

UNIT 1

HTML and CSS

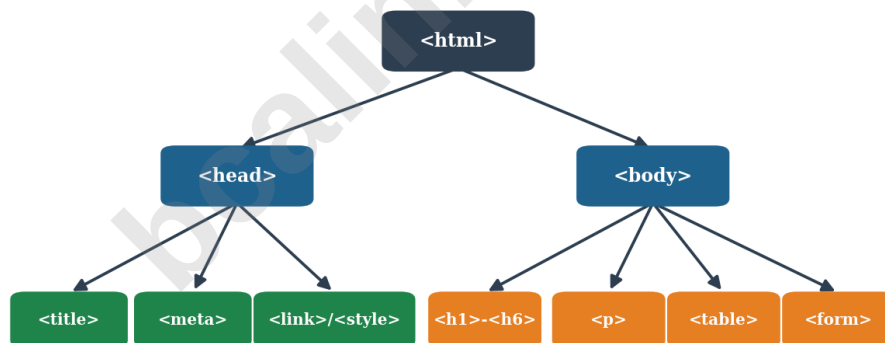
(15 Hours) — Web Technology (CAC205)

1.1 HTML Basics

HTML (HyperText Markup Language) is the standard markup language used to describe the structure of web pages. A browser reads an HTML file, interprets the *tags* in it, and renders the content as a formatted page. HTML is not a programming language; it only marks up text, images, links, tables and forms so that the browser knows how to display them.

An HTML **element** normally has three parts: a start tag, the content, and an end tag, e.g. `<p>Hello</p>`. Some elements are *empty* (void) and have no end tag, such as `<br>`, `<hr>` and `<img>`. Tag names are not case-sensitive, but lowercase is the convention.

1.1.1 Document Structure Tags



Every HTML page is a tree: `<html>` holds one `<head>` and one `<body>`

Figure 1.1 — Tree structure of an HTML document

Tag	Purpose
<code><!DOCTYPE html></code>	Tells the browser the document is HTML5 (not an element).
<code><html></code>	Root element; wraps the entire document. Attribute <i>lang</i> sets the language.
<code><head></code>	Holds metadata that is not displayed: title, meta, link, style, script.
<code><title></code>	Text shown in the browser tab / title bar and in search results.
<code><meta></code>	Metadata such as character set, description, keywords, viewport.

Tag	Purpose
<body>	Contains all visible content of the page.

Minimal HTML document

```

<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>My First Page</title>
</head>
<body>
  <h1>Welcome</h1>
  <p>This is my first web page.</p>
</body>
</html>

```

1.1.2 Global Attributes

Global attributes can be used on almost every HTML element, regardless of its type.

Attribute	Meaning	Example
id	Unique identifier of the element (used by CSS/JavaScript).	id="header"
class	One or more class names; used for styling groups of elements.	class="box red"
style	Inline CSS applied to that single element.	style="color:red"
title	Advisory text shown as a tooltip on hover.	title="Click me"
lang	Language of the element's content.	lang="ne"
dir	Text direction: ltr or rtl.	dir="rtl"
hidden	Hides the element from display.	hidden
tabindex	Order of focus when the Tab key is pressed.	tabindex="1"
accesskey	Keyboard shortcut to activate/focus the element.	accesskey="s"
contenteditable	Makes the element's content editable by the user.	contenteditable="true"

1.1.3 Text Level and Block Level Formatting

HTML elements are broadly divided into **block-level** and **inline (text-level)** elements. A block-level element always starts on a new line and stretches to the full available width (e.g. <p>, <div>, <h1>,). An inline element flows within a line of text and takes only as much width

as its content (e.g. `<span>`, `<a>`, `<b>`, `<em>`).

Text-level tag	Effect	Block-level tag	Effect
<code></code> / <code></code>	Bold / strong importance	<code><h1></code> ... <code><h6></code>	Headings, largest to smallest
<code><i></code> / <code></code>	Italic / emphasis	<code><p></code>	Paragraph
<code><u></code>	Underline	<code><div></code>	Generic block container
<code><sub></code> / <code><sup></code>	Subscript / superscript	<code><pre></code>	Preformatted text (keeps spaces)
<code><small></code>	Smaller text	<code><blockquote></code>	Long quotation, indented
<code><mark></code>	Highlighted text	<code><hr></code>	Horizontal rule
<code><code></code>	Computer code	<code><address></code>	Contact information
<code>
</code>	Line break	<code><center></code>	Centres content (deprecated)
<code></code>	Generic inline container	<code><section></code> , <code><article></code>	Semantic HTML5 sections

Note: `<b>`, `<i>`, `<u>` are presentational; prefer `<strong>`, `<em>` and CSS in modern HTML because they carry meaning (semantics) and help accessibility.

1.1.4 List Tags

Type	Tags	Notes
Unordered list	<code></code> <code></code>	Bullets; <i>type</i> = disc, circle, square (CSS <i>list-style-type</i> in HTML5).
Ordered list	<code></code> <code></code>	Numbers; attributes <i>type</i> (1, A, a, I, i), <i>start</i> , <i>reversed</i> .
Definition list	<code><dl></code> <code><dt></code> <code><dd></code>	Term (<i>dt</i>) followed by its description (<i>dd</i>).
Nested list	list inside <code></code>	Creates multi-level outlines.

Lists

```
<ol type="A" start="3">
  <li>Client tier</li>
  <li>Server tier
    <ul><li>Web server</li><li>Database</li></ul>
  </li>
</ol>
<dl>
  <dt>HTML</dt><dd>Markup language for web pages</dd>
  <dt>CSS</dt><dd>Language for styling web pages</dd>
</dl>
```

1.1.5 Hyperlink Tags

Hyperlinks are created with the anchor tag `<a>`. The `href` attribute holds the destination. Links may point to another site (absolute URL), another page of the same site (relative URL), a location inside the same page (fragment using `#id`), an e-mail address (`mailto:`) or a phone number (`tel:`).

Attribute	Use
<code>href</code>	Destination URL.
<code>target</code>	Where to open: <code>_self</code> (same window, default), <code>_blank</code> (new tab), <code>_parent</code> , <code>_top</code> , or a frame name.
<code>title</code>	Tooltip text.
<code>name</code> / <code>id</code>	Defines a named anchor (bookmark) that other links can jump to.
<code>rel</code>	Relationship, e.g. <code>nofollow</code> , <code>noopener</code> .
<code>download</code>	Downloads the target instead of navigating to it.

Hyperlink examples

```
<a href="https://www.example.com" target="_blank">Visit site</a>
<a href="notes/unit1.html">Relative link</a>
<a href="#section2">Jump to Section 2</a>
<h2 id="section2">Section 2</h2>
<a href="mailto:info@example.com">E-mail us</a>
```

1.1.6 Executable Content Tags

Executable content is code or a program that runs inside or alongside the page. The tags used to embed it are:

- `<script>` — embeds or links JavaScript (`<script src="app.js"></script>`).
`<noscript>` gives fallback content.
- `<applet>` — embedded Java applets (obsolete, removed from HTML5).
- `<object>` and `<embed>` — embed plug-in content such as PDF, Flash (obsolete), video or other media.
- `<iframe>` — embeds another HTML document inside the page.
- `<audio>` and `<video>` — HTML5 native multimedia playback.

1.2 Images and Image Maps

Images are inserted with the empty tag `<img>`. Its important attributes are:

Attribute	Description
src	URL/path of the image file (required).
alt	Alternative text shown if the image cannot load and read by screen readers (required for accessibility).
width, height	Size of the image in pixels (or CSS).
border, align, hspace, vspace	Old presentational attributes; replaced by CSS.
usemap	Associates the image with a client-side image map (#mapname).
ismap	Marks the image as a server-side image map.

Image tag

```

```

1.2.1 Image Maps

An **image map** is a single image in which different regions (hot spots) act as separate hyperlinks. Clicking a region takes the user to a different URL. There are two kinds.

	Client-Side Image Map	Server-Side Image Map
Processed by	The browser	The web server
Markup	<map> and <area> tags, usemap	ismap attribute on img inside a link
What is sent	Nothing; browser resolves the region locally	Click coordinates (x,y) appended to the URL
Speed	Fast, no server round trip	Slower; needs a server program/map file
Browser feedback	Status bar shows the target URL	Only coordinates displayed
Preferred?	Yes — the standard approach	Legacy; used when server logic is needed

Client-side image map

```

<map name="campus">
  <area shape="rect" coords="10,10,120,90" href="library.html" alt="Library">
  <area shape="circle" coords="200,60,40" href="canteen.html" alt="Canteen">
  <area shape="poly" coords="250,10,320,10,300,90" href="lab.html" alt="Lab">
  <area shape="default" href="index.html" alt="Home">
</map>
```

Server-side image map

```
<a href="/cgi-bin/imagemap.cgi">
  
</a>
```

```
<!-- Clicking sends: /cgi-bin/imapemap.cgi?125,48 -->
```

Using both together: an image may carry both `usemap` and `ismap` and be wrapped in a link. Browsers that understand client-side maps use them; older browsers fall back to the server-side map through the surrounding anchor. **Alternative text** must be supplied with the `alt` attribute on each `<area>` so that text-only browsers and screen-reader users can still follow the links. Repeating the links as normal text links below the image is also good practice.

Shape	coords format
rect	x1,y1,x2,y2 (top-left and bottom-right corners)
circle	x,y,r (centre and radius)
poly	x1,y1,x2,y2,...,xn,yn (each polygon vertex)
default	Anything not covered by other areas

1.3 Tables

An HTML table arranges data in rows and columns. Tables were also widely used as a *layout tool* in early web design, but modern practice reserves them for tabular data and uses CSS for layout.

1.3.1 Table Tags and Structure

Tag	Description
<code><table></code>	Defines the whole table.
<code><tr></code>	Table row.
<code><th></code>	Header cell — bold and centred by default.
<code><td></code>	Data cell.
<code><caption></code>	Title of the table (placed first inside table).
<code><thead></code> , <code><tbody></code> , <code><tfoot></code>	Group header, body and footer rows (table sections).
<code><colgroup></code> , <code><col></code>	Define properties (width, colour) for one or more columns.

1.3.2 Table Attributes

Purpose	Attribute / Tag	Notes
Border	<code>border</code>	Width in pixels of the table border.

Purpose	Attribute / Tag	Notes
Width	width	In pixels or %, e.g. width="80%".
Background colour	bgcolor	On table, row or cell (CSS background-color preferred).
Spacing within a cell	cellpadding	Space between cell content and cell border.
Spacing between cells	cellspacing	Gap between adjacent cells.
Aligning entire table	align	left, center, right on .
Alignment within a row	align, valign	On : horizontal (left/center/right) and vertical (top/middle/bottom).
Alignment within a cell	align, valign	On /; overrides the row setting.
Spanning columns	colspan	Cell stretches across n columns.
Spanning rows	rowspan	Cell stretches across n rows.
Content summary	summary	Text description for non-visual browsers (obsolete in HTML5; use caption).

A complete table using spans and sections

```
<table border="1" cellpadding="6" cellspacing="0" width="80%" align="center">
  <caption>Student Marks</caption>
  <thead>
    <tr bgcolor="#dddddd"><th>Roll</th><th colspan="2">Marks</th></tr>
  </thead>
  <tbody>
    <tr><td rowspan="2" align="center">1</td><td>Web Tech</td><td>78</td></tr>
    <tr><td>Database</td><td>81</td></tr>
  </tbody>
  <tfoot><tr><td colspan="3" align="right">End of report</td></tr></tfoot>
</table>
```

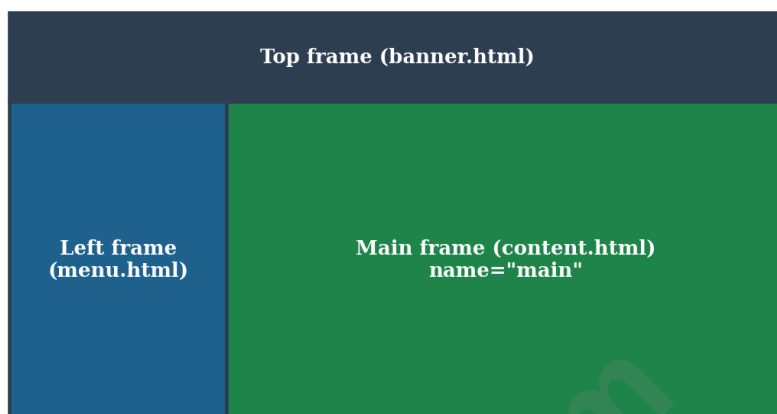
Elements that can be placed in a table cell: any body content — text, images, links, lists, forms, even another (nested) table. **Tables as a design tool:** before CSS layout was mature, designers nested tables with invisible borders to position menus, banners and columns. This produces heavy, inflexible, inaccessible pages, so today CSS (Flexbox, Grid) is used for layout and tables only for data.

1.4 Frames

Frames divide the browser window into multiple independent rectangular regions, each displaying its own HTML document. A typical use is a fixed navigation menu in one frame and changing content in another. Frames are *obsolete in HTML5* (except `<iframe>`) but remain part of the syllabus.

1.4.1 Frameset Document

A frames document has a `<frameset>` *instead of* a `<body>`. The `rows` or `cols` attribute splits the window using pixels, percentages or `*` (remaining space). Framesets can be **nested** to build complex layouts.



`<frameset rows="22%,*">` containing `<frame>` and a nested `<frameset cols="28%,*">`

Figure 1.2 — A nested frameset layout

Nested framesets

```
<html>
<head><title>Frames Demo</title></head>
<frameset rows="22%,*">
  <frame src="banner.html" name="top" noresize scrolling="no">
  <frameset cols="28%,*">
    <frame src="menu.html" name="menu">
    <frame src="content.html" name="main">
  </frameset>
</frameset>
<noframes><body>Your browser does not support frames.</body></noframes>
</frameset>
</html>
```

1.4.2 Targeting, Floating and Hidden Frames

- **Targeting named frames:** give a frame a name and use `<a href="page2.html" target="main">` so that the linked page loads in that frame. Special targets: `_self`, `_parent`, `_top`, `_blank`.
- **Floating frames:** `<iframe>` places a frame inside the normal flow of a page (attributes `src`, `name`, `width`, `height`, `frameborder`).
- **Hidden frames:** a frame with zero height/width (e.g. `rows="0, *"`) used to hold scripts or exchange data with the server without showing anything.

Floating frame

```
<iframe src="map.html" name="mapframe" width="400" height="300"></iframe>
```

```
<a href="uk.html" target="mapframe">Show UK</a>
```

Applications	Drawbacks
Permanent navigation menu / header	Bookmarking and printing are awkward
Displaying documentation with a table of contents	Poor for search engines and accessibility
Embedding third-party content (iframe)	Extra HTTP requests; not supported in HTML5 frameset

1.5 Forms

HTML forms collect input from users and send it to a server for processing. A form is created with the `<form>` tag and contains input controls.

1.5.1 The `<FORM>` Tag

Attribute	Meaning
action	URL of the program (script) that processes the submitted data.
method	GET : data appended to URL (visible, limited length, bookmarkable). POST : data in the request body (hidden, larger, used for sensitive/changing data).
enctype	Encoding of data: application/x-www-form-urlencoded (default), multipart/form-data (needed for file upload), text/plain.
name / id	Names the form so scripts can reference it.
target	Where to display the response.
onsubmit, onreset	Event handlers run when the form is submitted or reset.

1.5.2 The `<INPUT>` Tag and Form Controls

The `<input>` tag's `type` attribute decides the kind of control. Every control that sends data must have a `name` (the **named input field**); the name/value pair is what the server receives.

Control	Markup	Notes
Text	<code><input type="text" name="user"></code>	Single-line text; maxlength, size, value, placeholder.
Password	<code><input type="password"></code>	Characters masked.
Hidden	<code><input type="hidden" name="id" value="7"></code>	Not shown; passes data invisibly.

Control	Markup	Notes
Text area	<code><textarea rows cols></code>	Multiple-line text window.
Radio	<code><input type="radio" name="g" value="M"></code>	One choice from a group (same name).
Checkbox	<code><input type="checkbox" name="hobby"></code>	Zero or more choices.
Select / Option	<code><select><option></code>	Drop-down; add multiple/size for a list box.
File upload	<code><input type="file"></code>	Needs method=post, enctype=multipart/form-data.
Submit	<code><input type="submit"></code>	Sends the form to the action URL.
Reset	<code><input type="reset"></code>	Restores initial values.
Button	<code><input type="button"> / <button></code>	Generic button; used with scripts (action buttons).

1.5.3 Labels, Grouping, Disabled and Read-Only Fields

- **Labelling:** `<label for="id">` links text to a control; clicking the label focuses the control.
- **Grouping related fields:** `<fieldset>` draws a box around related controls; `<legend>` gives it a caption.
- **disabled** — the control is greyed out and its value is *not* submitted.
- **readonly** — the value cannot be edited but *is* submitted with the form.

1.5.4 Forms and Scripting, Event Handlers, Passing Data

Forms are usually validated on the client with JavaScript before submission. Common **form field event handlers** are `onfocus`, `onblur`, `onchange`, `onclick`, `onselect`, `onkeyup`, `onsubmit` and `onreset`. **Passing form data:** when submitted, the browser encodes each control as `name=value` pairs joined by `&` (e.g. `user=ram&pass=123`) and sends them by GET (in the query string) or POST (in the body).

Complete form example

```
<form action="register.php" method="post" onsubmit="return check()">
  <fieldset>
    <legend>Registration</legend>
    <label for="n">Name:</label>
    <input type="text" id="n" name="name" required><br>
    <label for="p">Password:</label>
    <input type="password" id="p" name="pass"><br>
    Gender:
```

```

☒ Male
☐ Female<br>
Interests:
☐ Web
☐ Database<br>
Country:
<select name="country">
  <option value="np" selected="selected">Nepal</option>
  <option value="in">India</option>
</select><br>
<textarea name="msg" rows="3" cols="30"></textarea><br>


```

Example 1.1 — GET vs POST

A search form uses `method="get"` with a text field named `q`. The user types *html*.

Which URL is requested, and why is POST better for a login form?

Solution

GET request: `search.php?q=html` — the data is visible in the address bar and browser history.

POST is better for login because the password is sent in the request body, is not stored in the URL/history, and has no length limit.

1.6 Style Sheets (CSS)

Cascading Style Sheets (CSS) describe the presentation — colours, fonts, spacing and layout — of HTML documents. HTML defines *what* the content is; CSS defines *how it looks*.

1.6.1 Importance of Style Sheets

- **Separation of content and presentation** — cleaner, smaller HTML.
- **Consistency** — one file controls the look of an entire site.
- **Easy maintenance** — change one rule to update every page.
- **Faster loading** — a style sheet is cached and reused.
- **Richer control** — layout, positioning, animation and responsive design impossible with plain HTML.
- **Accessibility** — different style sheets for screen, print, mobile, etc.

CSS syntax: a rule consists of a selector and a declaration block: `selector { property: value; }`. Selectors include element (`p`), class (`.note`), id (`#top`), grouping (`h1, h2`), descendant (`div p`) and universal (`*`).

1.6.2 Different Approaches to Style Sheets

Approach	How	Scope	Priority
Inline	<code><p style="color:red"></code>	One element only	Highest
Embedded (internal)	<code><style>...</style></code> in	One page	Medium
Linked (external)	<code><link rel="stylesheet" href="s.css"></code>	Whole site	Lowest (but most reusable)
Imported	<code>@import url("s.css");</code>	Inside a style sheet/style tag	Depends on order

Three ways to apply CSS

```
<!-- 1. External: linking to style information in a separate file -->
<head>
  <link rel="stylesheet" type="text/css" href="css/style.css">

  <!-- 2. Embedded: using the <style> tag -->
  <style>
    body { background:#f4f6f7; font-family: Georgia, serif; }
    h1 { color:#1f618d; text-align:center; }
    .note{ color:purple; font-style:italic; }
    #top { border-bottom:2px solid navy; }
  </style>
</head>
<body>
  <!-- 3. Inline style information -->
  <p style="color:red; font-size:14pt;">Inline styled paragraph</p>
</body>
```

Using multiple approaches (the cascade): when several style sheets apply to the same element, the browser resolves conflicts by (1) *importance* (`!important`), (2) *specificity* (`id > class > element`), and (3) *source order* (the later rule wins). In general, inline styles override embedded styles, which override linked styles, which override the browser default.

Example 1.2 — Cascade resolution

External CSS: `p { color: blue; }` Embedded CSS: `p { color: green; }` Inline: `style="color:red"`.

What colour is the paragraph?

Solution

Red. The inline declaration has the highest priority. Without the inline style it would be green if the embedded rule comes after the link tag, since a later rule of equal specificity wins.

Key Takeaways — Unit 1

- ✓ HTML structures content with tags; every page has html → head + body.
- ✓ Client-side image maps (map/area/usemap) are processed by the browser; server-side maps (ismap) send click coordinates to the server.
- ✓ Tables use table/tr/th/td with colspan/rowspan, cellpadding/cellspacing; use CSS, not tables, for layout.
- ✓ Frames split the window (frameset/frame); iframes float inside a page; use target to load links into named frames.
- ✓ Forms send name=value pairs by GET or POST; know the input types, fieldset/legend, disabled vs readonly.
- ✓ CSS can be inline, embedded (style) or linked (link); the cascade decides which rule wins.

— *End of Unit 1* —