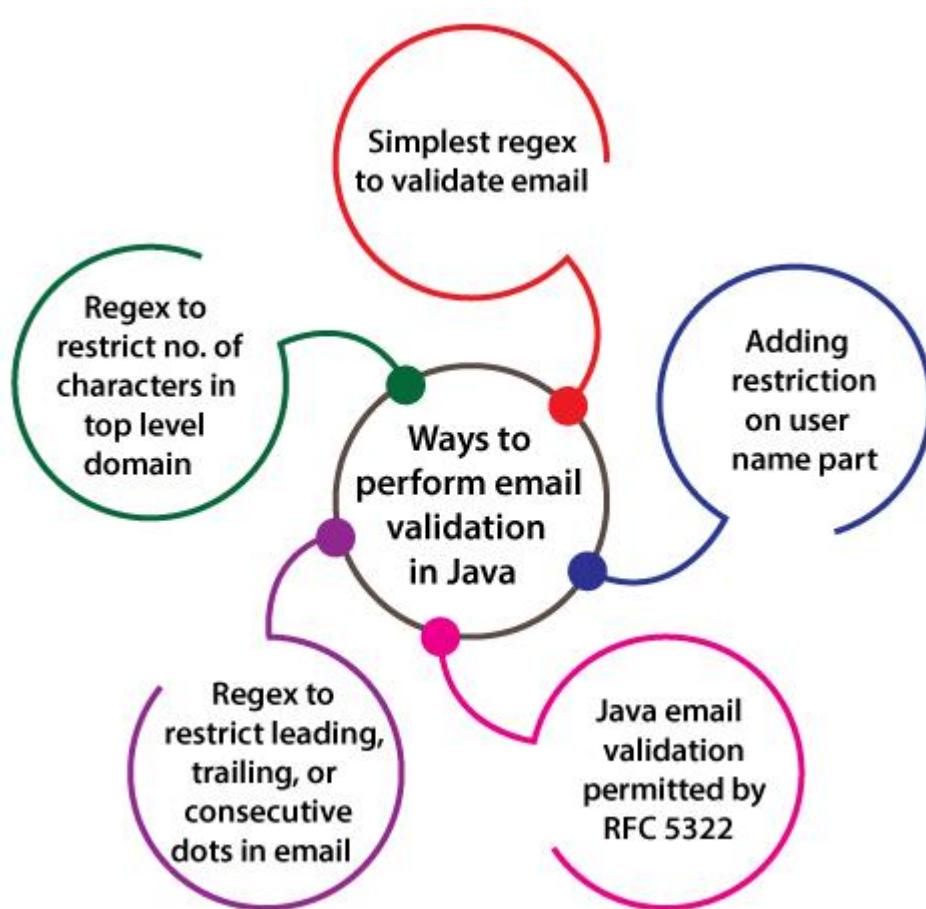
- email id must contain the @ and . character
- There must be at least one character before and after the @.
- There must be at least two characters after . (dot).

In designing forms, email plays an important role. The email can be of our username or login id. An email has its own structure, and before using it, we need to validate it. In Java, email validation is performed by using the regular expression.

Email validation is required in any of the application that looks for email addresses as required information at the registration stage.

There are five ways through which we can perform email validation using a regular expression.

1. Simplest regex to validate email.
2. Adding restriction on username part.
3. E-mail validation permitted by RFC 5322.
4. Regex to restrict leading, trailing, or consecutive dots in emails
5. Regex to restrict no. of characters in the top-level domain.
6. Let's see the simple example to validate the email field.



email using JavaScript :

An email is a string (a subset of ASCII characters) separated into two parts by @ symbol. a "personal\_info" and a domain, that is personal\_info@domain. The length of the personal\_info part may be up to 64 characters long and domain name may be up to 253 characters.

The personal\_info part contains the following ASCII characters.

- Uppercase (A-Z) and lowercase (a-z) English letters.
- Digits (0-9).
- Characters ! # \$ % & ' \* + - / = ? ^ \_ ` { | } ~
- Character .( period, dot or fullstop) provided that it is not the first or last character and it will not come one after the other.

Validating an email is a very important point while validating an HTML form. In this page we have discussed how to validate an

The domain name [for example com, org, net, in, us, info] part contains letters, digits, hyphens, and dots.

**Example of valid email id**

- mysite@ourearth.com
- my.ownsite@ourearth.org
- mysite@you.me.net

**Example of invalid email id**

- mysite.ourearth.com [@ is not present]
- mysite@.com.my [ tld (Top Level domain) can not start with dot "."]
- @you.me.net [ No character before @ ]
- mysite123@gmail.b [ ".b" is not a valid tld ]
- mysite@.org.org [ tld can not start with dot "."]
- .mysite@mysite.org [ an email should not start with "."]
- mysite()@gmail.com [ here the regular expression only allows character, digit, underscore, and dash ]
- mysite..1234@yahoo.com [double dots are not allowed]

**JavaScript code to validate an email id**

```
function ValidateEmail(mail)
{
if (/^[\w+([\.-]?\w+)*@\w+([\.-]?\w+)*(\.\w{2,3})+$/.test(myForm.emailAddr.value))
{
    return (true)
}
alert("You have entered an invalid email address!")
return (false)
}
```

Let apply the above JavaScript function in an HTML form.

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>JavaScript form validation - checking email</title>
<link rel='stylesheet' href='form-style.css' type='text/css' />
</head>
<body onload='document.form1.text1.focus()'>
<div class="mail">
<h2>Input an email and Submit</h2>
<form name="form1" action="#">
```

```
<ul>
<li><input type='text' name='text1' /></li>
<li>&nbsp;</li>
<li class="submit"><input type="submit" name="submit" value="Submit"
    onclick="ValidateEmail(document.form1.text1)"/></li>
<li>&nbsp;</li>
</ul>
</form>
</div>
<script src="email-validation.js"></script>
</body>
</html>
```

### JavaScript code

```
function ValidateEmail(inputText)
{
    var mailformat = /^[^\w+([.-]?\w+)*@\w+([.-]?\w+)*(\.\w{2,3})+$/;
    if(inputText.value.match(mailformat))
    {
        alert("Valid email address!");
        document.form1.text1.focus();
        return true;
    }
    else
    {
        alert("You have entered an invalid email address!");
        document.form1.text1.focus();
        return false;
    }
}
```

### CSS Code

```
li {list-style-type: none;
font-size: 16pt;
}
.mail {
margin: auto;
padding-top: 10px;
padding-bottom: 10px;
width: 400px;
background : #D8F1F8;
```

```
border: 1px solid silver;  
}  
.mail h2 {  
margin-left: 38px;  
}  
input {  
font-size: 20pt;  
}  
input:focus, textarea:focus{  
background-color: lightyellow;  
}  
input submit {  
font-size: 12pt;  
}  
.rq {  
color: #FF0000;  
font-size: 10pt;  
}
```

## **Input an email and Submit**

**Submit**

### **13.6 Summary**

- Form validation normally used to occur at the server, after the client had entered all the necessary data and then pressed the Submit button
- If an HTML document contains more than one forms, they can be accessed as either by `document.form_name` where `form_name` is the value of the name attribute of the form element or by `document.forms[i]` where `i` is `0, 1, 2, 3, ...` and `document.forms[0]` refers to the first form of the document, `document.forms[1]` refers to the second form of the document and so on.
- Elements of a form can be accessed by `document.form_name.form_element` where `form_name` is the value of the name attribute of the form element, `form_element` is the value of the name attribute of the form's element.

- JavaScript is a lightweight, interpreted programming language. It is designed for creating network-centric applications. It is complimentary to and integrated with Java.
- JavaScript is very easy to implement because it is integrated with HTML. It is open and cross-platform

### **13.7 Keywords**

**Basic Validation** – First of all, the form must be checked to make sure all the mandatory fields are filled in. It would require just a loop through each field in the form and check for data.

**Data Format Validation** – Secondly, the data that is entered must be checked for correct form and value. Your code must include appropriate logic to test correctness of data.

**Client side validation** - This is really important to verify any user input before submitting it to the server and Javascript plays an important role in validating those inputs at front-end itself.

**Manipulating HTML Pages** - Javascript helps in manipulating HTML page on the fly. This helps in adding and deleting any HTML tag very easily using javascript and modify your HTML to change its look and feel based on different devices and requirements.

**User Notifications** - You can use Javascript to raise dynamic pop-ups on the webpages to give different types of notifications to your website visitors.

**Back-end Data Loading** - Javascript provides Ajax library which helps in loading back-end data while you are doing some other processing. This really gives an amazing experience to your website visitors.

**Presentations** - JavaScript also provides the facility of creating presentations which gives website look and feel. JavaScript provides RevealJS and BespokeJS libraries to build a web-based slide presentations.

**Server Applications** - Node JS is built on Chrome's Javascript runtime for building fast and scalable network applications. This is an event based library which helps in developing very sophisticated server applications including Web Servers.

### **13.8 Self-Assessment Questions**

1. In which part does the form validation occur?

- a) Client
- b) Server
- c) Both Client and Server
- d) User side

2. What is the default type of 'type' attribute of <input> element?

- a) Text
- b) Password
- c) Numerals
- d) Special Characters

3. Which attribute is used for activation of JavaScript?

- a) button
- b) checkbox
- c) url
- d) submit

4. Which attribute defines the file-select field?

- a) file

b) checkbox

c) button

d) text

5. Which attribute is not used on new forms?

a) size

b) text

c) name

d) maxlength

6. Which of the following is not used with password attribute?

a) name

b) size

c) maxlength

d) min

7. Which element is used to create multi-line text input?

a) text

b) textarea

c) submit

d) radio button

8. Which attribute is not used for the radio type?

a) name

b) value

c) checked

d) selected

9. Which one of the following is incorrect?

A. <label> tag in HTML is used for creating a tag for form elements.

B. <label> can be used to increase the clickable area of buttons

C. id attribute is used with <label> to increase the clickable area of form elements

D. None of the above

10. Which one of the following does not hold true regarding GET method in HTML?

A. Use of GET method in HTML is more secured.

B. Use of GET method enables us to bookmark the page.

C. GET has size limitation.

D. None of the above

11. What will be the output of the following JavaScript code?

```
<p id="demo"></p>
<script>
var js = 10;
js *= 5;
document.getElementById("demo").innerHTML = js;
</script>
```

a) 10

b) 50

c) 5

d) Error

12. Which of the following is not javascript data types?

a) Null type

b) Undefined type

c) Number type

d) All of the mentioned

13. What will be the output of the following JavaScript code snippet?

```
int a=1;  
if(a!=null) // JavaScript not equal to Operators  
    return 1;  
else  
    return 0;
```

a) 0

b) 1

c) compiler error

d) runtime error

14. Which of the following is not true 3. Which of the following is not true about JavaScr about JavaScript?

A.It is a scripting language.

B.It execute with preliminary compilation. preliminary compilation.

C.It is a lightweight programming language.

D.It can be embedded directly into HTML pages.

15. Which of the following JavaScript can do?

A.JavaScript can react to events.

B.JavaScript can manipulate HTML elements.

C.JavaScript can be use to validate data.

D.All of the Above

### **13.9 Review questions**

1. Discuss different form validation along with there usage

2. Write down the steps to create a form with the fields :

Name:

Age

Gender

Phone number

Using JavaScript submit the form.

3. What is an email validation? Discuss different validation used for creating an email form.

4. Explain different types of password validation.

### **Self-assessment answers**

1. B	2. A	3. A	4. A	5. A
6. D	7. B	8. D	9. C	10. A
11. B	12. D	13. B	14. B	15. D

### **13.10 Further**

 Book	<a href="https://www.doc-developpement-durable.org/file/Projets-informatiques/cours-&amp;-manuels-informatiques/htm-html-xml-ccs/Sams%20Teach%20Yourself%20HTML,%20CSS,%20and%20JavaScript%20All%20in%20One.pdf">https://www.doc-developpement-durable.org/file/Projets-informatiques/cours-&amp;-manuels-informatiques/htm-html-xml-ccs/Sams%20Teach%20Yourself%20HTML,%20CSS,%20and%20JavaScript%20All%20in%20One.pdf</a>
 Online Links	<a href="https://www.doc-developpement-durable.org/file/Projets-informatiques/cours-&amp;-manuels-informatiques/htm-html-xml-ccs/Sams%20Teach%20Yourself%20HTML,%20CSS,%20and%20JavaScript%20All%20in%20One.pdf">https://www.doc-developpement-durable.org/file/Projets-informatiques/cours-&amp;-manuels-informatiques/htm-html-xml-ccs/Sams%20Teach%20Yourself%20HTML,%20CSS,%20and%20JavaScript%20All%20in%20One.pdf</a>

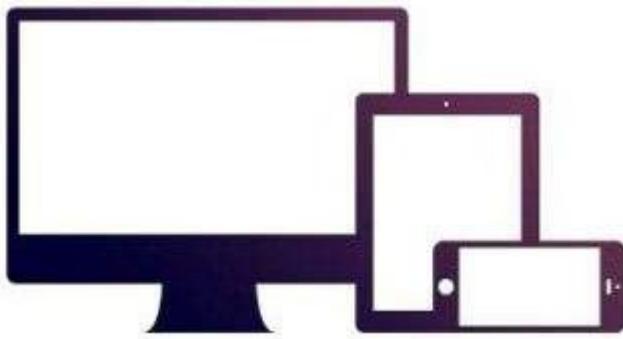
## Unit 14: Bootstrap

### Objectives

- To understand the use of Bootstrap
- To implement the different functions in bootstrap
- How to create a responsive website
- To learn about the advance CSS

### **14.1 Introduction**

Bootstrap is the most popular HTML, CSS and JavaScript framework for developing a responsive and mobile friendly website. It is absolutely free to download and use. It is a front-end framework used for easier and faster web development. It includes HTML and CSS based design templates for typography, forms, buttons, tables, navigation, modals, image carousels and many others. It can also use JavaScript plug-ins. It facilitates you to create responsive designs.



Bootstrap is a sleek, intuitive, and powerful, mobile first front-end framework for faster and easier web development. It uses HTML, CSS, and Javascript. Bootstrap was developed by Mark Otto and Jacob Thornton at Twitter. It was released as an open source product in August 2011 on GitHub.

### Why use Bootstrap?

- **Mobile first approach:** Bootstrap 3 framework consists of Mobile first styles throughout the entire library instead of them in separate files.
- **Browser Support:** It is supported by all popular browsers.
- **Easy to get started:** With just the knowledge of HTML and CSS anyone can get started with Bootstrap. Also the Bootstrap official site has a good documentation.
- **Responsive design:** Bootstrap's responsive CSS adjusts to Desktops, Tablets and Mobiles. More about the responsive design is in the chapter Bootstrap Responsive Design.



Provides a clean and uniform solution for building an interface for developers.

It contains beautiful and functional built-in components which are easy to customize.

It also provides webbased customization.

Andbest of all it is an open source.

## **What Bootstrap Package Includes?**

•**Scaffolding:** Bootstrap provides a basic structure with Grid System, link styles, and background. This is covered in detail in the section Bootstrap Basic Structure.

Nil **CSS:** Bootstrap comes with the feature of global CSS settings, fundamental HTML elements styled and enhanced with extensible classes, and an advanced grid system. This is covered in detail in the section Bootstrap with CSS.

Nil **Components:** Bootstrap contains over a dozen reusable components built to provide iconography, dropdowns, navigation, alerts, pop-overs, and much more. This is covered in detail in the section Layout Components.

Nil **JavaScript Plugins:** Bootstrap contains over a dozen custom jQuery plugins. You can easily include them all, or one by one. This is covered in details in the section Bootstrap Plugins.

Nil **Customize:** You can customize Bootstrap's components, LESS variables, and jQuery plugins to get your very own version.

## **14.2 Bootstrap – Environment Setup**

It is very easy to setup and start using Bootstrap. This chapter will explain how to download and setup Bootstrap. We will also discuss the Bootstrap file structure, and demonstrate its usage with an example.

### **Download Bootstrap**

You can download the latest version of Bootstrap from <http://getbootstrap.com/>. When you click on this link, you will get to see a screen as below:



Here you can see two buttons:

Nil **Download Bootstrap:** Clicking this, you can download the precompiled and minified versions of Bootstrap CSS, JavaScript, and fonts. No documentation or original source code files are included.

•**Download Source:** Clicking this, you can get the latest Bootstrap LESS and JavaScript source code directly from GitHub.

If you work with Bootstrap's uncompiled source code, you need to compile the LESS files to produce usable CSS files. For compiling LESS files into CSS, Bootstrap officially supports only Recess, which is Twitter's CSS hinter based on less.js.

For better understanding and ease of use, we shall use precompiled version of Bootstrap throughout the tutorial. As the files are compiled and minified, you don't have to bother every time including separate files for individual functionality. At the time of writing this tutorial the latest version (Bootstrap 3) was downloaded.

## File structure

### Precompiled Bootstrap

Once the compiled version Bootstrap is downloaded, extract the ZIP file, and you will see the following file/directory structure:

```
bootstrap/
  css/
    bootstrap.css
    bootstrap.min.css
    bootstrap-theme.css
    bootstrap-theme.min.css
  js/
    bootstrap.js
    bootstrap.min.js
  fonts/
    glyphicon-halflings-regular.eot
    glyphicon-halflings-regular.svg
    glyphicon-halflings-regular.ttf
    glyphicon-halflings-regular.woff
```

As you can see, there are compiled CSS and JS (bootstrap.\*), as well as compiled and minified CSS and JS (bootstrap.min.\*). Fonts from Glyphicons are included, as it is the optional Bootstrap theme.

### Bootstrap Source Code

If you have downloaded the Bootstrap source code then the file structure would be as follows:

```
  less/
  js/
  fonts/
  dist/
    css/
    js/
      fonts/
  docs-assets/
  examples/
  *.html
```

The files under less/, js/, and fonts/ are the source code for Bootstrap CSS, JS, and icon fonts (respectively).

The dist/ folder includes everything listed in the precompiled download section above.

docs/assets/, examples/, and all \*.html files are Bootstrap documentation.

### HTMLTemplate

A basic HTML template using Bootstrap would look like this:

```
<!DOCTYPE html>
<html>
<head>
<title>Bootstrap 101 Template</title>
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<!-- Bootstrap -->
<link href="css/bootstrap.min.css" rel="stylesheet">
<!-- HTML5 Shim and Respond.js IE8 support of HTML5 elements and media queries
-->
<!-- WARNING: Respond.js doesn't work if you view the page
```

```
via file:// -->
<!--[if lt IE 9]>
<script src="https://oss.maxcdn.com/libs/html5shiv/3.7.0/
html5shiv.js"></script>
<script src="https://oss.maxcdn.com/libs/respond.js/1.3.0/
respond.min.js"></script>
<![endif]-->
</head>
<body>
<h1>Hello, world!</h1>
<!-- jQuery (necessary for Bootstrap's JavaScript plugins) -->
<script src="https://code.jquery.com/jquery.js"></script>
<!-- Include all compiled plugins (below), or include individual files as needed
-->
<script src="js/bootstrap.min.js"></script>
</body>
</html>
```

Here you can see the jquery.js, bootstrap.min.js and bootstrap.min.css files that are included to make a normal HTM file to the Bootstrapped Template. Just make sure to include jQuery library before you include Bootstrap library. More details about each of the elements in this above piece of code will be discussed in the chapter Bootstrap CSS Overview. This template structure is already included as part of the Try it (online compiler) tool. Hence in all the examples (in the following chapters) of this tutorial you will only see the contents of the element. Once you click on the Try it option available at the top right corner of example, and you will see the entire code.

### **Example**

Now let's try an example using the above template. Try the following example using Tryit option available at the top right corner of the below sample code box on our website:

```
<h1>Hello, world!</h1>
```

In all the subsequent chapters we have used dummy text from the site

<http://www.lipsum.com/>.

### **14.3 Bootstrap – Grid System**

#### **What is Grid?**

In graphic design, a grid is a structure (usually two-dimensional) made up of a series of intersecting straight (vertical, horizontal) lines used to structure the content. It is widely used to design layout and content structure in print design. In web design, it is a very effective method to create a consistent layout rapidly and effectively using HTML and CSS.

To put in simple words, grids in web design organize and structure content, makes the websites easy to scan and reduces the cognitive load on users.

#### **What is Bootstrap Grid System?**

Bootstrap includes a responsive, mobile first fluid grid system that appropriately scales up to 12 columns as the device or viewport size increases. It includes predefined classes

for easy layout options, as well as powerful mixins for generating more semantic layouts. Let us understand the above statement. Bootstrap 3 is mobile first in the sense that the code for Bootstrap now starts by targeting smaller screens like mobile devices, tablets, and then "expands" components and grids for larger screens such as laptops, desktops.

### Mobile First Strategy

- Content

- o Determine what is most important.

- Layout

- o Design to smaller widths first.

- o Base CSS address mobile device first; media queries address for tablet, desktops.

- Progressive Enhancement

- o Add elements as screen size increases.

## 14.4 Working of Bootstrap Grid System

Grid systems are used for creating page layouts through a series of rows and columns that house your content. Here's how the Bootstrap grid system works:

- Rows must be placed within a .container class for proper alignment and padding.
- Use rows to create horizontal groups of columns
- Content should be placed within the columns, and only columns may be the immediate children of rows.
- Predefined grid classes like .row and .col-xs-4 are available for quickly making grid layouts. LESS mixins can also be used for more semantic layouts.
- Columns create gutters (gaps between column content) via padding. That padding is offset in rows for the first and the last column via negative margin on rows.
- Grid columns are created by specifying the number of twelve available columns you wish to span. For example, three equal columns would use three. col-xs-4.

### Media Queries

Media query is a really fancy term for "conditional CSS rule". It simply applies some CSS, based on certain conditions set forth. If those conditions are met, the style is applied. Media Queries in Bootstrap allow you to move, show and hide content based on the viewport size. Following media queries are used in LESS files to create the key breakpoints in the Bootstrap grid system.

```
* Extra small devices (phones, less than 768px) */
/* No media query since this is the default in Bootstrap */
/* Small devices (tablets, 768px and up) */
@media (min-width: @screen-sm-min) { ... }

/* Medium devices (desktops, 992px and up) */
@media (min-width: @screen-md-min) { ... }

/* Large devices (large desktops, 1200px and up) */
@media (min-width: @screen-lg-min) { ... }
```

Occasionally these are expanded to include a max-width to limit CSS to a narrower set of devices.

```
@media (max-width: @screen-xs-max) { ... }  
@media (min-width: @screen-sm-min) and (max-width: @screen-sm-max) { ... }  
@media (min-width: @screen-md-min) and (max-width: @screen-md-max) { ... }  
@media (min-width: @screen-lg-min) { ... }
```

Media queries have two parts, a device specification and then a size rule. In the above case, the following rule is set:

Let us consider this line:

```
@media (min-width: @screen-sm-min) and (max-width: @screen-sm-max) { ... }
```

For all devices no matter what kind with min-width: @screen-sm-min, if the width of the screen gets smaller than @screen-sm-max, then do something.

## **Grid Options**

The following table summarizes aspects of how Bootstrap grid system works across multiple devices:

	<b>Extra small devices Phones (&lt;768px)</b>	<b>Small devices Tablets (≥768px)</b>	<b>Medium devices Desktops (≥992px)</b>	<b>Large devices Desktops (≥1200px)</b>
Grid behavior	Horizontal at all times	Collapsed to start, horizontal above breakpoints	Collapsed to start, horizontal above breakpoints	Collapsed to start, horizontal above breakpoints
Max container width	None (auto)	750px	970px	1170px
Class prefix	.col-xs-	.col-sm-	.col-md-	.col-lg-
# of columns	12	12	12	12
Max column width	Auto	60px	78px	95px
Gutter width	30px (15px on each side of a column)	30px (15px on each side of a column)	30px (15px on each side of a column)	30px (15px on each side of a column)
Nestable	Yes	Yes	Yes	Yes
Offsets	Yes	Yes	Yes	Yes

## **Basic Grid Structure**

Following is basic structure of Bootstrap grid:

```
<div class="container">
<div class="row">
<div class="col-*-*"></div>
<div class="col-*-*"></div>
</div>
<div class="row">...</div>
</div>
<div class="container">....
```

### **Example: Stacked-to-horizontal**

Let us see a simple grid example with simple layout: two columns, two paragraphs per column. (Here styling for each column is used. You can avoid it.)

```
<div class="container">
<h1>Hello, world!</h1>
<div class="row">
<div class="col-md-6" style="background-color: #dedef8; box-shadow: inset 1px -1px 1px #444, inset -1px 1px 1px #444;">
<p>Lorem ipsum dolor sit amet, consectetur adipisciingelit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quisnostrud exercitation ullamcolaboris nisi utaliquip ex eacmodoconsequat.
</p>
<p>Sed utperspiciatisundeomnisistenatus error sit voluptatem accusantiumdoloremquelaudantium, totam rem aperiam, eaqueipsa quae ab illoinventoreveritatis et quasi architecto beatae vitae dicta sunt explicabo.
</p>
</div>
<div class="col-md-6" style="background-color: #dedef8;box-shadow: inset 1px -1px 1px #444, inset -1px 1px 1px #444;">
<p>Sed utperspiciatisundeomnisistenatus error sit voluptatem accusantiumdoloremquelaudantium.
</p>
<p>Nequeporroquisquamest, qui dolorem ipsum quia dolor sit amet, consectetur, adipiscivelit, sed quia non numquameius moditemporainciduntut labore et dolore magnamaliquamquaerat voluptatem.
</p>
</div>
</div>
```

## — Hello, world!

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.

Sed ut perspiciatis unde omnis iste natus error sit voluptatem accusantium doloremque laudantium, totam rem aperiam, eaque ipsa quae ab illo inventore veritas et quasi architecto beatis vitae dicta sunt explicabo.

Sed ut perspiciatis unde omnis iste natus error sit voluptatem accusantium doloremque laudantium.

Necque perro quisquam est, qui dolorem ipsum quia dolor sit amet, consectetur, adipisci velit, sed quia non numquam eius modi tempora incidunt ut labore et dolore magnam aliquam quaerat voluptatem.

### Details

- `<div class="container">...</div>` element is added to ensure proper centering and maximum width for layout.
- Once container is added, next you need to think in terms of rows. Add `<div class="row">...</div>` and columns `<div class="col-md-6"></div>` inside the rows.
- Every row in the grid is made up of 12 units and you can define the desired size of your columns using those units. In our example we have two columns each made of 6 units wide i.e  $6+6=12$ .

You can try some more options like:

`<div class="col-md-3"></div>` and `<div class="col-md-9"></div>`  
or `<div class="col-md-7"></div>` and `<div class="col-md-5"></div>`.

Experiment and make sure that the sum always needs to be 12.

### Example: Medium and Large Device

We have seen the basic grid system in ‘Example: Stacked-to-horizontal’. Here we have used 2 divs and gave them the 50%/50% split at the medium viewport width:

```
<div class="col-md-6">....</div>
<div class="col-md-6">....</div>
```

But at large, your design could really be better as a 33%/66%. So what we’re going to do is, set it up to change the column widths at the breakpoint:

```
<div class="col-md-6 col-lg-4">....</div>
<div class="col-md-6 col-lg-4">....</div>
```

Now Bootstrap is going to say “at the medium size, I look at classes with md in them and use those. At the large size, I look at classes with the word lg in them and use those. In this case, our 2 divs will go from a 50%/50% split and then up to a 33%/66%. Check it out in the following example. (Here styling for each column is used. You can avoid it.)

```
<div class="container">
<h1>Hello, world!</h1>
```

```

<div class="row">
<div class="col-md-6 col-lg-4" style="background-color: #dedef8;
box-shadow: inset 1px -1px 1px #444, inset -1px 1px 1px #444;">
<p>Lorem ipsum dolor sit amet, consectetur adipisciingelit, sed do
eiusmodtemporincididunt labore et dolore magna aliqua. Ut
enim ad minim veniam, quisnostrud exercitation ullamcolaboris
nisi utaliquip ex eaccommodoconsequat.

</p>
<p>Sed utpersiciatisundeomnisistenatus error sit voluptatem
accusantiumdoloremque laudantium, totam rem aperiam, eaqueipsa
quae ab illoinventoreveritatis et quasi architecto beatae
vitae dicta sunt explicabo.

</p>
</div>
<div class="col-md-6 col-lg-8"" style="background-color: #dedef8;
box-shadow: inset 1px -1px 1px #444, inset -1px 1px 1px #444;">
<p>Sed utpersiciatisundeomnisistenatus error sit voluptatem
accusantiumdoloremque laudantium.

</p>
<p>Nequeporroquisquamest, qui dolorem ipsum quia dolor sit
amet, consectetur, adipiscivelit, sed quia non numquameius
moditemporaincidunt labore et dolore magnamaliquamquaerat
voluptatem.

</p>
</div>
</div>

```

## Hello, world!

Latrem ipsum dolor sit amet, consectetur adipisciingelit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quisnostrud exercitation ullamcolaboris nisi utaliquip ex eaccommodoconsequat.

Sed ut persiciatis unde omnis iste natus error sit voluptatem accusantiumdoloremque laudantium. Neque porro quisquam est, qui dolorem ipsum quia dolor sit amet, consectetur, adipisci velit, sed quia non numquameius eius modi tempora incident ut labore et dolore magnamaliquamquaerat voluptatem.

Sed ut persiciatis unde omnis iste natus error sit voluptatem accusantiumdoloremque laudantium, totam rem aperiam, eaque ipsa quae ab illoinventoreveritatis et quasi architecto beatae vitae dicta sunt explicabo.

## 14.5 Bootstrap – CSSOverview

### HTML5doctype

Bootstrap makes use of certain HTML elements and CSS properties that require the useof the HTML5 doctype. Hence, include the below piece of code for HTML5 doctype at thebeginning of all your projects using Bootstrap.

```
<!DOCTYPE html>
<html>
....
```

```
</html>
```

## Mobile First

Since Bootstrap 3 has been launched, Bootstrap has become ‘mobile first’. It means mobile first styles can be found throughout the entire library instead of them in separate files. You need to add the viewport meta tag to the `<head>` element, to ensure proper rendering and touch zooming on mobile devices.

```
<meta name="viewport" content="width=device-width, initial-scale=1.0">
```

*width* property controls the width of the device. Setting it to `device-width` will make sure that it is rendered across various devices (mobiles, desktops, tablets...) properly.

*initial scale=1.0* ensures that when loaded, your web page will be rendered at a 1:1 scale, and no zooming will be applied out of the box.

Add **user-scalable=no** to the content attribute, to disable zooming capabilities on mobile devices as shown below. Users are only able to scroll and not zoom with this change, and results in your site feeling a bit more like a native application.

```
<meta name="viewport" content="width=device-width,  
initial-scale=1.0,  
maximum-scale=1.0,  
user-scalable=no">
```

Normally `maximum-scale=1.0` is used along with `user-scalable=no`. As mentioned above `user-scalable=no` may give users an experience more like a native app, hence Bootstrap doesn't recommend using this attribute.

## 14.6 Responsive Images

Bootstrap 3 allows you to make the images responsive by adding a class `.img-responsive` to the `<img>` tag. This class applies `max-width: 100%;` and `height: auto;` to the image so that it scales nicely to the parent element.

```

```

## Typography and Links

Bootstrap sets a basic global display (background), typography, and link styles:

Basic Global display: Sets `background-color: #fff;` on the `<body>` element.

Typography: Uses the `@font-family-base`, `@font-size-base`, and `@line-height-base` attributes as the typographic base.

Link styles: Sets the global link color via attribute `@linkcolor` and apply link underlines only on: hover.

If you intend to use LESS code, you may find all these within `scaffolding.less`.

## Containers

Use class `.container` to wrap a page's content and easily center the content's as shown below.

```
<div class="container">  
...  
</div>
```

Take a look at the .container class in bootstrap.css file:

```
.container {
padding-right: 15px;
padding-left: 15px;
margin-right: auto;
margin-left: auto;
}
```

Note that, due to padding and fixed widths, containers are not nestable by default.

Take a look at bootstrap.css file:

```
@media (min-width: 768px) {
.container {
width: 750px;
}
}
```

Here you can see that CSS has media-queries for containers with width. This helps for applying responsiveness and within those the container class is modified accordingly to render the grid system properly.

## 14.7 Bootstrap – Typography

Bootstrap uses Helvetica Neue, Helvetica, Arial, and sans-serif in its default font stack. Using typography feature of Bootstrap you can create headings, paragraphs, lists and other inline elements. Let's learn each one of these in the following sections.

### Headings

All HTML headings (h1 to h6) are styled in Bootstrap. An example is shown below:

```
<h1>I'm Heading1 h1</h1>
<h2>I'm Heading2 h2</h2>
<h3>I'm Heading3 h3</h3>
<h4>I'm Heading4 h4</h4>
<h5>I'm Heading5 h5</h5>
<h6>I'm Heading6 h6</h6>
```

The above code segment with Bootstrap will produce following result:

 <b>Notes</b>	<p>Internet Explorer 9 and down is not supported by Bootstrap 4. Although Internet Explorer 8-9 supported Bootstrap 3. So, if you have Internet Explorer 8-9, you should use Bootstrap 3. Bootstrap 3 is the most stable version of Bootstrap, and it is still supported by the team for critical bugfixes and documentation changes.</p>
---	---

I'm Heading1 h1

I'm Heading2 h2

I'm Heading3 h3

I'm Heading4 h4

I'm Heading5 h5

I'm Heading6 h6

---

#### **14.8 Summary**

- A website is called responsive website which can automatically adjust itself to look good on all devices, from smart phones to desktops etc.
- Bootstrap provides a basic structure with Grid System, link styles, and background.
- Bootstrap comes with the feature of global CSS settings, fundamental HTML elements style and an advanced grid system.
- Bootstrap is more than efficient to create a responsive and mobile first website, but it is not the best in the industry.
- In Bootstrap, container is used to set the content's margins dealing with the responsive behaviors of your layout. It contains the row elements, and the row elements are the container of columns (known as grid system).

#### **14.9 Keywords**

- Scaffolding: Bootstrap provides a basic structure with Grid System, link styles, and background.
- CSS: Bootstrap comes with the feature of global CSS settings, fundamental HTML elements style and an advanced grid system.
- Components: Bootstrap contains a lot of reusable components built to provide iconography, dropdowns, navigation, alerts, popovers, and much more.
- JavaScript Plugins: Bootstrap also contains a lot of custom jQuery plugins. You can easily include them all, or one by one.
- Customize: Bootstrap components are customizable, and you can customize Bootstrap's components, LESS variables, and jQuery plugins to get your own style.
- Container: Container is used to set the content's margins dealing with the responsive behaviors of your layout. It contains the row elements, and the row elements are the container of columns (known as grid system).

#### **14.10 Self-Assessment Questions**

1. Medium Devices Are Defined As Having A Screen Width From

A. 900 Pixels To 1000 Pixels

B. 768 Pixels To 991 Pixels

C. 512 Pixels To 2048 Pixels

D.992 Pixels To 1199 Pixels

2. What Layout Is Used For Providing 100% Width In Bootstrap?

A. Fluid Layout

B.Fixed Layout

C.Both (a)and (b)

D.None Of The Above

3. The Bootstrap Class Xs Means For

A. Tablets

B/Desktop

C.Phones

D.Larger Desktops

4. Which Class Indicates A Dropdown Menu?

A. dropdown

B..select

C..dropdown-list

D..dropup-list

5. Which Class Creates A List Of Items?

A. Lst-group

B.List-group

C.List-grp

D.Menu-group

6. Which Of The Following Contextual Class Is Used For Warning Purpose?

A. .active

B..warning

C..danger

D.AllOf The Above

7. Which Of The Following Class Is Used To Create A Button As A Link In Bootstrap?

A. .btn-hyperlink

B..btn-link

C..btn-anchor

D.NoneOf These

8. Which of the following grid class is used for desktops?

A. md

B.lg

C.sm

D.xs

9. Bootstrap Is Developed By

A. James Gosling

B.Mark Otto And Jacob Thornton

C.MarkJukervich

D.NoneOf Them

10. The Bootstrap Class Md Means For

A. Desktop

B.Tablets

C.Phones

D.Larger Desktops

11. A Standard Navigation Tab Is Created With:

A. <ul Class="navigation-tabs">

B.<ul Class="nav Tabs">

C.<ul Class="navnav-tabs">

D,<ul Class="navnav-navbar">

12. Which Plugin Is Used To Cycle Through Elements, Like A Slideshow?

A. Orbit

B.Scrollspy

C.Slideshow

D.Carousel

13. which class provides a responsive fixed width container in bootstrap?

A. .container-fixed

B..container

C..container-fluid

D.None of above

14.T hebootstrap grid system is based on how many columns?

A. 12 column

B.20 column

C.10 column

D.15 column

15. Bootstrap is used for

A. Data

B.IoT

C.Bigdata

D.Web applications

#### **14.11 Review Questions**

1. Discuss different containers used in Bootstraps?
2. Explain the applicability of Bootstrap.
3. What is the purpose of CSS?
4. What do you mean by Responsive web page or website? Explain different types of dimensions for different devices.

5. A	6. A	7. C	8. A	9. B
10. B	11. B	12. A	13. B	14. A
15. B	16. D	17. B	18. A	19. D

 Book	<a href="https://wiki.lib.sun.ac.za/images/0/07/Bootstrap-tutorial.pdf">https://wiki.lib.sun.ac.za/images/0/07/Bootstrap-tutorial.pdf</a>
 Online Links	<a href="https://www.pdfdrive.com/bootstrap-e41430539.html">https://www.pdfdrive.com/bootstrap-e41430539.html</a>

**LOVELY PROFESSIONAL UNIVERSITY**

Jalandhar-Delhi G.T. Road (NH-1)

Phagwara, Punjab (India)-144411

For Enquiry: +91-1824-521360

Fax.: +91-1824-506111

Email: odl@lpu.co.in

ISBN 978-81-19334-33-9

A standard linear barcode representing the ISBN number 978-81-19334-33-9.

9 788119 334339