PERIYAR UNIVERSITY

NAAC 'A++' Grade - State University - NIRF Rank 56 – State Public University Rank 25
SALEM - 636 011, Tamil Nadu, India.

CENTRE FOR DISTANCE AND ONLINE EDUCATION

(CDOE)

BACHELOR OF SCIENCE (COMPUTER SCIENCE)

SEMESTER - IV

SEC – INTRODUCTION TO HTML

(Candidates admitted from 2024 onwards)

PERIYAR UNIVERSITY

CENTRE FOR DISTANCE AND ONLINE EDUCATION (CDOE)

B.Sc.,(Computer Science) 2024 admission onwards

SEC – INTRODUCTION TO HTML

Prepared by:

Centre for Distance and Online Education (CDOE)

Periyar University, Salem – 11.

SYLLABUS

INTRODUCTION TO HTML

Unit I: Foundations of HTML and Basic elements: What is HTML? A simple HTML Document. - Web browsers–HTML Page structure- HTML History- Client-Server Model- HTML overview: HTML tags – Document Structure-Basics tags.

Unit – II: HTML element: Tag Vs Element, HTML Attributes: Core Attributes – Class Attributes – Internationalization Attributes. HTML formatting (Bold, talic, Underlined, Strike, Monospaced, Superscript, Subscript, Inserted, Deleted, Large, Smaller, Grouping Content) HTML Phrase tags(Emphasized, marked, strong, text abbreviation, Acronym element, Text Direction, Special terms, Quoting text, Short Quotations, Text citations, Computer code, Keyboard text, Programming variables, Address Text), HTML meta tags

Unit - III: Lists: Types of lists: Ordered, Unordered– Nesting Lists– Other tags: Marquee, HR, BR. Links: Hyperlinks – HTML links syntax – HTML links Target attribute – Absolute URLs Vs Relative URLs, HTML links Using Images.

Unit – IV: Tables: HTML Tables- Definition and Usage- Define a HTML table- Table Cells – Table rows – Table headers – HTML Table tags- HTML Tables Colspan & Rowspan – HTML Table Padding & Spacing.

Unit – V: HTML iFrames: Frameset–Targeted Links–HTML iframe tag– Forms: Input, Text fields – the <label> element – the Name Attribute for<input> - HTML Input types.

LIST OF CONTENTS

	Spacing.	
5	HTML iFrames: Frameset–Targeted Links–HTML iframe tag–Forms: Input, Text fields – the <label> element – the Name Attribute for<input> - HTML Input types.	129 - 152

INTRODUCTION TO HTML

UNIT 1 - Foundations of HTML and Basic elements
What is HTML? A simple HTML Document. - Web browsers–HTML Page structure- HTML History- Client-Server Model- HTML overview: HTML tags – Document Structure-Basics tags.

TABLE OF CONTENTS

UNIT OBJECTIVES :

In this unit, learners will have a deep foundations in HTML and its basic elements. Learners will be able to create a web page using basic commands, and able to apply the same concept for real time applications in our day to day life.

1.1 Foundations of HTML and Basic elements

HTML is the standard markup language for creating Web pages.

1.1.1. What is HTML?

- HTML stands for Hyper Text Markup Language
- HTML is the standard markup language for creating Web pages
- HTML describes the structure of a Web page
- HTML consists of a series of elements
- HTML elements tell the browser how to display the content
- HTML elements label pieces of content such as "this is a heading", "this is a paragraph", "this is a link", etc.

1.2. A Simple HTML Document

Example

```
<!DOCTYPE html>
<html>
<head>
<title>Page Title</title>
</head>
<body>

<h1>My First Heading</h1>
<p>My first paragraph.</p>
```

</body>

</html>

Example Explained

- The <!DOCTYPE html> declaration defines that this document is an HTML5 document
- The <html> element is the root element of an HTML page
- The <head> element contains meta information about the HTML page
- The <title> element specifies a title for the HTML page (which is shown in the browser's title bar or in the page's tab)
- The <body> element defines the document's body, and is a container for all the visible contents, such as headings, paragraphs, images, hyperlinks, tables, lists, etc.
- The <h1> element defines a large heading
- The element defines a paragraph

HTML Element

An HTML element is defined by a start tag, some content, and an end tag:

<tagname> Content goes here... </tagname>

The HTML **element** is everything from the start tag to the end tag:

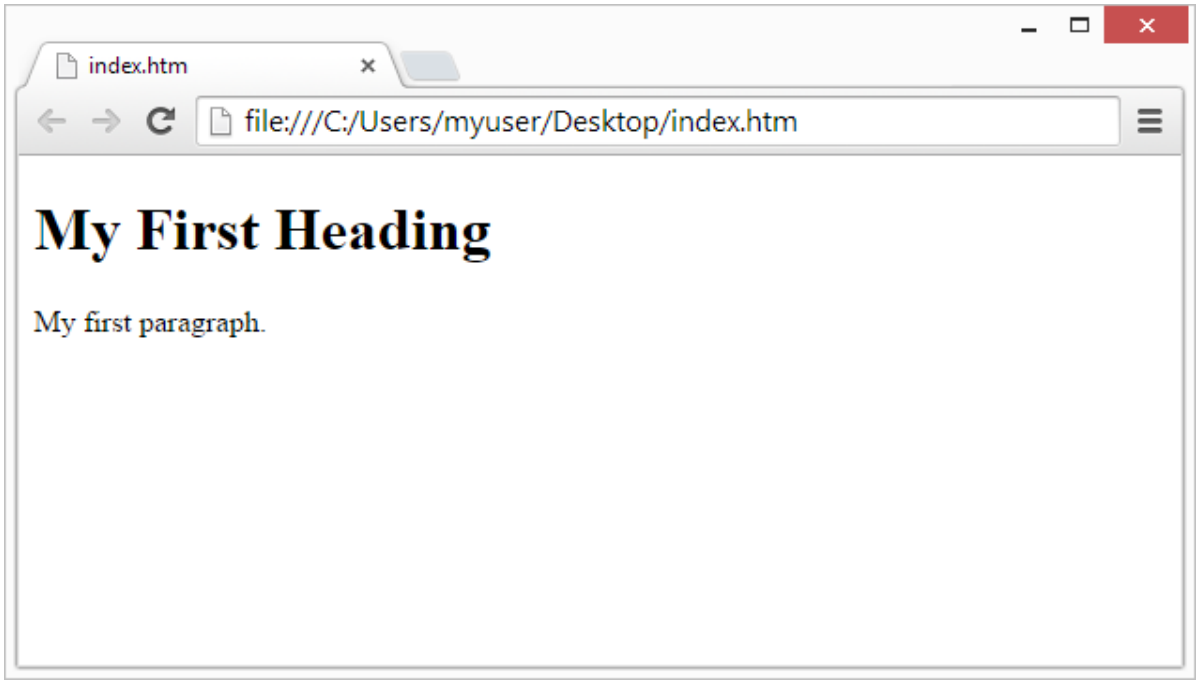
<h1>My First Heading</h1>

My first paragraph.

Start tag	Element content	End tag
<h1>	My First Heading	</h1>
	My first paragraph.	
 	none	*none*

1.3.Web Browsers

The purpose of a web browser (Chrome, Edge, Firefox, Safari) is to read HTML documents and display them correctly.A browser does not display the HTML tags, but uses them to determine how to display the document.



1.4. HTML Page Structure

Below is a visualization of an HTML page structure:

```
<html>
<head>
<title>Page title</title>
</head>
<body>
<h1>This is a heading</h1>
<p>This is a paragraph.</p>
<p>This is another paragraph.</p>
```

</body>

</html>

The content inside the <body> section will be displayed in a browser. The content inside the <title> element will be shown in the browser's title bar or in the page's tab.

1.5. HTML History

Since the early days of the World Wide Web, there have been many versions of HTML:

Year	Version
1989	Tim Berners-Lee invented www
1991	Tim Berners-Lee invented HTML
1993	Dave Raggett drafted HTML+
1995	HTML Working Group defined HTML 2.0
1997	W3C Recommendation: HTML 3.2
1999	W3C Recommendation: HTML 4.01
2000	W3C Recommendation: XHTML 1.0
2008	WHATWG HTML5 First Public Draft
2012	WHATWG HTML5 Living Standard
2014	W3C Recommendation: HTML5
2016	W3C Candidate Recommendation: HTML 5.1
2017	W3C Recommendation: HTML5.1

2nd Edition

2017 W3C Recommendation: HTML5.2

1.6. Client-Server Model

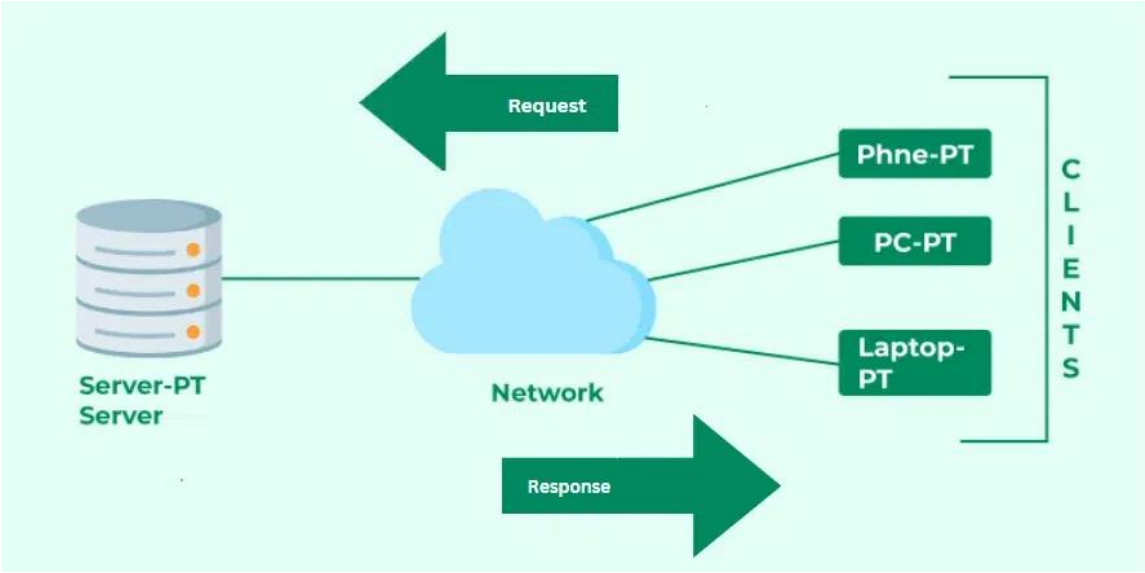
The Client-server model is a distributed application structure that partitions tasks or workloads between the providers of a resource or service, called servers, and service requesters called clients. In the client-server architecture, when the client computer sends a request for data to the server through the internet, the server accepts the requested process and delivers the data packets requested back to the client. Clients do not share any of their resources. Examples of the Client-Server Model are Email, World Wide Web, etc.

How Does the Client-Server Model Work?

In this article, we are going to take a dive into the **Client-Server** model and have a look at how the **Internet** works via, web browsers. This article will help us have a solid WEB foundation and help us easily work with WEB technologies.

- **Client:** When we say the word **Client**, it means to talk of a person or an organization using a particular service. Similarly in the digital world, a **Client** is a computer (**Host**) i.e. capable of receiving information or using a particular service from the service providers (**Servers**).
- **Servers:** Similarly, when we talk about the word **Servers**, It means a person or medium that serves something. Similarly in this digital world, a **Server** is a remote computer that provides information (data) or access to particular services.

  So, it is the **Client** requesting something and the **Server** serving it as long as it is in the database.



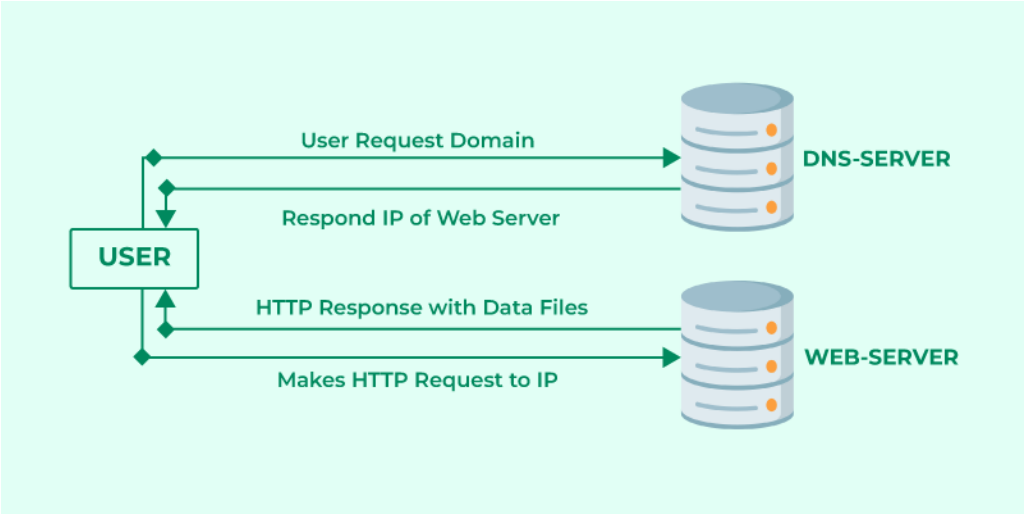
Client Server Model

How the Browser Interacts With the Servers?

There are a few steps to follow to interact with the servers of a client.

- User enters the **URL**(Uniform Resource Locator) of the website or file. The Browser then requests the **DNS**(DOMAIN NAME SYSTEM) Server.
- **DNS Server** lookup for the address of the **WEB Server**.
- The **DNS Server** responds with the **IP address** of the **WEB Server**.
- The Browser sends over an **HTTP/HTTPS** request to the **WEB Server's IP** (provided by the **DNS server**).
- The Server sends over the necessary files for the website.
- The Browser then renders the files and the website is displayed. This rendering is done with the help of **DOM** (Document Object Model) interpreter, **CSS** interpreter, and **JS Engine** collectively

known as the **JIT** or (Just in Time) Compilers.



Client Server Request and Response

Advantages of Client-Server Model

- Centralized system with all data in a single place.
- Cost efficient requires less maintenance cost and Data recovery is possible.
- The capacity of the Client and Servers can be changed separately.

Disadvantages of Client-Server Model

- Clients are prone to viruses, Trojans, and worms if present in the Server or uploaded into the Server.
- Servers are prone to Denial of Service (DOS) attacks.
- Data packets may be spoofed or modified during transmission.
- Phishing or capturing login credentials or other useful information of the user are common and MITM(Man in the Middle) attacks are common.
- The client-server architecture consolidates resources on servers for greater control and security, allows for flexible client options, and relies on a robust network for scalability and efficiency. While there are cost implications, the client-server model remains fundamental and has been shaped by trends such as cloud computing.

1.7. HTML Overview

HTML stands for Hypertext Markup Language, and it is the most widely used language to write Web Pages.

- Hypertext refers to the way in which Web pages (HTML documents) are linked together. Thus, the link available on a webpage is called Hypertext.
- As its name suggests, HTML is a Markup Language which means you use HTML to simply "mark-up" a text document with tags that tell a Web browser how to structure it to display.

Originally, HTML was developed with the intent of defining the structure of documents like headings, paragraphs, lists, and so forth to facilitate the sharing of scientific information between researchers.

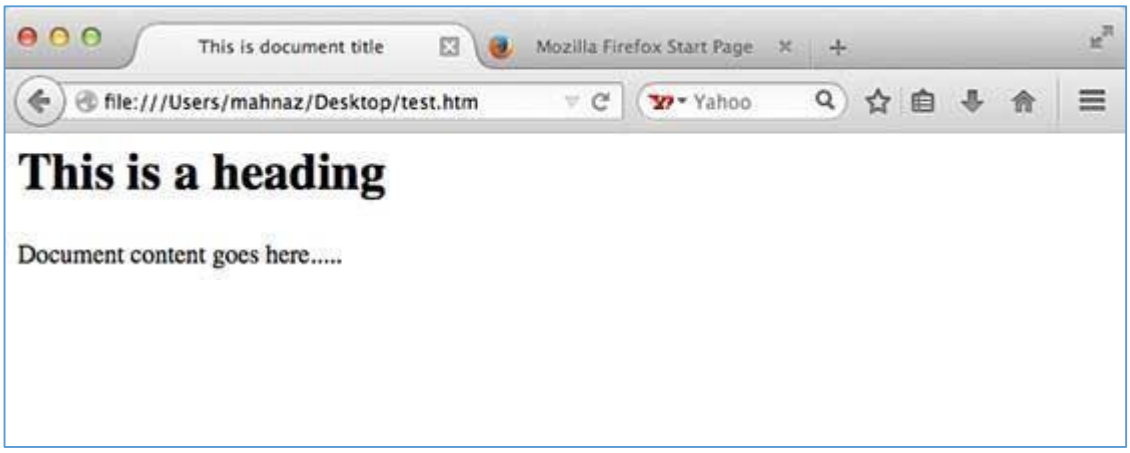
Now, HTML is being widely used to format web pages with the help of different tags available in HTML language.

Basic HTML Document

In its simplest form, following is an example of an HTML document:

```
<!DOCTYPE html>
<html>
<head>
<title>This is document title</title>
</head>
<body>
<h1>This is a heading</h1>
<p>Document content goes here   </p>
</body>
```

Either you can use Try it option available at the top right corner of the code box to check the result of this HTML code, or let's save it in an HTML file test.htm using your favorite text editor. Finally open it using a web browser like Internet Explorer or Google Chrome, or Firefox etc. It must show the following output:



HTML Tags

As told earlier, HTML is a markup language and makes use of various tags to format the content. These tags are enclosed within angle braces <Tag Name>. Except few tags, most of the tags have their corresponding closing tags. For example, <html> has its closing tag</html> and <body> tag has its closing tag </body> tag etc.

Above example of HTML document uses the following tags:

Tag	Description
<!DOCTYPE...>	This tag defines the document type and HTML version.
<html>	This tag encloses the complete HTML document and mainly comprises of document header which is represented by <head>...</head> and document body which is represented by <body>...</body> tags.
<head>	This tag represents the document's header which can keep other HTML tags like <title>, <link> etc.

<title>	The <title> tag is used inside the <head> tag to mention the document title.
<body>	This tag represents the document's body which keeps other HTML tags like <h1>, , <p> etc.
<h1>	This tag represents the heading.
<p>	This tag represents a paragraph.

To learn HTML, you will need to study various tags and understand how they behave, while formatting a textual document. Learning HTML is simple as users have to learn the usage of different tags in order to format the text or images to make a beautiful webpage.

World Wide Web Consortium (W3C) recommends to use lowercase tags starting from HTML 4.

1.8. HTML Document Structure

A typical HTML document will have the following structure:

```
Document declaration tag
<html>
   <head>
       Document header related tags
   </head>

   <body>
       Document body related tags
   </body>
</html>
```

We will study all the header and body tags in subsequent chapters, but for now let's see

what is document declaration tag.

The <!DOCTYPE> Declaration

The <!DOCTYPE> declaration tag is used by the web browser to understand the version of the HTML used in the document. Current version of HTML is 5 and it makes use of the following declaration:

```
<!DOCTYPE html>
```

There are many other declaration types which can be used in HTML document depending on what version of HTML is being used. We will see more details on this while discussing
<!DOCTYPE...> tag along with other HTML tags.

1.9. HTML Basic tags

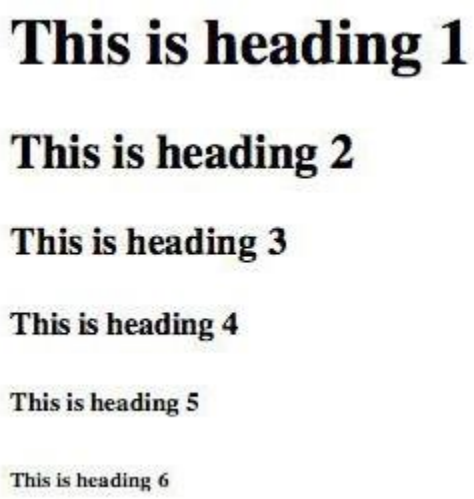
Heading Tags

Any document starts with a heading. You can use different sizes for your headings. HTML also has six levels of headings, which use the elements **<h1>, <h2>, <h3>, <h4>, <h5>, and**
<h6>. While displaying any heading, browser adds one line before and one line after that heading.

Example

```
<!DOCTYPE html>
<html>
<head>
<title>Heading Example</title>
</head>
<body>
<h1>This is heading 1</h1>
<h2>This is heading 2</h2>
<h3>This is heading 3</h3>
<h4>This is heading 4</h4>
<h5>This is heading 5</h5>
<h6>This is heading 6</h6>
</body>
</html>
```

This will produce the following result:



Paragraph Tag

The ** ** tag offers a way to structure your text into different paragraphs. Each paragraph of text should go in between an opening and a closing tag as shown below in

the example:

Example

```
<!DOCTYPE html>
<html>
<head>
<title>Paragraph Example</title>
</head>
<body>
<p>Here is a first paragraph of text.</p>
<p>Here is a second paragraph of text.</p>
<p>Here is a third paragraph of text.</p>
</body>
</html>
```

This will produce the following result:

```
Here is a first paragraph of text.
Here is a second paragraph of text.
Here is a third paragraph of text.
```

Line Break Tag

Whenever you use the **
** element, anything following it starts from the next line. This tag is an example of an **empty** element, where you do not need opening and closing tags, as there is nothing to go in between them.

The
 tag has a space between the characters **br** and the forward slash. If you omit this space, older browsers will have trouble rendering the line break, while if you miss the forward slash character and just use
 it is not valid in XHTML.

Example

```
<!DOCTYPE html>
<html>
<head>
<title>Line Break  Example</title>
</head>
<body>
<p>Hello<br />
You delivered your assignment on time.<br />
Thanks<br />
Mahnaz</p>
</body>
</html>
```

This will produce the following result:

```
Hello
You delivered your assignment on time.
Thanks
Mahnaz
```

Centering Content

You can use ** ** tag to put any content in the center of the page or any table cell.

Example

```
<!DOCTYPE html>
<html>
<head>
<title>Centring Content Example</title>
</head>
<body>
<p>This text is not in the center.</p>
<center>
<p>This text is in the center.</p>
</center>
</body>
</html>
```

This will produce the following result:

This text is not in the center.

This text is in the center.

Horizontal Lines

Horizontal lines are used to visually break-up sections of a document. The **<hr>** tag creates a line from the current position in the document to the right margin and breaks the line accordingly.

For example, you may want to give a line between two paragraphs as in the given example below:

Example

```
<!DOCTYPE html>
<html>
<head>
<title>Horizontal Line Example</title>
</head>
<body>
<p>This is paragraph one and should be on top</p>
<hr />
<p>This is paragraph two and should be at bottom</p>
</body>
</html>
```

This will produce the following result:

This is paragraph one and should be on top

This is paragraph two and should be at bottom

Again **<hr />** tag is an example of the **empty** element, where you do not need opening and closing tags, as there is nothing to go in between them.

The **<hr />** element has a space between the characters **hr** and the forward slash. If you omit this space, older browsers will have trouble rendering the horizontal line, while if you miss the forward slash character and just use **<hr>** it is not valid in XHTML

Preserve Formatting

Sometimes, you want your text to follow the exact format of how it is written in the HTML document. In these cases, you can use the preformatted tag **<pre>**.

Any text between the opening **<pre>** tag and the closing **</pre>** tag will preserve the formatting of the source document.

Example

```
<!DOCTYPE html>
<html>
<head>
<title>Preserve Formatting Example</title>
</head>
<body>
<pre>
function testFunction( strText
   ){ alert (strText)
}
</pre>
</body>
</html>
```

This will produce the following result: function testFunction(strText){

  alert (strText)
}

Try using the same code without keeping it inside **<pre>...</pre>** tags

Nonbreaking Spaces

Suppose you want to use the phrase "12 Angry Men." Here, you would not want a browser to split the "12, Angry" and "Men" across two lines:

```
An example of this technique appears in the movie "12 Angry Men."
```

In cases, where you do not want the client browser to break text, you should

use a nonbreaking space entity ** ** instead of a normal space. For example, when coding the "12 Angry Men" in a paragraph, you should use something similar to the following code:

Example

```
<!DOCTYPE html>
<html>
<head>
<title>Nonbreaking Spaces Example</title>
</head>
<body>
<p>An example of this technique appears in the movie
"12 Angry Men."</p>
</body>
</html>
```

Summary:

1) **What is HTML?** HTML (HyperText Markup Language) is the standard language used to create and design webpages. It structures web content by defining elements and their relationships within a document.
2) **A Simple HTML Document:** An HTML document is composed of elements enclosed in tags. A basic HTML document includes:
   i) <!DOCTYPE html> declaration
   ii) <html> element
   iii) <head> section with metadata and links to resources
   iv) <body> section containing the content visible to users
3) **Web Browsers:** Web browsers interpret and render HTML documents. They read HTML code and display the structured content as webpages.
4) **HTML Page Structure:** HTML documents have a specific structure:

i) <!DOCTYPE html>: Defines the document type and HTML version
ii) <html>: The root element
iii) <head>: Contains meta-information, links to stylesheets, and scripts
iv) <body>: Contains the content of the webpage, such as text, images, and links

5) **HTML History:** HTML has evolved from its initial version in the early 1990s to HTML5, which introduced new elements and APIs to support modern web applications and multimedia.
6) **Client-Server Model:** In the client-server model, the client (browser) requests HTML documents from a server. The server responds by sending the requested HTML files, which the browser then renders.
7) **HTML Overview:** HTML tags are used to create elements and structure content. Common tags include:
   i) <h1> to <h6>: Headings
   ii) <p>: Paragraphs
   iii) <a>: Hyperlinks
   iv) <img>: Images
   v) : Division or section
8) **Document Structure and Basic Tags:** HTML documents are structured using a combination of tags. Basic tags and their purposes include:
   i) <html>: Encloses the entire document
   ii) <head>: Contains meta-information
   iii) <title>: Sets the document title
   iv) <body>: Contains the content visible to users
   v) <h1>, <h2>, <h3>, etc.: Define headings
   vi) <p>: Defines paragraphs
   vii) <a>: Creates hyperlinks
   viii) <img>: Embeds images

Glossary :

- **HTML (HyperText Markup Language):** A standard language used to create and design webpages, providing structure and organization for web content.
- **HTML Document:** A file containing HTML code, which structures and formats content for web browsers.
- **Web Browser:** Software application that interprets and displays HTML documents as webpages (e.g., Chrome, Firefox, Safari).
- **HTML Tags:** Special codes enclosed in angle brackets that define elements within an HTML document (e.g., <p>, <a>).
- **HTML Page Structure:** The organized layout of an HTML document, including <!DOCTYPE html>, <html>, <head>, and <body> elements.
- **DOCTYPE Declaration:** <!DOCTYPE html>: A declaration at the beginning of an HTML document that specifies the version of HTML used.
- **Client-Server Model:** A network architecture where a client (e.g., web browser) requests resources from a server, which responds with the requested content.
- **Head Section:** The <head> part of an HTML document that contains metadata, links to stylesheets, and scripts but does not display content directly.
- **Body Section:** The <body> part of an HTML document that contains the visible content of the webpage, such as text, images, and links.
- **HTML5:** The fifth version of HTML, which includes new elements and APIs for enhanced functionality and multimedia support.
- **Headings:** HTML tags <h1> to <h6> that define headings of different levels, with <h1> being the highest and <h6> the lowest.
- **Paragraph Tag (<p>):** An HTML tag used to define paragraphs of text within a webpage.
- **Anchor Tag (<a>):** An HTML tag used to create hyperlinks that link to other webpages, files, or locations within the same document.

- **Image Tag (<img>):** An HTML tag used to embed images in a webpage.
- **Division Tag ( ):** A generic container element used to group and style sections of content within a webpage.
- **Metadata:** Information about a webpage, such as its title or character set, included in the <head> section of an HTML document.
- **HTML Elements:** Components of an HTML document defined by tags, including text, images, and links, which are structured to create webpages.
- **HTML Attribute:** Additional information provided within HTML tags to modify the behavior or appearance of an element (e.g., href in <a href="...">).

Self Assessment Questions:

1. **Basic Understanding:**
   i. What does HTML stand for, and what is its primary purpose?
   ii. How does HTML contribute to the creation of webpages?
2. **HTML Document Structure:**
   i. What are the main components of a simple HTML document?
   ii. Describe the purpose of the <!DOCTYPE html> declaration in an HTML document.
   iii. What is the role of the <html>, <head>, and <body> elements in an HTML document?
3. **Web Browsers and HTML:**
   i. How do web browsers interact with HTML documents?
   ii. What does a web browser do when it receives an HTML file from a server?
4. **HTML Tags and Elements:**
   i. What is an HTML tag, and how is it used to define elements in an HTML document?
   ii. Name three common HTML tags and describe their functions.
5. **HTML History and Evolution:**
   i. Briefly outline the evolution of HTML from its inception to HTML5.
   ii. What are some of the new features introduced in HTML5?
6. **Client-Server Model:**

i. Explain the client-server model in the context of web development.
ii. How does a web browser (client) obtain an HTML document from a server?

7. **Document Structure:**
   i. What elements are typically found within the <head> section of an HTML document?
   ii. What kind of content is placed inside the <body> section of an HTML document?
8. **Basic HTML Tags and Their Uses:**
   i. What is the purpose of the <h1> to <h6> tags in HTML?
   ii. How is the <p> tag used to format text on a webpage?
   iii. Describe the function of the <a> tag and give an example of its usage.
   iv. What is the role of the <img> tag in an HTML document?
9. **HTML Elements and Attributes:**
   i. Define what an HTML element is and provide an example.
   ii. How do HTML attributes modify the behavior or appearance of an element? Provide an example.
10. **Practical Application:**
   i. Create a simple HTML document that includes a heading, a paragraph, a hyperlink, and an image.
   ii. Modify the document to include a DOCTYPE declaration and describe its impact on the document's rendering.

Activities and Exercises:

I. **Basic HTML Document Creation:**

Objective: Create a simple HTML document from scratch.

Instructions: Write an HTML document that includes a title, a heading, a paragraph of text, and an image. Use the following specifications:

i. Title: "My First Webpage"
ii. Heading: "Welcome to My Webpage"
iii. Paragraph: A brief introduction about yourself or a topic of your choice.

iv. Image: Include an image from a URL or a local file.

II. **HTML Structure Identification:**

Objective: Identify and describe the different parts of an HTML document.

Instructions: Given a sample HTML document, label and describe the purpose of each section: <!DOCTYPE html>, <html>, <head>, and <body>. Identify any HTML tags used and their functions.

III. **Tag Function Practice:**

Objective: Practice using various HTML tags.

Instructions: Create a webpage that includes:

i. An ordered list and an unordered list.

ii. A hyperlink that links to an external website.

iii. A table with at least two rows and two columns.

iv. A blockquote with a quote from a famous person.

IV. **Styling with HTML:**

Objective: Understand basic HTML formatting and styling.

Instructions: Use HTML tags to style text in different ways. Include:

i. Bold and italic text.

ii. A hyperlink with a tooltip (using the title attribute).

iii. A line break between two paragraphs.

V. **Client-Server Interaction Simulation:**

Objective: Understand how the client-server model works.

Instructions: Simulate a request-response cycle:

i. Use a tool like Postman or a simple web server to create and send a request for an HTML file.

ii. Observe and describe the server's response and how the browser renders the content.

Case Studies:

I. **Case Study: Building a Personal Portfolio Page**

Objective: Apply HTML skills to create a personal portfolio webpage.

Instructions: Develop a webpage to showcase a personal portfolio. Include sections such as:

About Me: Use headings and paragraphs to introduce yourself.

Projects: Create a list or table of projects with descriptions and links.

Contact: Provide a form with fields for name, email, and a message.

Include styling for layout and readability.

II. **Case Study: Analyzing an Existing Webpage**

Objective: Analyze the structure and tags used in a real-world webpage.

Instructions: Choose a well-known website and inspect its HTML source code using browser developer tools. Analyze:

a. The use of headings, paragraphs, and lists.
b. How images and links are implemented.
c. The structure of the <head> section (meta tags, links to stylesheets).
d. Any embedded multimedia or interactive elements.

III. **Case Study: Updating a Legacy HTML Page**

Objective: Update an older HTML page to meet modern standards.

Instructions: Take an outdated HTML page and update it to use HTML5 standards. Focus on:

a. Replacing deprecated tags with HTML5 equivalents.
b. Adding semantic HTML5 elements (e.g., <header>, <footer>, <article>).
c. Ensuring proper use of the <!DOCTYPE html> declaration.

IV. **Case Study: Client-Server Model in Practice**

Objective: Demonstrate a real-world application of the client-server model.

Instructions: Build a simple web application that includes:

a. A front-end HTML page that sends a request to a back-end server.
b. A server-side script (e.g., using Node.js or PHP) that processes the request and returns a response.
c. Display the server's response on the client-side webpage.

UNIT II

UNIT 2 - HTML element
Tag Vs Element, HTML Attributes: Core Attributes – Class Attributes – Internationalization Attributes. HTML formatting (Bold, talic, Underlined, Strike, Monospaced, Superscript, Subscript, Inserted, Deleted, Large, Smaller, Grouping Content) HTML Phrase tags(Emphasized, marked, strong, text abbreviation, Acronym element, Text Direction, Special terms, Quoting text, Short Quotations, Text citations, Computer code, Keyboard text, Programming variables, Address Text), HTML meta tags

TABLE OF CONTENTS

UNIT OBJECTIVES :

In this unit, learners will have a get knowledge about HTML element and formatting. Learners will be able to innovate a web page using these formatting commands, and will be able to make the world to turn into their web page by applying the style of the content.

HTML element

An **HTML element** is defined by a starting tag. If the element contains other content, it ends with a closing tag, where the element name is preceded by a forward slash as shown below with few tags:

Start Tag	Content	End Tag
	This is paragraph content.	
<h1>	This is heading content.	</h1>
	This is division content.	
 	Void elements	

So here ** .... ** is an HTML element, **<h1>...</h1>** is another HTML element. There are some HTML elements which don't need to be closed, such as **<img.../>**, **<hr />** and
**
** elements. These are known as **void elements**.
HTML documents consists of a tree of these elements and they specify how HTML documents should be built, and what kind of content should be placed in

what part of an HTML document.

2.1. HTML Tag vs. Element

An HTML element is defined by a *starting tag*. If the element contains other content, it ends with a *closing tag*.

For example, ** ** is starting tag of a paragraph and ** ** is closing tag of the same paragraph but ** This is paragraph ** is a paragraph element.

Nested HTML Elements

It is very much allowed to keep one HTML element inside another HTML element:

Example

```
<!DOCTYPE html>
<html>
<head>
<title>Nested Elements Example</title>
</head>
<body>
<h1>This is <i>italic</i> heading</h1>
<p>This is <u>underlined</u> paragraph</p>
</body>
</html>
```

This will display the following result:

This is *italic* heading

This is<u>underlined</u>paragraph

2.2. HTML Attributes

We have seen few HTML tags and their usage like heading tags **<h1>, <h2>,** paragraph tag **<p>** and other tags. We used them so far in their simplest form, but most of the HTML tags can also have attributes, which are extra bits of information.

An attribute is used to define the characteristics of an HTML element and is placed inside the element's opening tag. All attributes are made up of two parts: a **name** and a **value**:

- The **name** is the property you want to set. For example, the paragraph **<p>** element in the example carries an attribute whose name is **align**, which you can use to indicate the alignment of paragraph on the page.
- The **value** is what you want the value of the property to be set and always put within quotations. The below example shows three possible values of align attribute: **left, center** and **right**.

Attribute names and attribute values are case-insensitive. However, the World Wide Web Consortium (W3C) recommends lowercase attributes/attribute values in their HTML 4 recommendation.

Example

```
<!DOCTYPE html>
<html><head><title>Align Attribute  Example</title>
</head><body>
<p align="left">This is left aligned</p>
<p align="center">This is center aligned</p>
<p align="right">This is right aligned</p>
</body>
</html>
```

This will display the following result:

This is left aligned

This is center aligned

This is right aligned

2.2.1 Core Attributes

The four core attributes that can be used on the majority of HTML elements (although not all) are:

- Id
- Title
- Class
- Style

The Id Attribute

The **id** attribute of an HTML tag can be used to uniquely identify any element within an HTML page. There are two primary reasons that you might want to use an id attribute on an element:

- If an element carries an id attribute as a unique identifier, it is possible to identify just that element and its content.
- If you have two elements of the same name within a Web page (or style sheet), you can use the id attribute to distinguish between elements that have the same name.

We will discuss style sheet in separate tutorial. For now, let's use the id attribute to distinguish between two paragraph elements as shown below.

Example

```
<p id="html">This para explains what is HTML</p>
<p id="css">This para explains what is Cascading Style Sheet</p>
```

The title Attribute

The **title** attribute gives a suggested title for the element. They syntax for the **title** attribute is similar as explained for **id** attribute:

The behavior of this attribute will depend upon the element that carries it, although it is often displayed as a tooltip when cursor comes over the element or while the element is loading.

Example

```
<!DOCTYPE html>
<html>
<head>
<title>The title Attribute Example</title>
</head>
<body>
<h3 title="Hello HTML!">Titled Heading Tag Example</h3>
</body>
</html>
```

This will produce the following result:

Titled Heading Tag Example

Now try to bring your cursor over "Titled Heading Tag Example" and you will see that whatever title you used in your code is coming out as a tooltip of the cursor.

2.2.2 The class Attribute

The **class** attribute is used to associate an element with a style sheet, and specifies the class of element. You will learn more about the use of the class attribute when you will learn Cascading Style Sheet (CSS). So for now you can avoid it.

The value of the attribute may also be a space-separated list of class names. For example:

```
class="className1 className2 className3"
```

The style Attribute

The style attribute allows you to specify Cascading Style Sheet (CSS) rules within the element.

```
<!DOCTYPE html>
<html>
<head>
<title>The style Attribute</title>
</head>
<body>
<p style="font-family:arial; color:#FF0000;">Some text...</p>
</body>
</html>
```

This will produce the following result:

Some text...

At this point of time, we are not learning CSS, so just let's proceed without bothering much about CSS. Here, you need to understand what are HTML attributes and how they can be used while formatting content.

2.2.3 Internationalization Attributes

There are three internationalization attributes, which are available for most (although not all) XHTML elements.

- dir
- lang
- xml:lang

The dir Attribute

The **dir** attribute allows you to indicate to the browser about the direction in which the text should flow. The dir attribute can take one of two values, as you can see in the table that follows:

Value	Meaning
ltr	Left to right (the default value)
rtl	Right to left (for languages such as Hebrew or Arabic that are read right to left)

Example

```
<!DOCTYPE html>
<html dir="rtl">
<head>
<title>Display Directions</title>
</head>
<body>
This is how IE 5 renders right-to-left directed text.
</body>
```

This will produce the following result:

```
                                This is how IE 5 renders right-to-left directed text.
```

When *dir* attribute is used within the <html> tag, it determines how text will be presented within the entire document. When used within another tag, it controls the text's direction for just the content of that tag.

When *dir* attribute is used within the <html> tag, it determines how text will be presented within the entire document. When used within another tag, it controls the text's direction for just the content of that tag.

The lang Attribute

The **lang** attribute allows you to indicate the main language used in a document, but this attribute was kept in HTML only for backwards compatibility with earlier versions of HTML. This attribute has been replaced by the **xml:lang** attribute in new XHTML documents.

The values of the *lang* attribute are ISO-639 standard two-character language codes. Check **HTML Language Codes: ISO 639** for a complete list of language codes.

Example

```
<!DOCTYPE html>
<html lang="en">
<head>
<title>English Language Page</title>
</head>
<body>
This page is using English Language
</body>
</html>
```

The xml:lang Attribute

The *xml:lang* attribute is the XHTML replacement for the *lang* attribute. The value of the*xml:lang* attribute should be an ISO-639 country code as mentioned in previous section.

Generic Attributes

Here's a table of some other attributes that are readily usable with many of the HTML tags.

Attribute	Options	Function
align	right, left, center	Horizontally aligns tags
valign	top, middle, bottom	Vertically aligns tags within an HTML element.
bgcolor	numeric, hexidecimal, RGB values	Places a background color behind an element
background	URL	Places a background image behind an element

id	User Defined	Names an element for use with Cascading Style Sheets.
class	User Defined	Classifies an element for use with Cascading Style Sheets.
width	Numeric Value	Specifies the width of tables, images, or table cells.
height	Numeric Value	Specifies the height of tables, images, or table cells.
title	User Defined	"Pop-up" title of the elements.

We will see related examples as we will proceed to study other HTML tags. For a complete list of HTML Tags and related attributes please check reference to **HTML Tags List**.

2.4. HTML Formatting

If you use a word processor, you must be familiar with the ability to make text bold, italicized, or underlined; these are just three of the ten options available to indicate how text can appear in HTML and XHTML.

Bold Text

Anything that appears within ... element, is displayed in bold as shown below:

Example

```
<!DOCTYPE html>
<html>
<head>
<title>Bold Text Example</title>
</head>
<body>
<p>The following word uses a <b>bold</b> typeface.</p>
</body>
</html>
```

This will produce the following result:

```
The following word uses a bold typeface.
```

Italic Text

Anything that appears within **...** element is displayed in italicized as shown below:

Example

```
<!DOCTYPE html>
<html>
<head>
<title>Italic Text Example</title>
</head>
<body>
<p>The following word uses a <i>italicized</i> typeface.</p>
</body>
</html>
```

This will produce the following result:

```
The following word uses an italicized typeface.
```

Underlined Text

Anything that appears within **<u>...</u>** element, is displayed with underline as shown below:

Example

```
<!DOCTYPE html>
<html>
<head>
<title>Underlined Text Example</title>
</head>
<body>
<p>The following word uses a <u>underlined</u> typeface.</p>
</body>
</html>
```

This will produce the following result:

The following word uses an underlined typeface.

Strike Text

Anything that appears within **<strike>...</strike>** element is displayed with strikethrough, which is a thin line through the text as shown below:

Example

```
<!DOCTYPE html>
<html><head><title>Strike Text Example</title></head>
<body>
<p>The following word uses a <strike>strikethrough</strike>
typeface.</p>
</body></html>
```

This will produce the following result:

The following word uses a ~~strikethrough~~ typeface.

Monospaced Font

The content of a **<tt>...</tt>** element is written in monospaced font. Most of the fonts are known as variable-width fonts because different letters are of different widths (for example, the letter 'm' is wider than the letter 'i'). In a monospaced font, however, each letter has the same width.

Example

```
<!DOCTYPE html>
<html><head><title>Monospaced Font Example</title>
</head>
<body>
<p>The following word uses a <tt>monospaced</tt> typeface.</p>
</body></html>
```

This will produce the following result:

```
The following word uses a monospaced typeface.
```

Superscript Text

The content of a **^{...}** element is written in superscript; the font size used is the same size as the characters surrounding it but is displayed half a character's height above the other characters.

Example

```
<!DOCTYPE html>
<html>
<head>
<title>Superscript Text Example</title>
</head>
<body>
<p>The following word uses a <sup>superscript</sup> typeface.</p>
</body>
</html>
```

This will produce the following result:

```
The following word uses a superscript typeface.
```

Subscript Text

The content of a **_{...}** element is written in subscript; the font size used is the same as the characters surrounding it, but is displayed half a character's height beneath the other characters.

Example

```
<!DOCTYPE html>
<html><head><title>Subscript Text Example</title></head>
<body>
<p>The following word uses a <sub>subscript</sub> typeface.</p>
</body>
</html>
```

This will produce the following result:

The following word uses a $_{\text{subscript}}$ typeface.

Inserted Text

Anything that appears within **<ins>...</ins>** element is displayed as inserted text.

Example

```
<!DOCTYPE html>
<html>
<head>
<title>Inserted Text Example</title>
</head>
<body>
<p>I want to drink <del>cola</del> <ins>wine</ins></p>
</body>
</html>
```

This will produce the following result:

I want to drink ~~cola~~ wine

Deleted Text

Anything that appears within **~~...~~** element, is displayed as deleted text.

Example

```
<!DOCTYPE html>
<html>
<head>
<title>Deleted Text Example</title>
</head>
<body>
<p>I want to drink <del>cola</del> <ins>wine</ins></p>
</body>
</html>
```

This will produce the following result:

I want to drink ~~cola~~ wine

Larger Text

The content of the **<big>...</big>** element is displayed one font size larger than the rest of the text surrounding it as shown below:

Example

```
<!DOCTYPE html>
<html>
<head>
<title>Larger Text Example</title>
</head>
<body>
<p>The following word uses a <big>big</big> typeface.</p>
</body>
</html>
```

This will produce the following result:

```
The following word uses a big typeface.
```

Smaller Text

The content of the **<small>...</small>** element is displayed one font size smaller than the rest of the text surrounding it as shown below:

Example

```
<!DOCTYPE html>
<html>
<head>
<title>Smaller Text Example</title>
</head>
<body>
<p>The following word uses a <small>small</small> typeface.</p>
</body>
</html>
```

This will produce the following result:

```
The following word uses a small typeface.
```

Grouping Content

The ** ** and **** elements allow you to group together several elements to create sections or subsections of a page.

For example, you might want to put all of the footnotes on a page within a element to indicate that all of the elements within that element relate to the footnotes. You might then attach a style to this element so that they appear using a special set of style rules.

Example

```
<!DOCTYPE html>
<html>
<head>
<title>Div Tag Example</title>
</head>
<body>
<div id="menu" align="middle" >
<a href="/index.htm">HOME</a> |
<a href="/about/contact_us.htm">CONTACT</a> |
<a href="/about/index.htm">ABOUT</a>
</div>

<div id="content" align="left" bgcolor="white">
<h5>Content Articles</h5>
<p>Actual content goes here</p>
</div>
</body>
</html>
```

This will produce the following result:

HOME | CONTACT | ABOUT

CONTENT ARTICLES

Actual content goes here.....

The element, on the other hand, can be used to group inline elements only. So, if you have a part of a sentence or paragraph which you want to group together, you could use the element as follows

Example

```
<!DOCTYPE html>
<html>
<head>
<title>Span Tag Example</title>
</head>
<body>
<p>This is the example of <span style="color:green">span tag</span>
and the <span style="color:red">div tag</span> alongwith CSS</p>
</body>
</html>
```

This will produce the following result:

This is the example of span tag and the div tag along with CSS

These tags are commonly used with CSS to allow you to attach a style to a section of a page.

2.5. HTML Phrase tags

The phrase tags have been desicolgned for specific purposes, though they are displayed in a similar way as other basic tags like **<b>, <i>, <pre>,** and **<tt>,** you have seen in previous chapter. This chapter will take you through all the important phrase tags, so let's start seeing them one by one.

Emphasized Text

Anything that appears within **...** element is displayed as emphasized text.

Example

```
<!DOCTYPE html>
<html>
<head>
<title>Emphasized Text Example</title>
</head>
<body>
<p>The following word uses a <em>emphasized</em> typeface.</p>
</body>
</html>
```

This will produce the following result:

The following word uses an *emphasized* typeface.

Marked Text

Anything that appears with-in **<mark>...</mark>** element, is displayed as marked with yellow ink.

Example

```
<!DOCTYPE html>
<html>
<head>
<title>Marked Text Example</title>
</head>
<body>
<p>The following word has been <mark>marked</mark> with yellow</p>
</body>
</html>
```

Strong Text

Anything that appears within **...** element is displayed as important text.

Example

```
<!DOCTYPE html>
<html>
<head>
<title>Strong Text Example</title>
</head>
<body>
<p>The following word uses a <strong>strong</strong> typeface.</p>
</body>
</html>
```

This will produce the following result:

The following word uses a **strong** typeface.

Text Abbreviation

You can abbreviate a text by putting it inside opening <abbr> and closing </abbr> tags. If present, the title attribute must contain this full description and nothing else.

```
<!DOCTYPE html>
<html><head><title>Text Abbreviation</title>
</head><body>
<p>My best friend's name is  <abbr title="Abhishek">Abhy</abbr>.</p>
</body></html>
```

Example

This will produce the following result:

My best friend's name is Abhy.

Acronym Element

The **<acronym>** element allows you to indicate that the text between <acronym> and </acronym> tags is an acronym.

At present, the major browsers do not change the appearance of the content of the <acronym> element.

Example

```
<!DOCTYPE html>
<html><head><title>Acronym Example</title>
</head><body>
<p>This chapter covers marking up text in
<acronym>XHTML</acronym>.</p>
</body></html>
```

This will produce the following result:

This chapter covers marking up text in XHTML.

Text Direction

The **<bdo>...</bdo>** element stands for Bi-Directional Override and it is used to override the current text direction.

Example

```
<!DOCTYPE html>
<html>
<head>
<title>Text Direction Example</title>
</head>
<body>
<p>This text will go left to right.</p>
<p><bdo dir="rtl">This text will go right to left.</bdo></p>
</body>
```

This will produce the following result:

This text will go left to right.

This text will go right to left.

Special Terms

The **<dfn>...</dfn>** element (or HTML Definition Element) allows you to specify that you are introducing a special term. It's usage is similar to italic words in the midst of a paragraph.

Typically, you would use the <dfn> element the first time you introduce a key term. Most recent browsers render the content of a <dfn> element in an italic font.

Example

```
<!DOCTYPE html>
<html>
<head>
<title>Special Terms Example</title>
</head>
<body>
<p>The following word is a <dfn>special</dfn> term.</p>
</body>
</html>
```

This will produce the following result:

The following word is a *special* term.

Quoting Text

When you want to quote a passage from another source, you should put it in between**<blockquote>...</blockquote>** tags.

Text inside a <blockquote> element is usually indented from the left and right edges of the surrounding text, and sometimes uses an italicized font.

Example

This will produce the following result:

```
<!DOCTYPE html>
<html>
<head>
<title>Blockquote Example</title>
</head>
<body>
<p>The following description of XHTML is taken from the W3C Web site:</p>

<blockquote>XHTML 1.0 is the W3C's first Recommendation for XHTML, following on from earlier work on HTML 4.01, HTML 4.0, HTML 3.2 and HTML 2.0.</blockquote>
</body>
</html>
```

The following description of XHTML is taken from the W3C Web site:

> XHTML 1.0 is the W3C's first Recommendation for XHTML, following on from earlier work on HTML 4.01, HTML 4.0, HTML 3.2 and HTML 2.0.

Short Quotations

The **<q>...</q>** element is used when you want to add a double quote within a sentence.

Example

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

```
<title>Double Quote Example</title>
</head>
<body>
<p>Amit is in Spain, <q>I think I am wrong</q>.</p>
</body>
</html>
```

This will produce the following result:

Amit is in Spain, I think I am wrong.

Text Citations

If you are quoting a text, you can indicate the source placing it between an opening **<cite>**tag and closing **</cite>** tag

As you would expect in a print publication, the content of the <cite> element is rendered in italicized text by default.

Example

```
<!DOCTYPE html>
<html>
<head>
<title>Citations Example</title>
</head>
<body>
<p>This HTML tutorial is derived from <cite>W3 Standard for
HTML</cite>.</p>
</body>
</html>
```

This will produce the following result:

This HTML tutorial is derived from *W3 Standard for HTML*.

Computer Code

Any programming code to appear on a Web page should be placed inside **<code>...</code>**tags. Usually the content of the <code> element is presented in a monospaced font, just like the code in most programming books.

Example

```
<!DOCTYPE html>
<html>
<head>
<title>Computer Code Example</title>
</head>
<body>
<p>Regular text. <code>This is code.</code> Regular text.</p>
</body>
</html>
```

This will produce the following result:

Regular text. This is code.Regular text.

Keyboard Text

When you are talking about computers, if you want to tell a reader to enter some text, you can use the **<kbd>...</kbd>** element to indicate what should be typed in, as in this

example.

Example

```
<!DOCTYPE html>
<html>
<head>
<title>Keyboard Text Example</title>
</head>
<body>
<p>Regular text. <kbd>This is inside kbd element</kbd> Regular
text.</p>
</body>
</html>
```

This will produce the following result:

Regular text. This is inside kbd elementRegular text.

Programming Variables

This element is usually used in conjunction with the **<pre>** and **<code>** elements to indicate that the content of that element is a variable.

Example

```
<!DOCTYPE html>
<html>
<head>
<title>Variable Text Example</title>
</head>
<body>
<p><code>document.write("<var>user-name</var>")</code></p>
</body>
</html>
```

This will produce the following result:

document.write("*user-name*")

Program Output

The **<samp>...</samp>** element indicates sample output from a program, and script etc. Again, it is mainly used when documenting programming or coding concepts.

Example

```
<!DOCTYPE html>
<html><head><title>Program Output Example</title>
</head><body>
<p>Result produced by the program is <samp>Hello World!</samp></p>
</body></html>
```

This will produce the following result:

Result produced by the program is Hello World!

Address Text

The **<address>...</address>** element is used to contain any address.

Example

```
<!DOCTYPE html>
<html>
<head>
<title>Address Example</title>
</head>
<body>
<address>388A, Road No 22, Jubilee Hills -  Hyderabad</address>
</body>
</html>
```

This will produce the following result:

388A, Road No 22, Jubilee Hills - Hyderabad

2.6. HTML Meta tags

HTML lets you specify metadata - additional important information about a document in a variety of ways. The META elements can be used to include name/value pairs describing properties of the HTML document, such as author, expiry date, a list of keywords, document author etc.

The **<meta>** tag is used to provide such additional information. This tag is an empty element and so does not have a closing tag but it carries information within its attributes.

You can include one or more meta tags in your document based on what information you want to keep in your document but in general, meta tags do not impact physical appearance of the document so from appearance point of view, it does not matter if you include them or not.

Adding Meta Tags to Your Documents

You can add metadata to your web pages by placing <meta> tags inside the header of the document which is represented by **<head>** and **</head>** tags. A meta tag can have following attributes in addition to core attributes:

Attribute	Description
Name	Name for the property. Can be anything. Examples include, keywords, description, author, revised, generator etc.
content	Specifies the property's value.
scheme	Specifies a scheme to interpret the property's value (as declared in the content attribute).

http-equiv	Used for http response message headers. For example, http-equiv can be used to refresh the page or to set a cookie. Values include content-type, expires, refresh and set-cookie.

Specifying Keywords

You can use <meta> tag to specify important keywords related to the document and later these keywords are used by the search engines while indexing your webpage for searching purpose.

Example

Following is an example, where we are adding HTML, Meta Tags, Metadata as important keywords about the document.

```
<!DOCTYPE html>
<html>
<head>
<title>Meta Tags Example</title>
<meta name="keywords" content="HTML, Meta Tags, Metadata" />
</head>
<body>
<p>Hello HTML5!</p>
</body>
</html>
```

This will produce the following result:

```
Hello HTML5!
```

Document Description

You can use <meta> tag to give a short description about the document. This again can be used by various search engines while indexing your webpage for searching purpose.

Example

```
<!DOCTYPE html>
<html>
<head>
<title>Meta Tags Example</title>
<meta name="keywords" content="HTML, Meta Tags, Metadata" />
<meta name="description" content="Learning about Meta Tags." />
</head>
<body>
<p>Hello HTML5!</p>
</body>
</html>
```

Document Revision Date

You can use <meta> tag to give information about when last time the document was updated. This information can be used by various web browsers while refreshing your webpage.

Example

```
<!DOCTYPE html>
<html><head><title>Meta Tags Example</title>
<meta name="keywords" content="HTML, Meta Tags, Metadata" />
<meta name="description" content="Learning about Meta Tags." />
<meta name="revised" content="Tutorialspoint, 3/7/2014" />
</head>
<body><p>Hello HTML5!</p></body></html>
```

Document Refreshing

A <meta> tag can be used to specify a duration after which your web page will keep refreshing automatically.

Example

If you want your page keep refreshing after every 5 seconds then use the following syntax.

```
<!DOCTYPE html>
<html>
<head>
<title>Meta Tags Example</title>
<meta name="keywords" content="HTML, Meta Tags, Metadata" />
<meta name="description" content="Learning about Meta Tags." />
<meta name="revised" content="Tutorialspoint, 3/7/2014" />
<meta http-equiv="refresh" content="5" />
</head>
<body>
<p>Hello HTML5!</p>
</body>
</html>
```

Page Redirection

You can use <meta> tag to redirect your page to any other webpage. You can also specify a duration if you want to redirect the page after a certain number of seconds.

Example

Following is an example of redirecting current page to another page after 5 seconds. If you want to redirect page immediately then do not specify *content* attribute.

```
<!DOCTYPE html>
<html>
<head>
<title>Meta Tags Example</title>
<meta name="keywords" content="HTML, Meta Tags, Metadata" />
<meta name="description" content="Learning about Meta Tags." />
<meta name="revised" content="Tutorialspoint, 3/7/2014" />
<meta http-equiv="refresh" content="5;
url=http://www.tutorialspoint.com" />
</head>
<body>
<p>Hello HTML5!</p>
</body>
</html>
```

Setting Cookies

Cookies are data, stored in small text files on your computer and it is exchanged between web browser and web server to keep track of various information based on your web application need.

You can use <meta> tag to store cookies on client side and later this information can be used by the Web Server to track a site visitor.

Example

Following is an example of redirecting current page to another page after 5 seconds. If you want to redirect page immediately then do not specify *content* attribute.

```
<!DOCTYPE html>
<html>
<head>
<title>Meta Tags Example</title>
<meta name="keywords" content="HTML, Meta Tags, Metadata" />
<meta name="description" content="Learning about Meta Tags." />
<meta name="revised" content="Tutorialspoint, 3/7/2014" />
</head>
<body>
<p>Hello HTML5!</p>
</body>
</html>
```

Summary :

1. HTML Element: Tag vs. Element

- **Tag:** A tag is a piece of HTML code enclosed in angle brackets (e.g., ,) used to define the start and end of an HTML element.
- **Element:** An HTML element consists of a start tag, content, and an end tag (e.g., This is a paragraph.). It represents a distinct piece of content or functionality in the HTML document.

2. HTML Attributes:

- **Core Attributes:** Fundamental attributes available for most HTML elements. Examples include:
  - id: Specifies a unique identifier for an element.
  - class: Assigns one or more class names to an element for styling or scripting.
  - style: Applies inline CSS styles to an element.
  - title: Provides additional information about an element, typically displayed as a tooltip.
- **Class Attributes:** Used to define one or more class names for an element, allowing for grouping and styling multiple elements with the same class.
- **Internationalization Attributes:** Attributes designed to support multiple languages and regional settings. Examples include:
  - lang: Specifies the language of the element's content (e.g.,).
  - dir: Defines the text direction (e.g., ltr for left-to-right, rtl for right-to-left).

3. HTML Formatting:

- **Bold:** <b> - Makes text bold.
- **Italic:** <i> - Italicizes text.
- **Underlined:** <u> - Underlines text.
- **Strike:** <s> or <del> - Strikes through text (e.g., <s>strikethrough</s>).
- **Monospaced:** <code> - Displays text in a monospaced font.
- **Superscript:** <sup> - Renders text as superscript (e.g., H²O).
- **Subscript:** <sub> - Renders text as subscript (e.g., H₂O).

- **Inserted:** <ins> - Marks inserted text, often underlined.
- **Deleted:** <del> - Marks deleted text, often struck through.
- **Large:** <big> - Increases text size.
- **Smaller:** <small> - Decreases text size.
- **Grouping Content:** - A block-level container for grouping content.
- **Inline Grouping:** - An inline container for grouping small pieces of content.

4. HTML Phrase Tags:

- **Emphasized:** - Emphasizes text (typically rendered as italic).
- **Marked:** <mark> - Highlights text.
- **Strong:** - Indicates strong importance (typically rendered as bold).
- **Text Abbreviation:** <abbr> - Represents an abbreviation or acronym.
- **Acronym Element:** <acronym> - Used for acronyms (Note: deprecated in HTML5, use <abbr> instead).
- **Text Direction:** dir attribute - Defines the text direction.
- **Special Terms:** <dfn> - Marks a term being defined.
- **Quoting Text:** <q> - Defines a short inline quotation.
- **Short Quotations:** <blockquote> - Defines a longer block quotation.
- **Text Citations:** <cite> - Represents the title of a cited work.
- **Computer Code:** <code> - Displays a single line of code.
- **Keyboard Text:** <kbd> - Represents user input from a keyboard.
- **Programming Variables:** <var> - Represents a variable in a program.
- **Address Text:** <address> - Contains contact information.

5. HTML Meta Tags:

- **Purpose:** Provide metadata about the HTML document, such as character set, author, and viewport settings.
- **Examples:**
  - <meta charset="UTF-8"> - Specifies the character encoding.
  - <meta name="description" content="A brief description of the page"> - Provides a description of the page content for search engines.

- <meta name="viewport" content="width=device-width, initial-scale=1.0"> - Controls the layout on mobile browsers.

Glossary:

1) **HTML Tag:** A piece of HTML code enclosed in angle brackets (e.g., ,) used to define the start and end of an HTML element.
2) **HTML Element:** A combination of a start tag, content, and an end tag that represents a piece of content or functionality within an HTML document (e.g., This is a paragraph.).
3) **Core Attributes:** Fundamental attributes available for most HTML elements. Examples include:
   a) **id:** A unique identifier for an element.
   b) **class:** Specifies one or more class names for an element.
   c) **style:** Applies inline CSS styles to an element.
   d) **title:** Provides additional information about an element, typically displayed as a tooltip.
4) **Class Attribute:** An attribute that assigns one or more class names to an HTML element for styling or scripting purposes.
5) **Internationalization Attributes:** Attributes designed to support different languages and regional settings. Examples include:
   a) **lang:** Specifies the language of an element's content.
   b) **dir:** Defines the text direction (e.g., ltr for left-to-right, rtl for right-to-left).
6) **Bold (<b>):** An HTML tag used to make text bold.
7) **Italic (<i>):** An HTML tag used to italicize text.
8) **Underlined (<u>):** An HTML tag used to underline text.
9) **Strike (<s> or <del>):** HTML tags used to apply a strikethrough effect to text.
10) **Monospaced (<code>):** An HTML tag used to display text in a monospaced font, typically for code snippets.
11) **Superscript (<sup>):** An HTML tag used to render text as superscript (e.g., H²O).

12) **Subscript (<sub>):** An HTML tag used to render text as subscript (e.g., H₂O).
13) **Inserted (<ins>):** An HTML tag used to mark inserted text, often rendered with an underline.
14) **Deleted (<del>):** An HTML tag used to mark deleted text, often rendered with a strikethrough.
15) **Large (<big>):** An HTML tag used to increase the text size.
16) **Smaller (<small>):** An HTML tag used to decrease the text size.
17) **Grouping Content ( ):** A block-level container element used to group and style sections of content.
18) **Inline Grouping ():** An inline container element used to group and style small pieces of content.
19) **Emphasized ():** An HTML tag used to emphasize text, typically rendered as italic.
20) **Marked (<mark>):** An HTML tag used to highlight text.
21) **Strong ():** An HTML tag used to indicate strong importance, typically rendered as bold.
22) **Text Abbreviation (<abbr>):** An HTML tag used to represent an abbreviation or acronym.
23) **Acronym Element (<acronym>):** An HTML tag used for acronyms (deprecated in HTML5; use <abbr> instead).
24) **Text Direction (dir attribute):** An attribute that defines the text direction, with possible values like ltr (left-to-right) and rtl (right-to-left).
25) **Special Terms (<dfn>):** An HTML tag used to mark a term being defined.
26) **Quoting Text (<q>):** An HTML tag used to define a short inline quotation.
27) **Block Quotations (<blockquote>):** An HTML tag used to define a longer block quotation.
28) **Text Citations (<cite>):** An HTML tag used to represent the title of a cited work.
29) **Computer Code (<code>):** An HTML tag used to display a single line of computer code.
30) **Keyboard Text (<kbd>):** An HTML tag used to represent user input from a keyboard.

31) **Programming Variables (<var>):** An HTML tag used to represent a variable in a program.
32) **Address Text (<address>):** An HTML tag used to contain contact information.
33) **Meta Tags:** HTML tags used to provide metadata about the HTML document, such as character encoding, document description, and viewport settings. Examples include:
    a) **<meta charset="UTF-8">:** Specifies the character encoding.
    b) **<meta name="description" content="...">:** Provides a description of the page content.
    c) **<meta name="viewport" content="width=device-width, initial-scale=1.0">:** Controls layout on mobile browsers.

Self Assessment Questions:

1. HTML Elements: Tag vs. Element

- What is the difference between an HTML tag and an HTML element?
- Provide an example of an HTML element and describe its components.

2. HTML Attributes

- What are core attributes in HTML? List and describe at least three core attributes.
- Explain the purpose of the class attribute. How can it be used in conjunction with CSS?
- What is the lang attribute used for? Provide an example of how it might be applied.
- Describe the dir attribute and its possible values. How does it affect text display?

3. HTML Formatting Tags

- How does the <b> tag differ from the tag? Provide examples of when each would be used.
- What is the purpose of the <code> tag? How is it different from the <pre> tag?
- Explain the use of the <sub> and <sup> tags. Provide an example of each.
- How do the <ins> and <del> tags affect the text they enclose?

4. HTML Phrase Tags

- What is the difference between the and <i> tags? Provide examples of their typical usage.
- Describe the purpose of the <mark> tag. When would you use it in a webpage?
- How does the <abbr> tag function, and what is its use case?
- Explain the use of the <blockquote> tag. How is it different from the <q> tag?

5. HTML Meta Tags

- What does the <meta charset="UTF-8"> tag specify, and why is it important?
- How does the <meta name="viewport" content="width=device-width, initial-scale=1.0"> tag affect the display of a webpage on mobile devices?
- Provide an example of how the <meta name="description" content="..."> tag is used and explain its significance.

6. Practical Application

- Create a simple HTML snippet that includes text formatted with bold, italic, and underline styles. Use the appropriate HTML tags.
- Write an HTML snippet that demonstrates the use of the , , and <mark> tags to emphasize and highlight text.
- Develop a basic HTML page that includes an <address> tag with contact information and a <code> tag displaying a line of code.
- Write a meta tag that sets the character encoding to UTF-8 and another that describes the content of the page for search engines.

7. Understanding and Application

- How would you use the class attribute to apply a common style to multiple HTML elements?
- Explain how the tag differs from the tag and provide examples of when you would use each.
- If you needed to include a short quotation within a paragraph, which tag would you use and why?

8. Problem Solving

- Given an HTML document with text that needs to be formatted in various styles (bold, italic, and monospaced), how would you write the necessary HTML to achieve this?
- How would you ensure that your webpage displays correctly across different languages and text directions?

Activities and Exercises:

1. **Creating a Styled Document:**
    - **Objective:** Practice using HTML tags for text formatting.
    - **Instructions:** Create an HTML document that includes:
        - A heading with bold and italic text.
        - A paragraph with some text formatted as superscript, subscript, and monospaced.
        - Another paragraph with text that is underlined, struck through, and highlighted.
    - **Deliverable:** A complete HTML file demonstrating the use of formatting tags.
2. **Implementing HTML Attributes:**
    - **Objective:** Apply HTML attributes to style and identify elements.
    - **Instructions:** Write an HTML snippet that includes:
        - A `div` element with a `class` attribute applied for styling with CSS.
        - An `img` element with `src`, `alt`, and `title` attributes.
        - A `p` element with `lang` and `dir` attributes.
    - **Deliverable:** A sample HTML file showcasing the use of different attributes and their effects.
3. **Using Meta Tags for SEO and Responsiveness:**
    - **Objective:** Learn to use meta tags to improve SEO and responsiveness.
    - **Instructions:** Create an HTML document that includes:
        - Meta tags for character encoding, viewport settings, and description.

- A title tag and a meta tag for keywords.

- **Deliverable:** An HTML file with proper meta tags for a responsive and SEO-friendly webpage.

4. **Text Quotation and Citations:**
   - **Objective:** Use HTML tags for quoting and citing text.
   - **Instructions:** Write an HTML document that includes:
     - A short quotation using the <q> tag.
     - A block quotation using the <blockquote> tag.
     - A citation with the <cite> tag.
   - **Deliverable:** A well-formatted HTML file demonstrating the use of quoting and citation tags.
5. **Formatting a Personal Bio Page:**
   - **Objective:** Apply various formatting and phrasing tags.
   - **Instructions:** Create a personal bio page that includes:
     - A heading, subheading, and a paragraph with various text formatting (bold, italic, etc.).
     - Emphasized and strong text, marked text, and definitions.
     - An address section with contact information.
   - **Deliverable:** An HTML file representing a styled personal biography.

Case Studies:

1. **Case Study: Developing a Corporate Webpage:**
   - **Objective:** Apply HTML formatting, attributes, and meta tags in a real-world scenario.
   - **Instructions:** Design a corporate webpage that includes:
     - A header section with navigation links.
     - Main content sections with formatted text, lists, and highlighted key information.
     - A footer with contact details, using the <address> tag.
     - Proper meta tags for SEO and responsive design.

- **Deliverable:** A complete HTML webpage that meets the specified requirements.

2. **Case Study: Localization and Internationalization:**
   - **Objective:** Implement internationalization attributes in a multi-language webpage.
   - **Instructions:** Create an HTML document with:
     - Sections in different languages using the `lang` attribute.
     - Text displayed in different directions using the `dir` attribute.
     - Include a mix of internationalized content with different text formats.
   - **Deliverable:** An HTML file showcasing effective use of language and direction attributes.
3. **Case Study: Updating Legacy HTML to Modern Standards:**
   - **Objective:** Refactor an outdated HTML document to use current standards and best practices.
   - **Instructions:** Given an outdated HTML document:
     - Update deprecated tags to their modern equivalents.
     - Implement HTML5 semantic elements (e.g., `<header>`, `<footer>`, `<article>`).
     - Add modern meta tags for responsive design and SEO.
   - **Deliverable:** An updated HTML document demonstrating modern HTML practices.
4. **Case Study: Interactive Documentation Page:**
   - **Objective:** Create an interactive documentation page with proper text formatting and citations.
   - **Instructions:** Develop a documentation page that includes:
     - A table of contents with internal links.
     - Sections with inline and block quotations.
     - Code snippets formatted with the `<code>` tag.
     - References and citations using appropriate HTML tags.
   - **Deliverable:** An HTML file representing a well-structured and interactive documentation page.

UNIT III

UNIT 3 - Lists
Types of lists: Ordered, Unordered– Nesting Lists– Other tags: Marquee, HR, BR. Links: Hyperlinks – HTML links syntax – HTML links Target attribute – Absolute URLs Vs Relative URLs, HTML links Using Images.

TABLE OF CONTENTS

UNIT OBJECTIVES:

In this unit, learners will be capable to use ordered, unordered lists and Nesting lists in creating the webpages. Learners will be able to create a web page using hr and br commands and develop a projects which will be very much helpful for our society.

3.1. Lists

What is list?

In the context of HTML and web development, a **list** refers to a way of organizing and presenting information in a structured format. Lists are fundamental elements used to group related items together, making content more organized and easier to read for users.

3.2 What are the types of lists?

HTML provides several types of lists:

1. **Ordered List (`<ol>`)**: Represents a list of items where each item is sequentially numbered. It is typically used when the order of items is important.

Example:

```html
<ol>
  <li>First item</li>
  <li>Second item</li>
  <li>Third item</li>
</ol>
```

Renders as:

1. First item
2. Second item

3. Third item

2. **Unordered List (`<ul>`)**: Represents a list of items where each item is preceded by a bullet point. It is used when the order of items is not important.

Example:

```html
<ul>
  <li>Item A</li>
  <li>Item B</li>
  <li>Item C</li>
</ul>
```

Renders as:

- Item A
- Item B
- Item C

3. **Definition List (`<dl>`)**: Represents a list of terms and their definitions. It consists of `<dt>` (definition term) and `<dd>` (definition description) pairs.

Example:

```html
<dl>
  <dt>Term 1</dt>
  <dd>Definition 1</dd>
  <dt>Term 2</dt>
  <dd>Definition 2</dd>
</dl>
```

Renders as:

```
Term 1
Definition 1
Term 2
Definition 2
```

Lists in HTML are versatile and can be nested within each other to create more complex structures. They are commonly used for navigation menus, content summaries, itemized information, and other structured data presentations on web pages. The choice between ordered, unordered, or definition lists depends on how you want to organize and present your content to users.

HTML lists allow web developers to group a set of related items in lists.

Example

1. An ordered HTML list:

I. First item
II. Second item
III. Third item
IV. Fourth item

2. An unordered HTML list:

I. Item
II. Item
III. Item
IV. Item

3.2.1 Ordered HTML List

An ordered list starts with the <ol> tag. Each list item starts with the <li> tag.

The list items will be marked with numbers by default:

Example

```
<ol>
  <li>Coffee</li>
  <li>Tea</li>
  <li>Milk</li>
</ol>
```

3.2.2 Unordered HTML List

An unordered list starts with the <ul> tag. Each list item starts with the <li> tag.

The list items will be marked with bullets (small black circles) by default:

Example

```
<ul>
  <li>Coffee</li>
  <li>Tea</li>
  <li>Milk</li>
</ul>
```

HTML Description Lists

HTML also supports description lists.

A description list is a list of terms, with a description of each term.

The <dl> tag defines the description list, the <dt> tag defines the term (name), and the <dd> tag describes each term:

Example

```
<dl>
  <dt>Coffee</dt>
  <dd>- black hot drink</dd>
  <dt>Milk</dt>
  <dd>- white cold drink</dd>
</dl>
```

Exercises

HTML List Tags

Tag	Description
	Defines an unordered list
	Defines an ordered list
	Defines a list item
<dl>	Defines a description list
<dt>	Defines a term in a description list
<dd>	Describes the term in a description list

3.3. Nested list in HTML

A nested list in HTML is a list that contains other lists within its list items. This creates a hierarchical structure, where each sublist is indented to visually represent its relationship to the parent list item. It's commonly used for organizing and presenting information in a structured manner, enhancing readability and clarity of content.

Table of Content

Nested Unordered List in HTML

An unordered list in HTML is a collection of items represented by bullet points, providing a flexible way to display information without a specific order.

Approach:

- Use the <ul> tag to create an unordered list.

- Utilize the <li> tag to list individual items within the <ul> tag.
- The <ul> tag serves as the parent container for <li> tags, establishing the structure of the list.
- Nested lists can be created by placing additional <ul> or <ol> tags within <li> tags, enabling the creation of hierarchical structures.

Example: Implementation to create a nested unordered list.

- HTML

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport"
        content="width=device-width, initial-scale=1.0">
    <title>Nested Unordered List</title>
    <style>


    </style>
</head>

<body>
    <ul>
        <li>
            <h2>Frontend</h2>
            <ul>
                <li>HTML</li>
                <li>CSS
                    <ul>
                        <li>onsubmit Attribute</li>
                        <li>onclick Attribute</li>
                    </ul>
                </li>
```

```
            <li>JavaScript</li>
          </ul>
        </li>
        <li>
          <h2>Library/Framework</h2>
          <ul>
            <li>ReactJS
              <ul>
                <li>Hoisting</li>
                <li>Props</li>
              </ul>
            </li>
          </ul>
        </li>
      </ul>
</body>

</html>
```

Output:

- **Frontend**
  - HTML
  - CSS
    - onsubmit Attribute
    - onclick Attribute
  - JavaScript
- **Library/Framework**
  - ReactJS
    - Hoisting
    - Props

Nested Ordered List in HTML

In HTML, an ordered list organizes items in a numbered sequence, providing a structured way to present information.

Approach:

- Use the <ol> tag to create an ordered list.
- Employ the <li> tag to list individual items within the <ol> tag, which are automatically numbered.
- The <ol> tag acts as the parent container for <li> tags, defining the sequential order of the list.
- To create nested ordered lists, nest additional <ol> tags within <li> tags, facilitating the creation of hierarchical lists.

Example: Implementation to create a nested ordered list.

- HTML

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport"
        content="width=device-width, initial-scale=1.0">
    <title>Nested Ordered List</title>
    <style>

    </style>
</head>

<body>
```

```
<ol>
    <li>Subjects</li>
    <ol>
        <li>Mathematics</li>
        <li>Science</li>
        <li>Literature</li>
    </ol>
    <li>Programming Languages</li>
    <ol>
        <li>C</li>
        <li>Java</li>
        <li>Python</li>
        <li>Ruby</li>
        <li>Swift</li>
    </ol>
    <li>Computer Science Concepts</li>
    <ol>
        <li>Data Structures</li>
        <li>Algorithms</li>
        <li>Operating Systems</li>
        <li>Database Management</li>
        <li>Networking</li>
    </ol>
    <li>Web Technologies</li>
    <ol>
        <li>HTML</li>
        <li>CSS</li>
        <li>JavaScript</li>
        <ol>
```

```
            <li>React</li>
            <li>Angular</li>
            <li>Vue</li>
          </ol>
          <li>Bootstrap</li>
        </ol>
    </ol>

</body>

</html>
```

Output:

1. Subjects
    1. Mathematics
    2. Science
    3. Literature
2. Programming Languages
    1. C
    2. Java
    3. Python
    4. Ruby
    5. Swift
3. Computer Science Concepts
    1. Data Structures
    2. Algorithms
    3. Operating Systems
    4. Database Management
    5. Networking
4. Web Technologies
    1. HTML
    2. CSS
    3. JavaScript
        1. React
        2. Angular
        3. Vue
    4. Bootstrap

3.4. OTHER TAGS:

HTML <marquee> tag

The **<marquee> tag in HTML** creates a scrolling text or image effect within a webpage. It allows content to move horizontally or vertically across the screen, providing a simple way to add dynamic movement to elements. It includes attributes like direction to specify whether the content moves left, right, up, or down.

Syntax

```
<marquee>

  <--- contents --->

</marquee>
```

Attributes :

Attributes	Values	Description
bgcolor	Color Name	Define the background color of the marquee.
direction	Top, Down, Left, Right	Define the direction of scrolling the content
loop	Number	Specifies how many times content moves. The default value is infinite.
height	px or %	Define the height of marquee
width	px or %	Define the width of marquee
hspace	px	Specify horizontal space around marquee

Attributes	Values	Description
vspace	px	Specify vertical space around marquee

Methods :

Method	Description
start()	Used to start the scrolling of the <marquee> tag.
stop()	Used to stop the scrolling of the <marquee> tag.

Event Handlers :

Event Handler	Description
onbounce	Triggered when a scrolling <marquee> reaches the end, exclusive to ‘alternate’ behavior.
onfinish	Activates when the <marquee> completes scrolling loops specified by the ‘loop’ attribute.
onstart	Fired at the initiation of scrolling for the HTML <marquee> tag.

Examples of HTML <marquee> Tag

Example 1: In this example, we will see implementation of above tag with right to left.

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

<head>
    <title>Marquee Tag</title>
    <style>
        .main {
            text-align: center;
        }

        .marq {
            padding-top: 30px;
            padding-bottom: 30px;
        }

        .geek1 {
            font-size: 36px;
            font-weight: bold;
            color: white;
            padding-bottom: 10px;
        }
    </style>
</head>

<body>
    <div class="main">
        <marquee class="marq" bgcolor="Green"
                 direction="left" loop="">
            <div class="geek1">
                GeeksforGeeks
```

```
            </div>
            <div class="geek2">
                A computer science portal for geeks
            </div>
        </marquee>
    </div>
</body>

</html>
```

HTML <hr> Tag

Example

Use the <hr> tag to define thematic changes in the content:

```
<h1>The Main Languages of the Web</h1>

<p>HTML is the standard markup language for creating Web pages. HTML describes the structure of a Web page, and consists of a series of elements. HTML elements tell the browser how to display the content.</p>

<hr>

<p>CSS is a language that describes how HTML elements are to be displayed on screen, paper, or in other media. CSS saves a lot of work, because it can control the layout of multiple web pages all at once.</p>

<hr>

<p>JavaScript is the programming language of HTML and the Web. JavaScript
```

can change HTML content and attribute values. JavaScript can change CSS. JavaScript can hide and show HTML elements, and more.</p>

Definition and Usage

The <hr> tag defines a thematic break in an HTML page (e.g. a shift of topic).

The <hr> element is most often displayed as a horizontal rule that is used to separate content (or define a change) in an HTML page.

HTML
 Tag

Example

Insert single line breaks in a text:

```
<p>To force<br> line breaks<br> in a text,<br> use the br<br> element.</p>
```

Definition and Usage

The
 tag inserts a single line break.

The
 tag is useful for writing addresses or poems.

The
 tag is an empty tag which means that it has no end tag.

Note: Use the
 tag to enter line breaks, not to add space between paragraphs.

3.5. Links

Links are found in nearly all web pages. Links allow users to click their way from page to page.

3.5.1 HTML Links - Hyperlinks

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.

3.5.2 HTML Links - Syntax

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

link text

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.

Example

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

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

By default, links will appear as follows in all browsers:

- An unvisited link is underlined and blue
- A visited link is underlined and purple
- An active link is underlined and red

3.5.3 HTML Links - The target Attribute

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

Example

Use target="_blank" to open the linked document in a new browser window or tab:

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

3.6. Absolute URLs vs. Relative URLs

Both examples above are using an **absolute URL** (a full web address) in the href attribute.

A local link (a link to a page within the same website) is specified with a **relative URL** (without the "https://www" part):

Example

```
<h2>Absolute URLs</h2>
<p><a href="https://www.w3.org/">W3C</a></p>
<p><a href="https://www.google.com/">Google</a></p>

<h2>Relative URLs</h2>
<p><a href="html_images.asp">HTML Images</a></p>
<p><a href="/css/default.asp">CSS Tutorial</a></p>
```

3.7. HTML Links - Use an Image as a Link

To use an image as a link, just put the <img> tag inside the <a> tag:

Example

```
<a href="default.asp">
<img src="smiley.gif" alt="HTML tutorial" style="width:42px;height:42px;">
</a>
```

Output:

Summary :

Lists: Types of Lists

1. **Ordered Lists (<ol>):**
   - **Description:** Used to create a list of items where the order is significant. Each item in the list is numbered automatically.

- **Example:**

```
html
Copy code
<ol>
  <li>First item</li>
  <li>Second item</li>
  <li>Third item</li>
</ol>
```

2. **Unordered Lists (<ul>):**
   - **Description:** Used to create a list of items where the order is not important. Each item is preceded by a bullet point.
   - **Example:**

```
html
Copy code
<ul>
  <li>Item A</li>
  <li>Item B</li>
  <li>Item C</li>
</ul>
```

3. **Nesting Lists:**
   - **Description:** Lists can be nested within each other to create hierarchical structures. An ordered list can contain an unordered list, and vice versa.
   - **Example:**

```
html
Copy code
<ul>
  <li>Parent item
    <ul>
      <li>Child item 1</li>
      <li>Child item 2</li>
    </ul>
  </li>
</ul>
```

Other Tags

1. **Marquee (<marquee>):**

- **Description:** Used to create a scrolling effect for text or images. It is not recommended for modern web design due to lack of support and accessibility issues.
- **Example:**

  html

  Copy code

  <marquee>Scrolling text</marquee>

2. **Horizontal Rule (<hr>):**
   - **Description:** Creates a horizontal line across the page, often used to separate content or sections.
   - **Example:**

     html

     Copy code

     <hr>
3. **Line Break (
):**
   - **Description:** Inserts a line break within text, moving subsequent content to a new line. Useful for formatting text without creating a new paragraph.
   - **Example:**

     html

     Copy code

     Line one
Line two

Links:

1. **Hyperlinks:**
   - **Description:** Links that connect users to other web pages or resources. Created using the <a> tag.
   - **Example:**

     html

     Copy code

     Visit Example
2. **HTML Links Syntax:**
   - **Description:** The href attribute within the <a> tag specifies the URL of the linked resource.
   - **Example:**

html

Copy code

```
<a href="https://www.example.com">Link Text</a>
```

3. **HTML Links Target Attribute:**
   - **Description:** Defines where the linked document will open. Common values include:
     - "_self" (default, opens in the same frame)
     - "_blank" (opens in a new tab or window)
     - "_parent" (opens in the parent frame)
     - "_top" (opens in the full body of the window)
   - **Example:**

     html

     Copy code

     ```
     <a href="https://www.example.com" target="_blank">Open in new tab</a>
     ```
4. **Absolute URLs vs. Relative URLs:**
   - **Absolute URL:** Includes the full path to the resource, including the protocol (e.g., https://).
     - **Example:** https://www.example.com/page
   - **Relative URL:** Provides a path relative to the current page or domain, omitting the protocol and domain.
     - **Example:** /page or page.html
5. **HTML Links Using Images:**
   - **Description:** Links can be created using images as clickable elements by nesting an <img> tag within an <a> tag.
   - **Example:**

     html

     Copy code

     ```
     <a href="https://www.example.com">
       <img src="logo.png" alt="Example Logo">
     </a>
     ```

Glossary :

1. **Ordered List (<ol>):**

- **Definition:** An HTML element used to create a list of items where each item is numbered in a sequence.
- **Example:** First item
Second item

2. **Unordered List (<ul>):**
   - **Definition:** An HTML element used to create a list of items where each item is marked with a bullet point rather than a number.
   - **Example:** Item A
Item B
3. **Nesting Lists:**
   - **Definition:** Placing one list inside another, allowing the creation of hierarchical or multi-level lists.
   - **Example:** Parent item
Child item
4. **Marquee (<marquee>):**
   - **Definition:** An HTML tag used to create a scrolling text or image effect. It is deprecated and not recommended for modern web design.
   - **Example:** <marquee>Scrolling text</marquee>
5. **Horizontal Rule (<hr>):**
   - **Definition:** An HTML tag that creates a horizontal line across the page, often used to separate content or sections visually.
   - **Example:** <hr>
6. **Line Break (
):**
   - **Definition:** An HTML tag that inserts a line break in the text, moving the following content to the next line without creating a new paragraph.
   - **Example:** Line one
Line two
7. **Hyperlink (<a>):**
   - **Definition:** An HTML element used to create a clickable link to another web page, document, or resource. The href attribute specifies the destination URL.
   - **Example:** Visit Example
8. **HTML Links Syntax:**
   - **Definition:** The structure used to create hyperlinks in HTML, where the href attribute defines the link's destination URL.

- **Example:** Link Text

9. **Target Attribute:**
   - **Definition:** An attribute used with the <a> tag to specify where to open the linked document. Common values include:
     - "_self" (opens in the same frame)
     - "_blank" (opens in a new tab or window)
     - "_parent" (opens in the parent frame)
     - "_top" (opens in the full body of the window)
   - **Example:** Open in new tab
10. **Absolute URL:**
    - **Definition:** A complete URL that includes the protocol (e.g., https://), domain name, and path to the resource.
    - **Example:** https://www.example.com/page
11. **Relative URL:**
    - **Definition:** A URL that provides a path relative to the current page or domain, omitting the protocol and domain.
    - **Example:** /page or page.html
12. **HTML Links Using Images:**
    - **Definition:** An HTML technique where an image is used as a clickable link by embedding an <img> tag within an <a> tag.
    - **Example:** <img src="logo.png" alt="Example Logo">

Self-Assessment Questions

1. Lists:

1.1. Ordered and Unordered Lists

- What is the primary difference between an ordered list (<ol>) and an unordered list (<ul>) in HTML?
- Write an HTML snippet that demonstrates the use of an ordered list and an unordered list in the same document.

1.2. Nesting Lists

- How can you create a nested list in HTML? Provide an example where an unordered list contains an ordered list.
- What is the visual effect of nesting lists within each other?

2. Other Tags:

2.1. Marquee

- What is the purpose of the <marquee> tag, and why is it generally not recommended for use in modern web design?
- Provide an example of how the <marquee> tag would be used, and explain an alternative method to achieve a similar effect.

2.2. Horizontal Rule

- How does the <hr> tag affect the layout of a webpage? In what scenarios might it be used?
- Write an HTML snippet that includes a horizontal rule between two paragraphs of text.

2.3. Line Break

- When should the
 tag be used instead of a new paragraph tag (<p>)?
- Provide an example of using the
 tag within a block of text.

3. Links:

3.1. Hyperlinks

- How do you create a hyperlink in HTML? What is the role of the href attribute?
- Write an HTML snippet that creates a link to "https://www.example.com" with the text "Visit Example".

3.2. HTML Links Syntax

- What is the correct syntax for creating a link to an external website?
- Explain how the href attribute works and what kind of values it can take.

3.3. Target Attribute

- What are the different values for the target attribute in an <a> tag, and what does each value do?
- Write an HTML snippet that opens a link in a new tab using the target attribute.

3.4. Absolute vs. Relative URLs

- What is the difference between an absolute URL and a relative URL? Provide examples of each.
- Given a webpage located at https://www.example.com/page, write both an absolute and a relative URL to link to a page named contact.html located in the same directory.

3.5. Links Using Images

- How can you create a clickable image that links to another webpage? Describe the HTML structure required.
- Write an HTML snippet that uses an image as a link to "https://www.example.com", with the image file named "logo.png".

4. Practical Application

4.1. Combining Lists and Links

- Create an HTML snippet that includes an unordered list of links, each linking to different external websites.
- Explain how you would style these links to ensure they are clearly distinguishable as clickable items.

4.2. Formatting with Other Tags

- How can you use the <hr> and
 tags together in a document to format content? Provide an example demonstrating their use in a single HTML document.

4.3. Linking and Navigation

- Develop a simple HTML page with a navigation bar using an unordered list of links. Ensure each link leads to a different section of the same page using anchor links.

Activities and Exercises

1. Lists:

1.1. Create Lists

- **Activity:** Build a webpage that includes both an ordered list and an unordered list. The ordered list should contain steps for a recipe, and the unordered list should include ingredients.
- **Instructions:**
    1. Use <ol> to create a numbered list for the recipe steps.
    2. Use <ul> to list the ingredients with bullet points.
    3. Include at least five items in each list.

1.2. Nested Lists

- **Exercise:** Create a webpage with a nested list. The outer list should represent a category of items, and the inner list should detail subcategories.
- **Instructions:**
    1. Use <ul> for the main category list.
    2. Inside one of the <li> items, add another <ul> to list subcategories.
    3. Ensure at least one nested level with several items.

2. Other Tags:

2.1. Marquee Tag

- **Activity:** Use the <marquee> tag to create a scrolling text effect on a webpage. Then, replace the <marquee> tag with CSS animations to achieve a similar effect.
- **Instructions:**
    1. Implement the scrolling text with <marquee>.
    2. Write equivalent CSS to create a scrolling effect using keyframes.

2.2. Horizontal Rule and Line Break

- **Exercise:** Design a webpage that uses <hr> to separate different sections of content and
 to format a block of text into multiple lines.
- **Instructions:**
    1. Add <hr> tags to separate three different sections of content.
    2. Use
 within a paragraph to break lines and format text properly.

3. Links:

3.1. Basic Hyperlink

- **Activity:** Create a webpage with several hyperlinks. Include links to external websites, internal pages, and email addresses.
- **Instructions:**
    1. Create at least three external links with different `href` values.
    2. Add one internal link that points to another page in the same directory.
    3. Include an email link that opens the default email client.

3.2. Link Target Attribute

- **Exercise:** Build a webpage with a navigation menu. Ensure that some links open in the same tab and others open in new tabs using the `target` attribute.
- **Instructions:**
    1. Create a list of links in the navigation menu.
    2. Set `target="_self"` for some links and `target="_blank"` for others.
    3. Test to ensure that links behave as expected.

3.3. Absolute vs. Relative URLs

- **Activity:** Create two versions of a link on a webpage—one using an absolute URL and one using a relative URL. Both should link to the same resource.
- **Instructions:**
    1. Use an absolute URL to link to a specific webpage.
    2. Use a relative URL to achieve the same link within the same directory.
    3. Compare the two methods and note their differences.

3.4. Image as a Link

- **Exercise:** Design a webpage where an image is used as a clickable link. The image should link to an external website.
- **Instructions:**
    1. Insert an `<img>` tag within an `<a>` tag.
    2. Set the `href` attribute to link to an external website.
    3. Ensure the `alt` attribute of the image is descriptive.

4. Case Studies

4.1. Case Study: Product Catalog

- **Description:** Develop a product catalog page for an e-commerce site. Use both ordered and unordered lists to display product categories and featured products.
- **Instructions:**
    1. Create an ordered list of product categories.
    2. For each category, use an unordered list to list featured products.
    3. Include links for each product that lead to detailed product pages.

4.2. Case Study: Event Page

- **Description:** Create a webpage for an event with a timeline of activities and a list of speakers. Use nested lists to organize the event schedule.
- **Instructions:**
    1. Design an ordered list for the event schedule, with each list item representing a time slot.
    2. Inside each time slot, use an unordered list to detail the activities or speakers.
    3. Add hyperlinks to speakers' names that link to their profiles or biographies.

4.3. Case Study: Personal Portfolio

- **Description:** Build a personal portfolio website with a navigation menu. Include sections for biography, projects, and contact information. Use links to navigate between sections and external resources.
- **Instructions:**
    1. Create a navigation menu using an unordered list.
    2. Link each menu item to different sections of the page using anchor links.
    3. Include external links to social media profiles or other relevant sites.

4.4. Case Study: Blog Post Formatting

- **Description:** Format a blog post with proper use of horizontal rules and line breaks to separate sections and enhance readability.
- **Instructions:**
    1. Write a blog post with multiple sections.
    2. Use <hr> tags to separate major sections or topics.
    3. Apply
 tags within paragraphs to break lines for better formatting.

UNIT IV

UNIT 4 - Tables
HTML Tables- Definition and Usage- Define a HTML table- Table Cells – Table rows – Table headers – HTML Table tags- HTML Tables Colspan & Rowspan – HTML Table Padding & Spacing.

TABLE OF CONTENTS

UNIT OBJECTIVES :

In this unit, learners are capable to use Tables in creating webpages. The learners will use the cell alignment for each text effectively and makes proper arrangements for the text. To make the content attractive is necessary for the online purchases websites. By doing so the learners are ready to develop a business and make more profits.

Tables

4.1. HTML Tables

HTML tables allow web developers to arrange data into rows and columns.

4.1.1. Definition and Usage

The <table> tag defines an HTML table.

An HTML table consists of one <table> element and one or more <tr>, <th>, and <td> elements.

The <tr> element defines a table row, the <th> element defines a table header, and the <td> element defines a table cell.

An HTML table may also include <caption>, <colgroup>, <thead>, <tfoot>, and <tbody> elements.

4.1.2. Define an HTML Table

A table in HTML consists of table cells inside rows and columns.

Example

A simple HTML table:

```
<table>
  <tr>
    <th>Company</th>
    <th>Contact</th>
    <th>Country</th>
```

```
  </tr>
  <tr>
    <td>Alfreds Futterkiste</td>
    <td>Maria Anders</td>
    <td>Germany</td>
  </tr>
  <tr>
    <td>Centro comercial Moctezuma</td>
    <td>Francisco Chang</td>
    <td>Mexico</td>
  </tr>
</table>
```

4.2. Table Cells

Each table cell is defined by a <td> and a </td> tag.

td stands for table data.

Everything between <td> and </td> are the content of the table cell.

Example

```
<table>
  <tr>
    <td>Emil</td>
    <td>Tobias</td>
    <td>Linus</td>
  </tr>
</table>
```

4.3.Table Rows

Each table row starts with a <tr> and ends with a </tr> tag.

tr stands for table row.

Example

```
<table>
  <tr>
    <td>Emil</td>
    <td>Tobias</td>
    <td>Linus</td>
  </tr>
  <tr>
    <td>16</td>
    <td>14</td>
    <td>10</td>
  </tr>
</table>
```

You can have as many rows as you like in a table; just make sure that the number of cells are the same in each row.

4.4. Table Headers

Sometimes you want your cells to be table header cells. In those cases use the <th> tag instead of the <td> tag:

th stands for table header.

Example

Let the first row be table header cells:

```
<table>
  <tr>
    <th>Person 1</th>
    <th>Person 2</th>
    <th>Person 3</th>
  </tr>
  <tr>
    <td>Emil</td>
    <td>Tobias</td>
    <td>Linus</td>
```

```
  </tr>
  <tr>
    <td>16</td>
    <td>14</td>
    <td>10</td>
  </tr>
</table>
```

By default, the text in <th> elements are bold and centered, but you can change that with CSS.

4.5. HTML Table Tags

Tag	Description
<table>	Defines a table
<th>	Defines a header cell in a table
<tr>	Defines a row in a table
<td>	Defines a cell in a table
<caption>	Defines a table caption
<colgroup>	Specifies a group of one or more columns in a table for formatting
<col>	Specifies column properties for each column within a <colgroup> element
<thead>	Groups the header content in a table
<tbody>	Groups the body content in a table

<tfoot>	Groups the footer content in a table

4.6. HTML Tables(row & col)

HTML tables allow web developers to arrange data into rows and columns.

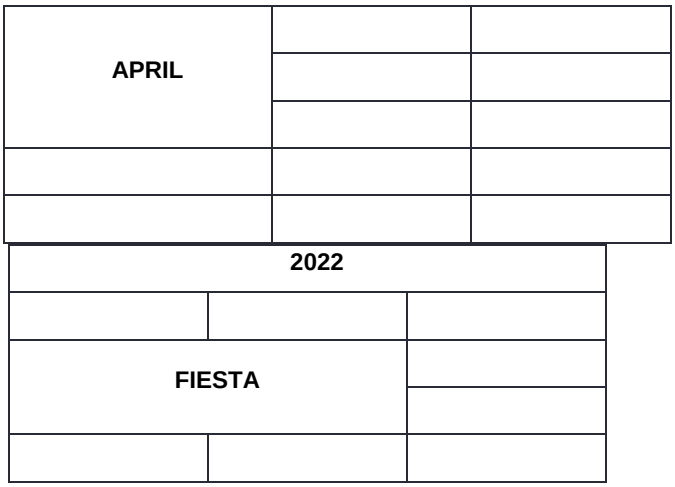
Example

Company	Contact	Country
Alfreds Futterkiste	Maria Anders	Germany
Centro comercial Moctezuma	Francisco Chang	Mexico
Ernst Handel	Roland Mendel	Austria
Island Trading	Helen Bennett	UK
Laughing Bacchus Winecellars	Yoshi Tannamuri	Canada
Magazzini Alimentari Riuniti	Giovanni Rovelli	Italy

HTML Table Colspan & Rowspan

HTML tables can have cells that span over multiple rows and/or columns.

NAME		



HTML Table - Colspan

To make a cell span over multiple columns, use the colspan attribute:

Example

```
<table>
  <tr>
    <th colspan="2">Name</th>
    <th>Age</th>
  </tr>
  <tr>
    <td>Jill</td>
    <td>Smith</td>
    <td>43</td>
  </tr>
  <tr>
    <td>Eve</td>
    <td>Jackson</td>
    <td>57</td>
  </tr>
</table>
```

Note: The value of the colspan attribute represents the number of columns to span.

HTML Table - Rowspan

To make a cell span over multiple rows, use the rowspan attribute:

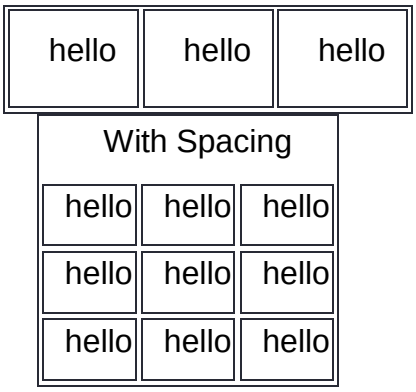
Example

```
<table>
  <tr>
    <th>Name</th>
    <td>Jill</td>
  </tr>
  <tr>
    <th rowspan="2">Phone</th>
    <td>555-1234</td>
  </tr>
  <tr>
    <td>555-8745</td>
</tr>
</table>
```

4.7. HTML Table Padding & Spacing

HTML tables can adjust the padding inside the cells, and also the space between the cells.

With Padding		
hello	hello	hello
hello	hello	hello



HTML Table - Cell Padding

Cell padding is the space between the cell edges and the cell content.

By default the padding is set to 0.

To add padding on table cells, use the CSS padding property:

Example

```
th, td {
  padding: 15px;
}
```

To add padding only above the content, use the padding-top property.

And the others sides with the padding-bottom, padding-left, and padding-right properties:

Example

```
th, td {
  padding-top: 10px;
  padding-bottom: 20px;
  padding-left: 30px;
  padding-right: 40px;
}
```

HTML Table - Cell Spacing

Cell spacing is the space between each cell.

By default the space is set to 2 pixels.

To change the space between table cells, use the CSS border-spacing property on the table element:

Example

```
table {
  border-spacing: 30px;
}
```

HTML ExercisesTop of Form

Exercise:

Use the correct CSS property to add 15 pixels of space between the cell border and cell content. The result should look like this:

:

hello	hello	hello
hello	hello	hello
hello	hello	hello

Summary:

HTML Tables: Overview

1. Definition and Usage:

- **Definition:** An HTML table is used to display data in a structured format of rows and columns. Tables are useful for presenting tabular data, such as schedules, financial reports, or any other data organized in a grid-like format.
- **Usage:** Tables are created using the <table> element, with rows and columns defined by <tr>, <td>, and <th> elements. They help in organizing data in a clear and systematic way, making it easier to compare and interpret.

2. Define an HTML Table:

- **Basic Structure:**

```
html
Copy code
<table>
  <tr>
    <th>Header 1</th>
    <th>Header 2</th>
  </tr>
  <tr>
    <td>Data 1</td>
    <td>Data 2</td>
  </tr>
</table>
```

  - **Explanation:** The <table> element contains the entire table. Inside it, <tr> elements define rows, <th> elements define header cells, and <td> elements define data cells.

3. Table Cells:

- **Table Cells (<td>):** Represent individual data points within a table. They are used inside <tr> elements to display content.
  - **Example:**

```
html
Copy code
<td>Cell Content</td>
```

4. Table Rows:

- **Table Rows (<tr>):** Group together a set of table cells in a horizontal line. Each row is a <tr> element, which contains one or more <td> or <th> elements.
  - **Example:**

```
html
Copy code
<tr>
  <td>Row 1, Cell 1</td>
  <td>Row 1, Cell 2</td>
</tr>
```

5. Table Headers:

- **Table Headers (<th>):** Define the header cells in a table, which are typically bold and centered by default. They are used to label the columns or rows.
  - **Example:**

```
html
Copy code
<th>Header 1</th>
```

6. HTML Table Tags:

- **Key Tags:**
  - <table>: The container element for the table.
  - <tr>: Defines a row in the table.
  - <td>: Defines a data cell within a row.
  - <th>: Defines a header cell within a row.

7. Colspan and Rowspan:

- **Colspan (colspan attribute):** Allows a cell to span across multiple columns.
  - **Example:**

```
html
Copy code
<td colspan="2">Spans 2 columns</td>
```

- **Rowspan (rowspan attribute):** Allows a cell to span across multiple rows.
  - **Example:**

```
html
Copy code
<td rowspan="2">Spans 2 rows</td>
```

8. Table Padding and Spacing:

- **Padding:** Refers to the space inside the cell between the cell border and its content. In HTML, padding can be controlled using CSS.
  - **Example:**

```
css
Copy code
td {
  padding: 10px;
}
```

- **Spacing:** Historically, cellspacing and cellpadding attributes were used directly in HTML to control the spacing between cells and padding inside cells. However, these attributes are now deprecated and should be handled using CSS.
  - **Example (CSS):**

```
css
Copy code
table {
  border-spacing: 5px; /* Space between cells */
}
td {
  padding: 10px; /* Space inside cells */
}
```

Summary:

HTML tables are essential for organizing data in a tabular format, with the ability to define headers, rows, and cells. Attributes like colspan and rowspan enable more complex table structures by merging cells across columns or rows. While HTML attributes for

padding and spacing are deprecated, modern practice uses CSS for styling table layouts.

Glossary :

1. **HTML Table (<table>):**
   - **Definition:** An HTML element that creates a table structure for displaying data in rows and columns.
   - **Example:** <table> ... </table>
2. **Table Row (<tr>):**
   - **Definition:** An HTML element that defines a horizontal row within a table. Each row can contain one or more cells.
   - **Example:** <tr> ... </tr>
3. **Table Data Cell (<td>):**
   - **Definition:** An HTML element that defines a standard cell in a table, used to hold data.
   - **Example:** <td>Data</td>
4. **Table Header Cell (<th>):**
   - **Definition:** An HTML element that defines a header cell in a table, typically used to label columns or rows. It is bold and centered by default.
   - **Example:** <th>Header</th>
5. **Colspan (colspan attribute):**
   - **Definition:** An attribute used in the <td> or <th> tag that specifies the number of columns a cell should span.
   - **Example:** <td colspan="3">Spans 3 columns</td>
6. **Rowspan (rowspan attribute):**
   - **Definition:** An attribute used in the <td> or <th> tag that specifies the number of rows a cell should span.
   - **Example:** <td rowspan="2">Spans 2 rows</td>

7. **Table Padding:**
   - **Definition:** The space between the content of a cell and its border. Padding can be controlled with CSS using the padding property.
   - **Example (CSS):** td { padding: 10px; }
8. **Table Spacing:**
   - **Definition:** Historically, cellspacing (space between cells) and cellpadding (space inside cells) attributes were used to control spacing. These attributes are now deprecated and handled using CSS.
   - **Example (CSS):** table { border-spacing: 5px; } (for spacing between cells)
9. **Table Borders:**
   - **Definition:** The lines that outline the edges of the table, rows, and cells. Borders can be styled using CSS.
   - **Example (CSS):** table { border: 1px solid black; }
10. **Table Caption (<caption>):**
    - **Definition:** An HTML element that provides a title or description for a table. It is placed immediately after the <table> tag.
    - **Example:** <caption>Table Title</caption>
11. **Table Footer (<tfoot>):**
    - **Definition:** An optional HTML element used to group footer content in a table, such as summary rows or totals.
    - **Example:** <tfoot> ... </tfoot>
12. **Table Body (<tbody>):**
    - **Definition:** An HTML element that groups the main content of a table, including the table rows.
    - **Example:** <tbody> ... </tbody>
13. **Table Header Group (<thead>):**
    - **Definition:** An HTML element that groups header content in a table, such as column headings.
    - **Example:** <thead> ... </thead>
14. **Table Alignment:**

- **Definition:** Refers to the alignment of the table itself and its content, which can be controlled using CSS properties such as text-align, vertical-align, and align.
- **Example (CSS):** td { text-align: center; }

15. **Deprecated Attributes:**
    - **Definition:** Attributes like cellspacing and cellpadding that were used in HTML 4.01 but are no longer recommended. CSS should be used for layout and styling.
    - **Example (HTML):** <table cellspacing="5" cellpadding="10"> ... </table>

Self Assessment Questions:

1. Basic Table Structure:

1.1. Define an HTML Table

- What are the essential HTML elements required to create a basic table structure?
- Write a simple HTML snippet that includes a table with two rows and two columns.

1.2. Table Cells and Rows

- How do <td> and <th> elements differ from each other in an HTML table?
- Describe how you would use <tr>, <td>, and <th> to create a table with a header row and two data rows.

2. Table Attributes:

2.1. Colspan and Rowspan

- What is the purpose of the colspan attribute in an HTML table? Provide an example where a cell spans across two columns.

- Explain the rowspan attribute and provide an example of a cell that spans across three rows.

2.2. Table Padding and Spacing

- How was cell padding traditionally controlled in HTML tables, and what is the modern method to achieve this?
- Describe how cellspacing and cellpadding were used in older HTML versions and how CSS is used for table spacing and padding today.

3. Practical Implementation:

3.1. Creating a Table

- Write HTML code to create a table with the following structure:
  - A header row with three columns: "Name", "Age", and "Occupation".
  - Two data rows with sample data for each column.

3.2. Using Colspan and Rowspan

- Create a table where one cell spans across two columns and another spans across two rows. Provide the HTML code for this table.

4. Advanced Table Features:

4.1. Table Layout and Styling

- How can you use CSS to style the borders of a table? Provide an example of CSS rules that apply a solid border to the table, rows, and cells.
- Explain how you would center-align the content of a table using CSS.

4.2. Adding Captions and Footers

- How do you add a caption to an HTML table? Write an example where a caption is included to describe the table's purpose.
- Describe the role of <tfoot>, <thead>, and <tbody> elements in organizing table data. Provide an example where each of these elements is used in a table.

5. Troubleshooting and Validation:

5.1. Common Issues

- What are some common issues that might arise when using colspan and rowspan attributes, and how can they be resolved?
- How can you ensure that your HTML table is accessible and properly structured for screen readers?

5.2. Best Practices

- What are some best practices for designing accessible and responsive tables?
- How would you validate an HTML table to ensure it meets web standards?

Practical Exercises

1. Exercise: Basic Table Creation

- **Task:** Create an HTML file with a table that includes headers for "Product", "Price", and "Quantity". Populate the table with three rows of sample data.
- **Deliverable:** An HTML snippet with the table structure and sample data.

2. Exercise: Advanced Table Layout

- **Task:** Build an HTML table where:
  - One cell in the header spans two columns.
  - One cell in the data section spans two rows.
- **Deliverable:** An HTML snippet demonstrating the use of colspan and rowspan.

3. Exercise: CSS Styling for Tables

- **Task:** Style a table using CSS to:
  - Apply a border to the entire table.
  - Set different border styles for rows and cells.
  - Add padding to cells and set a specific width for the table.
- **Deliverable:** An HTML file with embedded CSS or linked CSS file applying the described styles.

Case Study 1: E-Commerce Product Catalog

Objective: Create a product catalog page for an e-commerce website using HTML tables to display product information clearly.

Description: Design a table to showcase a list of products. Each product should have the following details:

- Product Name
- Description
- Price
- Availability Status

Requirements:

1. **Table Structure:**
   - The table should have a header row with columns for "Product Name," "Description," "Price," and "Availability."
   - Include at least five rows of product data.
2. **Attributes:**
   - Use colspan to create a cell that spans across multiple columns for special announcements or promotions.
   - Use rowspan if a product spans multiple rows due to a detailed description.
3. **Styling:**

- Apply CSS to ensure that the table has a border, padding, and spacing for readability.
- Highlight the header row with a different background color.

Deliverables:

- An HTML file with the table structure and sample data.
- A CSS file (or embedded CSS) for styling the table.

Case Study 2: Academic Schedule

Objective: Design a timetable for an academic semester using HTML tables to display class schedules.

Description: Create a table to present a weekly class schedule. The schedule should include:

- Days of the Week (Monday to Friday)
- Time Slots (e.g., 9 AM - 10 AM, 10 AM - 11 AM)
- Class Name
- Instructor
- Room Number

Requirements:

1. **Table Structure:**
   - Use <thead> to group the header information (days and time slots).
   - Use <tbody> to list the schedule for each time slot and day.
2. **Attributes:**
   - Use colspan for cells that span multiple time slots or days.
   - Use rowspan to indicate class durations that span multiple time slots.
3. **Styling:**

- Use CSS to define table borders, alternating row colors for better readability, and padding for cell content.
- Ensure the table is responsive for different screen sizes.

Deliverables:

- An HTML file with the table structure and schedule data.
- A CSS file (or embedded CSS) for styling the table and making it responsive.

Suggested Readings:

Books:

"HTML and CSS: Design and Build Websites" by Jon Duckett

- **Description:** This book provides a clear, visually appealing introduction to HTML and CSS, including detailed sections on creating and styling tables.
- **ISBN:** 978-1118008188
- **Why Read:** Offers practical examples and illustrations for designing and styling tables.

2. **"HTML5: The Missing Manual" by Matthew MacDonald**
   - **Description:** A comprehensive guide to HTML5 that covers the latest features and best practices, including advanced table features.
   - **ISBN:** 978-1491918641
   - **Why Read:** Provides detailed explanations and examples of HTML5 elements, including tables.
3. **"Head First HTML and CSS" by Elisabeth Robson and Eric Freeman**
   - **Description:** An engaging, beginner-friendly book that covers HTML and CSS fundamentals, with practical exercises on creating tables.
   - **ISBN:** 978-0596006303
   - **Why Read:** Interactive approach with clear explanations and exercises, ideal for hands-on learning.

4. **"Learning Web Design: A Beginner's Guide to HTML, CSS, JavaScript, and Web Graphics" by Jennifer Niederst Robbins**
    - **Description:** This book offers a thorough introduction to web design, including HTML tables, with step-by-step instructions and practical exercises.
    - **ISBN:** 978-1491960201
    - **Why Read:** Comprehensive guide for beginners, focusing on practical web design skills.

UNIT V

UNIT 5 - HTML iFrames
Frameset–Targeted Links–HTML iframe tag–Forms: Input, Text fields – the <label> element – the Name Attribute for<input> - HTML Input types.

TABLE OF CONTENTS

UNIT OBJECTIVES:

In this unit, learners will be able to apply frames in developing new projects. The learners will understand how to develop forms which helps to create so many apps which are in current trends. By using the forms the learners will provide security for their project since privacy plays a major role in the digital world.

5.1. HTML iframes

An HTML iframe is used to display a web page within a web page.

5.1.1 HTML Iframe Syntax

The HTML <iframe> tag specifies an inline frame.

An inline frame is used to embed another document within the current HTML document.

Syntax

<iframe src="*url*" title="*description*"></iframe>

Tip: It is a good practice to always include a title attribute for the <iframe>. This is used by screen readers to read out what the content of the iframe is.

Iframe - Set Height and Width

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

The height and width are specified in pixels by default:

Example

<iframe src="demo_iframe.htm" height="200" width="300" title="Iframe Example"></iframe>

Or you can add the style attribute and use the CSS height and width properties:

Example

```
<iframe src="demo_iframe.htm" style="height:200px;width:300px;" title="Iframe
Example"></iframe>
```

Iframe - Remove the Border

By default, an iframe has a border around it.

To remove the border, add the style attribute and use the CSS border property:

Example

```
<iframe src="demo_iframe.htm" style="border:none;" title="Iframe Example"></iframe>
```

With CSS, you can also change the size, style and color of the iframe's border:

Example

```
<iframe src="demo_iframe.htm" style="border:2px solid red;" title="Iframe
Example"></iframe>
```

5.2. Targeted Links

Iframe - Target for a Link

An iframe can be used as the target frame for a link.

The target attribute of the link must refer to the name attribute of the iframe:

Example

```
<iframe src="demo_iframe.htm" name="iframe_a" title="Iframe Example"></iframe>

<p><a href="https://www.w3schools.com" target="iframe_a">W3Schools.com</a></p>
```

5.3. HTML iframe Tag

Tag	Description
<iframe>	Defines an inline frame

5.4. HTML Forms

An HTML form is used to collect user input. The user input is most often sent to a server for processing.

Example

First name:

John

Last name:

Doe

Submit

The <form> Element

The HTML <form> element is used to create an HTML form for user input:

```
<form>
.
form elements
.
</form>
```

The <form> element is a container for different types of input elements, such as: text fields, checkboxes, radio buttons, submit buttons, etc.

All the different form elements are covered in this chapter: HTML Form Elements.

The <input> Element

The HTML <input> element is the most used form element.

An <input> element can be displayed in many ways, depending on the type attribute.

Here are some examples:

Type	Description
<input type="text">	Displays a single-line text input field
<input type="radio">	Displays a radio button (for selecting one of many choices)
<input type="checkbox">	Displays a checkbox (for selecting zero or more of many choices)
<input type="submit">	Displays a submit button (for submitting the form)
<input type="button">	Displays a clickable button

All the different input types are covered in this chapter: HTML Input Types.

Text Fields

The <input type="text"> defines a single-line input field for text input.

Example

A form with input fields for text:

```
<form>
  <label for="fname">First name:</label><br>
  <input type="text" id="fname" name="fname"><br>
  <label for="lname">Last name:</label><br>
  <input type="text" id="lname" name="lname">
</form>
```

This is how the HTML code above will be displayed in a browser:

First name:

dfg

Last name:

cv

Note: The form itself is not visible. Also note that the default width of an input field is 20 characters.

5.5. The <label> Element

Notice the use of the <label> element in the example above.

The <label> tag defines a label for many form elements.

The <label> element is useful for screen-reader users, because the screen-reader will read out loud the label when the user focuses on the input element.

The <label> element also helps users who have difficulty clicking on very small regions (such as radio buttons or checkboxes) - because when the user clicks the text within the <label> element, it toggles the radio button/checkbox.

The for attribute of the <label> tag should be equal to the id attribute of the <input> element to bind them together.

Radio Buttons

The <input type="radio"> defines a radio button.

Radio buttons let a user select ONE of a limited number of choices.

Example

A form with radio buttons:

```
<p>Choose your favorite Web language:</p>

<form>
  <input type="radio" id="html" name="fav_language" value="HTML">
  <label for="html">HTML</label><br>
  <input type="radio" id="css" name="fav_language" value="CSS">
  <label for="css">CSS</label><br>
  <input type="radio" id="javascript" name="fav_language" value="JavaScript">
  <label for="javascript">JavaScript</label>
</form>
```

This is how the HTML code above will be displayed in a browser:

Choose your favorite Web language:

- HTML
- CSS
- JavaScript

Checkboxes

The <input type="checkbox"> defines a **checkbox**.

Checkboxes let a user select ZERO or MORE options of a limited number of choices.

Example

A form with checkboxes:

```
<form>
  <input type="checkbox" id="vehicle1" name="vehicle1" value="Bike">
  <label for="vehicle1"> I have a bike</label><br>
  <input type="checkbox" id="vehicle2" name="vehicle2" value="Car">
  <label for="vehicle2"> I have a car</label><br>
  <input type="checkbox" id="vehicle3" name="vehicle3" value="Boat">
  <label for="vehicle3"> I have a boat</label>
</form>
```

This is how the HTML code above will be displayed in a browser:

- I have a bike
- I have a car
- I have a boat

The Submit Button

The <input type="submit"> defines a button for submitting the form data to a form-handler.

The form-handler is typically a file on the server with a script for processing input data.

The form-handler is specified in the form's action attribute.

Example

A form with a submit button:

```
<form action="/action_page.php">
  <label for="fname">First name:</label><br>
  <input type="text" id="fname" name="fname" value="John"><br>
  <label for="lname">Last name:</label><br>
  <input type="text" id="lname" name="lname" value="Doe"><br><br>
  <input type="submit" value="Submit">
</form>
```

This is how the HTML code above will be displayed in a browser:

First name:

John

Last name:

Doe

Submit

5.6. The Name Attribute for <input>

Notice that each input field must have a name attribute to be submitted.

If the name attribute is omitted, the value of the input field will not be sent at all.

Example

This example will not submit the value of the "First name" input field:

```
<form action="/action_page.php">
  <label for="fname">First name:</label><br>
  <input type="text" id="fname" value="John"><br><br>
  <input type="submit" value="Submit">
</form>
```

HTML Input Types

This chapter describes the different types for the HTML <input> element.

5.7. HTML Input Types

Here are the different input types you can use in HTML:

- <input type="button">
- <input type="checkbox">
- <input type="color">
- <input type="date">
- <input type="datetime-local">
- <input type="email">
- <input type="file">
- <input type="hidden">
- <input type="image">
- <input type="month">
- <input type="number">
- <input type="password">
- <input type="radio">
- <input type="range">
- <input type="reset">
- <input type="search">
- <input type="submit">
- <input type="tel">
- <input type="text">
- <input type="time">
- <input type="url">
- <input type="week">

Tip: The default value of the type attribute is "text".

Input Type Text

<input type="text"> defines a **single-line text input field**:

Example

```
<form>
  <label for="fname">First name:</label><br>
  <input type="text" id="fname" name="fname"><br>
  <label for="lname">Last name:</label><br>
  <input type="text" id="lname" name="lname">
</form>
```

This is how the HTML code above will be displayed in a browser:

First name:

Last name:

Input Type Password

<input type="password"> defines a **password field**:

Example

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

This is how the HTML code above will be displayed in a browser:

Username:

Password:

The characters in a password field are masked (shown as asterisks or circles).

Summary :

HTML iFrames and Framesets

1. iFrames (Inline Frames):

- **Definition:** An <iframe> (inline frame) is an HTML element used to embed another HTML document within the current page. It allows for the display of content from other sources without requiring the user to leave the page.
- **Syntax:**

```
html
Copy code
<iframe src="url" width="width" height="height" frameborder="border"></iframe>
```

- **Attributes:**
  - src: URL of the page to display within the iframe.
  - width and height: Dimensions of the iframe.
  - frameborder: Specifies whether to display a border around the iframe (deprecated in HTML5, use CSS instead).
  - sandbox: Provides security features to restrict the iframe's capabilities (e.g., sandbox="allow-scripts").

2. Frameset (Deprecated):

- **Definition:** <frameset> was used in earlier HTML versions to define a set of frames on a page. It allowed multiple HTML documents to be displayed in separate sections of the browser window.
- **Syntax:**

```
html
Copy code
<frameset cols="25%, 75%">
  <frame src="left.html">
```

```
  <frame src="right.html">
</frameset>
```

- **Attributes:**
  - cols: Specifies the width of each frame.
  - rows: Specifies the height of each frame.
- **Note:** The <frameset> element is deprecated in HTML5 and should be avoided in favor of using <iframe> or CSS for layout purposes.

3. Targeted Links:

- **Definition:** Targeted links involve using the target attribute in anchor tags to specify where the linked document should be opened.
- **Syntax:**

```
html
Copy code
<a href="url" target="frameName">Link Text</a>
```

- **Attributes:**
  - target="_blank": Opens the link in a new tab or window.
  - target="_self": Opens the link in the same frame (default).
  - target="_parent": Opens the link in the parent frame.
  - target="_top": Opens the link in the full window.

Forms in HTML

1. Input Fields:

- **Definition:** Input fields are elements within a form where users can enter data. They are created using the <input> tag.
- **Types:** Various input types control the behavior and appearance of the input field.
  - text: A single-line text field.

- password: A text field that obscures input (used for passwords).
- email: A field that validates email addresses.
- number: A field for numerical input.
- checkbox: A field for binary options (checked or unchecked).
- radio: A field for selecting one option from a set.
- submit: A button to submit the form.

2. The <label> Element:

- **Definition:** The <label> element provides a user-readable description for form controls, improving accessibility and usability.
- **Syntax:**

```
html
Copy code
<label for="inputId">Label Text</label>
<input type="text" id="inputId" name="inputName">
```

- **Attributes:**
  - for: Associates the label with a specific form control by matching the id of the control.

3. The name Attribute for <input>:

- **Definition:** The name attribute specifies the name of the input field, which is used to identify the data when the form is submitted.
- **Syntax:**

```
html
Copy code
<input type="text" name="username">
```

4. HTML Input Types:

- **Definition:** The type attribute of the <input> element defines the type of data the field will accept and affects the input's appearance and behavior.
- **Common Input Types:**
  - text: Standard text input.
  - password: Input for passwords.
  - email: Input for email addresses with validation.
  - number: Input for numerical values with optional restrictions.
  - date: Input for selecting dates.
  - file: Allows users to upload files.
  - button: Creates a clickable button.
  - submit: Submits the form data to the server.
  - reset: Resets the form fields to their default values.

Glossary :

HTML iFrames

1. **iFrame (<iframe>):**
   - **Definition:** An HTML element used to embed another HTML document within the current document. It allows for displaying external content on a webpage.
   - **Attributes:**
     - src: URL of the document to display in the iframe.
     - width: Specifies the width of the iframe.
     - height: Specifies the height of the iframe.
     - frameborder: (Deprecated) Specifies whether to display a border around the iframe.
     - sandbox: Restricts the iframe’s capabilities for security purposes.
2. **Sandbox Attribute:**
   - **Definition:** An attribute for <iframe> that enables additional security features by limiting the iframe's functionality.
   - **Example:** sandbox="allow-scripts"

Frameset (Deprecated)

3. **Frameset (<frameset>):**
    - **Definition:** An HTML element used in earlier versions of HTML to create multiple frames on a single webpage, allowing different HTML documents to be displayed in separate sections of the browser window.
    - **Attributes:**
        - cols: Defines the width of the frames in the frameset.
        - rows: Defines the height of the frames in the frameset.
4. **Frame (<frame>):**
    - **Definition:** A child element of <frameset> that specifies the content to be displayed in each frame.
    - **Attributes:**
        - src: URL of the document to display in the frame.

Targeted Links

5. **Target Attribute (target):**
    - **Definition:** An attribute used in <a> (anchor) elements to specify where to open the linked document.
    - **Values:**
        - _blank: Opens the link in a new tab or window.
        - _self: Opens the link in the same frame (default behavior).
        - _parent: Opens the link in the parent frame.
        - _top: Opens the link in the full window.

Forms

6. **Form (<form>):**
    - **Definition:** An HTML element that contains input fields and controls to collect user data and submit it to a server.
    - **Attributes:**

- action: URL where the form data will be submitted.
- method: HTTP method used for form submission (e.g., GET, POST).

7. **Input Field (<input>):**
   - **Definition:** A versatile HTML element used to create interactive fields in forms where users can enter data.
   - **Attributes:**
     - type: Defines the type of input field (e.g., text, password, email).
     - name: Name of the input field, used to identify the data when the form is submitted.
     - value: Specifies the initial value of the input field.
8. **Label (<label>):**
   - **Definition:** An HTML element used to define a label for a form control, enhancing accessibility and usability.
   - **Attributes:**
     - for: Associates the label with a specific form control using the control's id.
9. **Input Types:**
   - **Definition:** Specifies the type of data an <input> element should accept.
   - **Common Types:**
     - text: Standard single-line text input.
     - password: Obscured text input for passwords.
     - email: Input field for email addresses with built-in validation.
     - number: Numeric input with optional constraints.
     - date: Input field for selecting dates.
     - file: Allows users to upload files.
     - submit: Button to submit the form.
     - reset: Button to reset the form fields.
10. **Name Attribute:**
    - **Definition:** The name attribute of an <input> element specifies the name of the field, which is used to identify the data submitted by the form.

Self-Assessment Questions

HTML iFrames and Framesets

1. iFrames:

1. **What is an <iframe> and how is it used in HTML?**
   - Explain the purpose of the <iframe> element and provide an example of how it might be used on a webpage.
2. **List and describe at least three attributes of the <iframe> element.**
   - What does the src attribute do? How does the width attribute affect the iframe? What is the function of the sandbox attribute?
3. **Write a snippet of HTML code that embeds a YouTube video into a webpage using an <iframe>.**
   - Ensure the iframe is 560 pixels wide and 315 pixels high.

2. Framesets (Deprecated):

4. **What is a <frameset> and why is it considered deprecated in HTML5?**
   - Describe the purpose of <frameset> and its attributes. Explain why modern HTML practices have moved away from using framesets.
5. **Create an HTML snippet using <frameset> to display two documents side-by-side, with the left frame taking up 30% of the width and the right frame 70%.**
   - Provide an example with the appropriate cols attribute values.

3. Targeted Links:

6. **How can the target attribute in a link be used to open the linked document in a new window or tab?**
   - Explain the value _blank and its effect on link behavior.
7. **Write an example of a hyperlink that opens the linked page in the parent frame of a frameset.**

- Include the target attribute in your example.

Forms:

4. Input Fields:

8. **What are the different types of input fields available in HTML forms? Provide examples of at least five different types and their use cases.**
   - Examples could include text, password, email, checkbox, and radio.
9. **Write an HTML form that includes a text input field, a password field, and a submit button.**
   - Ensure each field has a name attribute and the form includes the action and method attributes.

5. The <label> Element:

10. **What is the purpose of the <label> element in a form? How does it improve accessibility?**
    - Describe how <label> enhances user experience and accessibility, and provide an example of its usage with an <input> field.
11. **Create a label for a text input field with the id username and provide a descriptive label text.**
    - Write the HTML code including the for attribute.

6. The name Attribute:

12. **Why is the name attribute important in form input fields?**
    - Explain its role in data submission and how it helps in identifying form data on the server.
13. **Provide an example of an input field with the name attribute set to email and the type attribute set to email.**
    - Include the HTML code snippet for this input field.

7. HTML Input Types:

14. **How does the `type` attribute of an `<input>` element affect the behavior and appearance of the input field?**
    - Describe the differences between `text`, `password`, `number`, and `email` input types.
15. **Write an HTML form that includes fields for `name`, `email`, and `age`, using appropriate input types for each field.**
    - Ensure the `name` and `type` attributes are set correctly for each input.

Practical Exercises

1. iFrames:

- **Task:** Embed an external website (such as Wikipedia) within an HTML page using an `<iframe>`. Set the iframe to be 800 pixels wide and 600 pixels high.
- **Deliverable:** An HTML file with the iframe code.

2. Framesets:

- **Task:** Create a simple HTML page using `<frameset>` to display two pages, one on the top and one on the bottom. The top frame should be 25% of the height and the bottom frame 75%.
- **Deliverable:** An HTML file with `<frameset>` and `<frame>` code.

3. Forms:

- **Task:** Design a registration form with fields for first name, last name, email address, password, and a submit button. Use appropriate input types and include `<label>` elements.
- **Deliverable:** An HTML file with the complete form and associated labels.

Activities and Exercises

1. iFrames

Activity 1: Embedding External Content

- **Objective:** Practice using the <iframe> element to embed external content within a webpage.
- **Task:**
  - Create an HTML file that includes an iframe embedding a Google Map showing a specific location of your choice (e.g., a famous landmark).
  - Set the iframe to be 600 pixels wide and 400 pixels high.
  - Add a border around the iframe using CSS for better visibility.
- **Deliverable:** An HTML file with the iframe code and a CSS file for styling.

Activity 2: Responsive iFrame

- **Objective:** Learn how to make an iframe responsive.
- **Task:**
  - Create a webpage with an iframe that embeds a YouTube video.
  - Use CSS to ensure the iframe resizes proportionally with the browser window. Implement a responsive container that maintains the aspect ratio of the video.
- **Deliverable:** An HTML file with an iframe and CSS code for responsiveness.

2. Framesets (Deprecated)

Activity 3: Basic Frameset Layout

- **Objective:** Understand the use of <frameset> and <frame> elements (note: for educational purposes as <frameset> is deprecated).
- **Task:**

  - Create an HTML file with a frameset that divides the window into two vertical frames. The left frame should occupy 30% of the width and the right frame 70%.
  - Include different HTML documents in each frame.
- **Deliverable:** An HTML file using <frameset> and <frame> elements.

Activity 4: Multi-Row Frameset Layout

- **Objective:** Create a more complex frameset layout.
- **Task:**
  - Design a frameset with three rows: a top frame (header), a middle frame (content), and a bottom frame (footer).
  - The top frame should be 20% of the height, the middle frame 60%, and the bottom frame 20%.
  - Include different HTML documents in each frame.
- **Deliverable:** An HTML file with a multi-row frameset layout.

3. Targeted Links

Activity 5: Targeted Link Navigation

- **Objective:** Use the target attribute to control where links open.
- **Task:**
  - Create an HTML page with several links.
  - Set up the links to open in different targets such as a new tab, the same frame, or a specific iframe on the same page.
  - Ensure one link targets an iframe embedded within the same page.
- **Deliverable:** An HTML file with links and target attributes.

Activity 6: Target Attribute Experimentation

- **Objective:** Experiment with different values for the target attribute.
- **Task:**

  - Create a webpage with a series of links that demonstrate each target attribute value (_blank, _self, _parent, _top).
  - Include explanations or comments in the HTML code about what each target value does.
- **Deliverable:** An HTML file showcasing various target attribute values.

4. Forms

Activity 7: Basic Form Creation

- **Objective:** Build a simple form with various input fields.
- **Task:**
  - Create an HTML form that includes text fields, a password field, radio buttons, checkboxes, and a submit button.
  - Each input field should have a corresponding <label> with descriptive text.
- **Deliverable:** An HTML file with a fully functional form.

Activity 8: Form Validation

- **Objective:** Implement form validation using HTML5 attributes.
- **Task:**
  - Design a registration form that includes fields for name, email, password, and date of birth.
  - Use HTML5 input attributes such as required, pattern, and minlength to validate the form fields.
  - Ensure that appropriate validation messages are displayed.
- **Deliverable:** An HTML file with validation attributes and a functional form.

Activity 9: Form Submission

- **Objective:** Understand how to handle form submission.
- **Task:**
  - Create an HTML form with fields for name, email, and message.

- Use the action and method attributes to configure form submission.
- Implement a simple PHP or server-side script (if available) to handle the form data (alternatively, use a form-handling service).

- **Deliverable:** An HTML form with action and method attributes, and a server-side script or configuration for handling form submission.

Case Studies

Case Study 1: E-Commerce Product Page

Objective: Design a product page for an e-commerce site using HTML tables and forms.

Description:

Create a webpage that displays a product with details such as product name, description, price, and availability using HTML tables.

- Include an embedded iframe showing a related video or review.
- Provide a form for customers to add the product to their cart, including input fields for quantity and a "Add to Cart" button.

Requirements:

1. **Product Table:**
   - Use a table to display product details with headers for "Name," "Description," "Price," and "Availability."
2. **iFrame:**
   - Embed a related video or review in an iframe on the page.
3. **Form:**
   - Include a form with a quantity input field and a submit button to simulate adding the product to a shopping cart.

Deliverables:

- An HTML file with the product details table, iframe, and form.

Case Study 2: Online Registration System

Objective: Build an online registration form for a conference or event.

Description:

- Design a registration form that includes fields for personal information, contact details, and registration preferences.
- Include form elements such as text fields, email fields, radio buttons, checkboxes, and a submit button.
- Use appropriate form validation attributes to ensure data is entered correctly.

Requirements:

1. **Form Fields:**
    - Include fields for first name, last name, email address, phone number, and registration preferences.
2. **Validation:**
    - Use HTML5 validation attributes to ensure required fields are filled out and data is in the correct format.
3. **Submission:**
    - Set up the form to submit to a specific URL or server-side script.

Deliverables:

- An HTML file with a registration form, including validation attributes and submission configuration.

Suggested Readings:

Books

1. **"HTML and CSS: Design and Build Websites"** by Jon Duckett
    - **Description:** This book offers a clear and visual approach to learning HTML and CSS, including chapters on forms and how to structure HTML documents effectively.
    - **Publisher:** Wiley
    - **ISBN:** 978-1118008188
2. **"HTML5: The Missing Manual"** by Matthew MacDonald
    - **Description:** A comprehensive guide to HTML5, covering new features and elements including forms and embedding content.
    - **Publisher:** O'Reilly Media
    - **ISBN:** 978-0596159900
3. **"Learning Web Design: A Beginner's Guide to HTML, CSS, JavaScript, and Web Graphics"** by Jennifer Niederst Robbins
    - **Description:** A beginner-friendly guide that introduces web design fundamentals, including HTML elements, forms, and best practices.
    - **Publisher:** O'Reilly Media
    - **ISBN:** 978-1492050728