18/5/21 Day 3

HTML Layouts - elements

<header> – defines a header for a document or a section

<nav> – defines a set of navigation links

<section> – defines a section in a document

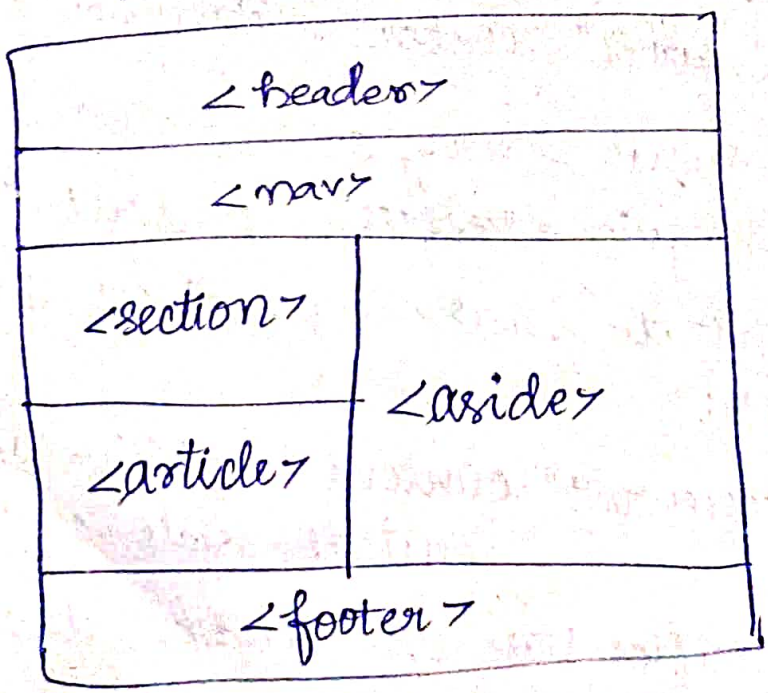
<article> – defines an independent, self-contained content.

<aside> – defines content aside from the content (like a sidebar).

<footer> – defines a footer for a document or section.

<details> – defines additional details that a user can open & close on demand

<summary> – heading for <details> element.



Layout techniques

4 techniques to create multicolumn layouts.

- CSS frameworks — w3.css or bootstrap
- CSS float property — using 'float' & 'clear' properties.
  cons: Floating elements are tied to doc flow, which may harm flexibility.
- CSS flexbox layout - it ensures that elements behave predictably when page layout must accomodate diff. screen sizes & diff. display sizes.
- CSS grid layout — grid based → rows & cols.
  ↳ easier to design a webpage.

* Responsive designing:
↓
fit to all devices.

1) Set viewport in <meta> tag
2) For an img to be responsive,
   i) <img src="" "width = 100%"> ← style =
   (or) ii)
3) To display diff images for diff screens, use <picture>
4) To make text responsive,
   <h1 style = "font-size: 10vw"> Hello </h1>
   ↓
   viewport width.

   10vw = 10% of viewport width

5) using media-Queries
6) Using frameworks like w3.css or bootstrap.

HTML computer code

1) <code>
x=5;
y=6;
z=x+y;
</code>

o/p: (default monospace)
x=5; y=6; z=x+y;

2) <pre>
<code>
x=5;
y=6;
z=x+y;
</code>
</pre>

o/p: x=5;
y=6;
z=x+y;

3) <kbd> → defines a keyboard input

Ex:

 Save document by pressing <kbd>Ctrl+S</kbd>

When u press Ctrl+s on keyboard, file saver opens.

4) <samp> — used to define sample output from a computer program.

Ex: Message from my computer:
 <samp> File not found. </samp>

o/p: Message from my computer:

File not found. → displayed in browsers default monospace font.

* HTML Computer code

1) <code>
x=5;
y=6;
z=x+y;
</code>

o/p: (default monospace)
x=5; y=6; z=x+y;

2) <pre>
<code>
x=5;
y=6;
z=x+y;
</code>
</pre>

o/p:
x=5;
y=6;
z=x+y;

3) <kbd> → defines a keyboard input

Ex:
Save document by pressing <kbd>Ctrl+S</kbd>

When u press Ctrl+s on keyboard, file saver opens.

4) <samp> — used to define sample output from a computer program.

Ex: Message from my computer:
 <samp> File not found. </samp>

o/p: Message from my computer:

File not found. → displayed in browser's default monospace font.

5) <var> → used to define a variable in programming or mathematical expression.
⇒ The content inside is typically displayed in italic.

Ex: Area of triangle is : 1/2 x <var>b</var> x <var>b</var>

Op: Area of triangle is: 1/2 x b x h

* Semantic elements – elements with a meaning.
Ex: <form>, <table>, <article> – clearly defines its content.

Non-semantic elements: ,
tells nothing about content.

Some semantic elements:

1) <section> → A thematic grouping of content, typically with a heading.
(can be nested one inside other)
2) <article> → independent, self-contained element.
3) <header> → container for introductory content or a set of navigational links.
↳ cannot be placed within <footer>, <address> or <header>
↳ i.e, not nested.

But, possible:
<article>
<header>
<h1></h1>
</header>
</article>

4) <footer> – defines a footer for a document or section
↳ Autorship info, Copyright info, contact info, back to top links etc.
→ There can be more than 1 footers in one doc.

5) <nav> → defines a set of navigation links

6) <aside> → defines some content aside from content it is placed in.

7) <figure> and <figcaption> → caption for <figure>
↓
specifies self-contained content, like illustrations, diagrams, photos, etc.

ex: <figure>
<img src="pic.jpg">
<figcaption> Fig.1 – Italy </figcaption>
</figure>

→ can be either first / last child under <figure> tag

Fig1.

8) <main> → specifies the main content of a document.

9) <summary> — visible heading for <details>

10) <time> — defines a date/time

11) <mark> – highlighted text.

9) Why Semantic elements?

A: A semantic web allows data to be shared and reused across web applic^n.s, enterprises & communities.

* HTML Styleguide: (clean & clean code)

1) Always declare Document type. <!doctype html>
2) Use lowercase for tags names i.e <p>, <body> etc.
3) Close all HTML elements

4) Use lowercase attribute names
5) Always quote attribute values
6) Always specify alt, width, height for images
7) Don't give tab spaces for indentation, use 2 spaces.
Don't give too many blank lines.
8) Never skip <title> element → for SEO readability
9) Don't omit <html>, <body> & <head>
10) close empty html elements Ex:
11) Add [lang = "en-us"] attribute, <meta charset="" >
for sure → SEO readability
12) Write viewport in <meta>
13) Short comments can be in single line, but write long comments on multiple lines.

<!-- This is short -->

<!-- This is too too
long comment
So write in
multiple lines -->

} for improving readability

14) styling in css

.p { font-size: 10px; }
→ space ✓
• body {
→ same line (bracket)
background-size: 100px;
font-size: 48px;
— — — :
— — — :
} → on new line

15) Use lowercase filenames ⇒ don't use mixedcase, some browsers are case-sensitive.
16) Save as .html or .htm → both are same.
17) Give default as "index.html" or "default.html".
Bcoz browsers search for them by default, even though not mentioned in URL.

* HTML character entities as alternative for reserved words.

1)   → non-breaking space, space that doesn't break into a new line.

entity name

entity number =

We mostly use entity names, entity no.s may not be supported by all browsers.

Result	Description	Entity name	Entity Number
	non-breaking space		
<	less than	<	<
>	greater than	>	>
&	ampersand	&	&
"	double quotes	"	"
'	single quotes	'	'
-	non-breaking hyphen	‑	
¢	cent	¢	¢
£	pound	£	£
¥	yen	¥	¥
€	euro	€	€
©	copyright	©	©
®	registered trademark	®	®

case sensitive

* Diacritical marks

Ex: à, á, â etc.

Mark	character	construct	Result
`	a	a ̀	à
/	a	a &# 769;	á
^	a	a &#l 770;	â
~	a	a &#+ 771;	ã
`	o	o&#+768;	ò
/	o	o&#+769;	ó
^	o	o&#+770;	ô
~	o	o&#+771;	õ

Mathematical symbols.

∀ → ∀
∃ → ∃
∅ → ∅
∇ → ∇
∈ → &isin
∉ → &notin
∏ → &prod
∑ → &sum

∂ → ∂
∋ → ∋

Greek:

A → Α
B → Β
Γ → Γ
Δ → Δ
E → Ε
Z → Ζ

⇒ TM → trademark ⇒ ™
← (left Arrow) ⇒ ←
→ (right ") ⇒ →
↑ (up ") ⇒ ↑
↓ (down ") ⇒ ↓
♠ (spades) ⇒ ♠
♣ (clubs) ⇒ ♣

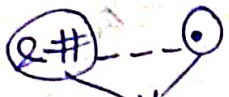
♡ → ♥
♢ → ♦
↳ (diamonds)

* HTML Emojis:

↓ not images/icons.

They are letters (chars) from UTF-8 (unicode) character set.

A → A (Ascii values)
B → B

⇒ 

says browser that it is UTF-8 code.

Emoji	value
 smile	😀
	😁
 laugh	😂
	😃
	😅
 clock	⏰
	⏳

* URLs — Uniform Resource locator.

↓ Another word for web Address.

→ can be composed of words or Ip address.

type of service (http/https) → 80 for http

scheme : // prefix . domain : port / path / filename

Ex: https : // www . w3schools . com / html / default.asp

http → hypertext transfer protocol
https → Secure " " " → Encrypted
ftp → file transfer protocol → downloading/uploading files

* URL encoding replaces non-ASCII chars with a "%" followed by hexadecimal digits.

⟹ XHTML → Extensible HTML.
→ stricter version of HTML
→ should follow HTML style guide mentioned before
→ more of XML based HTML

* HTML Forms — used to take user input
<input> tag is widely used inside <form></form>

Based on type attribute,

<input type="text"> — single line text i/p field ↓ default width=20
= "radio"
= "checkbox"
= "submit"
= "button"> → clickable button.

<label for=" "> → defines a label, which is useful for many screen readers.

⟹ For Radio-buttons, to click only 1 button at a time, use same value for name attribute.

eg:
```
<form>
  <label for="male" name="1"> Male </label>
  <input type="radio" id="male" value="Male">
  <label for="female" name="1"> Female </label>
  <input type="radio" id="female" value="Female">
</form>
```

only 1 clickable at a time → ○ Male ○ Female

for Checkbox use different names.

```
<form action="xyz.php">
  <input type="submit" value="Submit">
</form>
```
// on clicking on submit, user lands on xyz.php

* Form attributes

1) action → defines action to be performed when form is submitted.
   ↓
   if not specified, action is set to the current page.

2) target → _blank
   _self etc. → where to open form on submitting

3) method → Specifies the HTTP method to be used when submitting the form data.
   (default method = "GET")

* GET:

→ Appends form data to URL, in name/value pairs.
→ Never use it to send sensitive data since it will be visible in URL.
→ The length of URL is limited to (2048 characters).
→ Useful for form submissions when a user wants to bookmark the result.
→ GET is good for non-secure data, like query strings on Google.

* POST:

→ Appends the form data inside the body of Http request (not visible in URL)
→ no size limitations, can be used to send large amounts of data.
→ form submissions with POST cannot be bookmarked.

4) Autocomplete → Specifies autocomplete on | off.
   → If it is on, based on users previous entered values, browser automatically completes it.

Ex: <form action="/.php" autocomplete="on">

5) novalidate → specifies that input shouldn't be validated when submitted.

Ex: <form action="/action.php" novalidate>

6) name → specifies name of the form.

7) rel → specifies rel^n. ship b|w current & linked document.

Ex: <form action=" " rel="external"> "help" "license" "search", "next" etc

8) accept-charset

Ex: <form action=" " accept-charset="UTF-8">

* FORM elements

1) <form> → defines a HTML form for user input
2) <input> → defines an input control
3) <textarea> → defines a multiple line input control

Ex: <textared name="message" rows="10" cols="30"> → style="width: height:;">
Cat was playing.
</textarea>

4) <label> — label for several form elements.
5) <select> — defines a dropdown list.

Ex: <select id="cars" name="cars">
<option value="Volvo"> Volvo</option>
<option value="Saab"> Saab</option>
</select>

To display this as first, <option value="Saab" selected> Saab </option>

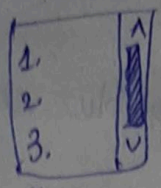
Volvo ▼
Volvo
Saab

O/p:

Saab ▼
Volvo
Saab

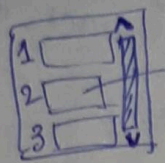
no. Visible values can be specified by using size attribute.

<select id="" name="" size="3">



⇒ for selecting more than one value among them,

<select id="" name="" size="" multiple>



→ Ctrl + select each one.

6) <button> → defines a clickable button

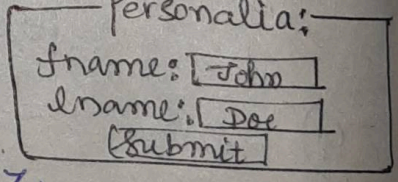
<button type="button" onclick="">click</button>

Important bcoz diff. browsers display buttons in diff ways.

7) <fieldset> and <legend>

used to group related data in a form

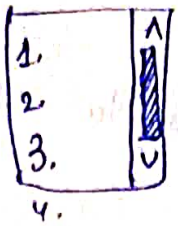
Caption for <fieldset>.



Ex:
```
<form action="">
  <fieldset>
    <legend>Personalia:</legend>
fname:<input type="text" value="John">
lname:<input type="text" value="Doe">
  </legend>
<fieldset>
```

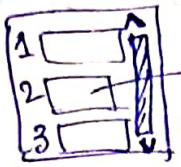
no. Visible values can be specified by using size attribute.

<select id=" " name=" " size="3">



⇒ for selecting more than one value among them,

<select id=" " name=" " size=" " multiple>



→ Ctrl + select each one.

6) <button> → defines a clickable button

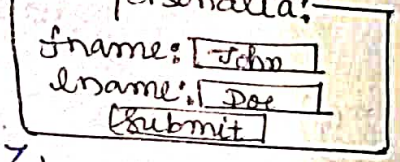
<button type="button" onclick=" ">click</button>

Important
bcoz diff. browsers display buttons in diff. ways.

7) <fieldset> and <legend>

used to group related data in a form

Caption for <fieldset>.



Ex:
```
<form action=" ">
  <fieldset>
    <legend>Personalia:</legend>
  fname:<input type="text" value="John">
  lname:<input type="text" value="Doe">
    </legend>
  </fieldset>
</form>
```

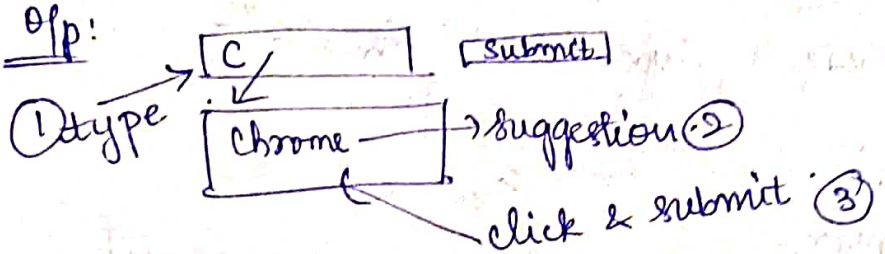
8) <datalist> → specifies a list of predefined options for an <input> element.

→ User will see a drop-down list of pre-defined options as they input the data.

* The 'list' attribute of the <input> element, must refer to 'id' attribute of <datalist>.

Ex:
```
<form action="/.php"
    <input list = "browsers"
    <datalist id = "browsers" >
        <option value = "Internet Explorer">
        <option value = "Firefox" >
        <option value = "chrome">
    </datalist>
    <input type = "submit" >
</form>
```

O/p:



9) <output> → represents result of a calculation.

⇒ To make an input field as mandatory field,

```
<form>
    <input type = " " required>
</form
```

* Input types

<input type=" " > → default type is text.

type=" "

↳ 1) button 2) checkbox 3) color 4) date 5) datetime-local 6) email 7) file 8) hidden 9) image 10) month 11) number 12) password 13) radio 14) range 15) reset 16) search

0 — 100

17) submit 18) tel 19) text 20) time 21) url
22) week

* Input attributes

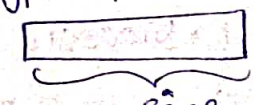
1) value → specifies initial value for an input field
2) readonly → the value cannot be modified but can be copied, it will be sent as o/p for form also
3) disabled → specifies input field is disabled or unusable or unclickable.

Ex: <input type="" disabled>

→ Value which is disabled is not sent when submitting form.

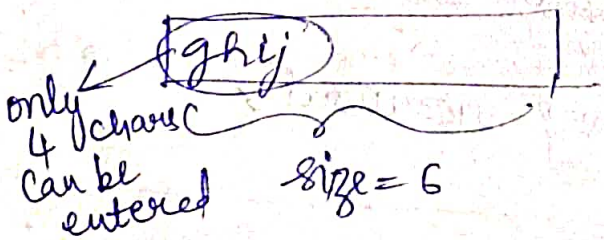
4) size → specifies visible width, in characters of an input field.

Ex: <input type="text" size="6">



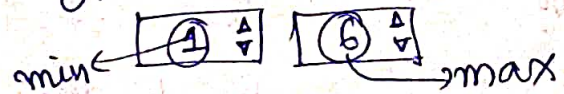
5) maxlength → max. no. of chars that can be entered

Ex: <input type="text" size="6" maxlength="4">



6) min, max ⇒ min. & max. values an input field accepts.

Ex: <input type="number" min="1" max="6">



⇒ even used for dates

7) multiple → for selecting multiples options or files

8) pattern → specifies a regular expression that the input field's value is checked against, when form is submitted.

Ex: <input type="tel" pattern="[0-9]{3}[0-9]{2}-[0-9]{3}" />

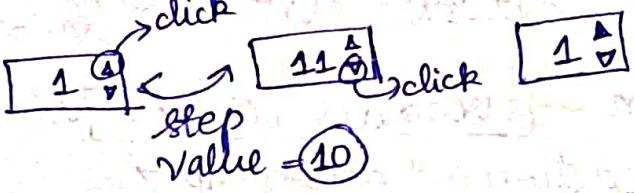
123-23-123

9) placeholder → specifies expected value of an input

10) required → mandatory field

11) step → intervals at which numbers jump

step="10"



12) autofocus → when page loads, automatically the field in which autofocus is written, is focussed (i.e, cursor blinks there)

Ex: <input type="" autofocus>

13) height & weight → specified for image input.

<input type="image" src="xyz" alt=" "
width="48" height="48">.

14) list → refers to <datalist> element.

15) autocomplete

16)

* Input form* attributes

1) form

Ex:
```
<form action=" " id="form1">
    <input>
    ---
    ---
</form>

<input type="text" form="form1">
```

not inside <form></form>
but still sent to server when submitted.

2) formaction — overrides action of <form>

Ex:
```
<form action="action.php">
    ---
    ---
    <input type="submit" formaction="page2.php"
        value="Submit">
</form>
```

on Submit → takes to page2.php not action.php

3) formenctype

```
<input type="submit"
    formenctype="multipart/form-data">
```

4) formmethod

Ex.
```
<form action=" " method="get">
    <input>
    ---
    <input type="submit" value="Submit using GET">

    <input type="Submit" formmethod="post"
        Value="submit using post">
</form>
```

Url has form info
Submit using GET → no info on url
Submit using POST

5) formtarget → <input formtarget="_blank" >

6) formnovalidate → <input formnovalidate = "formnovalidate"
value = "Submit without validation" >