Agenda

- Header,Footer,section,article,aside,nav Tags
- CSS
  - Inline
  - Internal
  - External
- Selector
- Box Model
- Position
- Display

article

- The
  HTML element represents a self-contained composition in a document, page, application, or site, which is intended to be independently distributable or reusable.
- Examples include a forum post, a magazine or newspaper article, or a blog entry, a product card, a user-submitted comment, an interactive widget or gadget, or any other independent item of content.

aside

- The
  HTML element represents a portion of a document whose content is only indirectly related to the document's main content.
- Asides are frequently presented as sidebars or call-out boxes.

header

- The
  element can define a global site header, described as a banner in the accessibility tree.
- It usually includes a logo, company name, search feature, and possibly the global navigation or a slogan.
- It is generally located at the top of the page.

footer

- The
  HTML element represents a footer for its nearest ancestor sectioning content or sectioning root element.
- It typically contains information about the author of the section, copyright data or links to related documents.

nav

- The
  HTML element represents a section of a page whose purpose is to provide navigation links, either within the current document or to other documents.

- Common examples of navigation sections are menus, tables of contents, and indexes.
- It's not necessary for all links to be contained in a
  element.
- is intended only for a major block of navigation links

section

- The
  HTML element represents a generic standalone section of a document, which doesn't have a more specific semantic element to represent it.

CSS

- CSS stands for Cascading Style Sheets.
- It makes an HTML website presentable.
- It adds style to various HTML elements.
- It helps you to define how the elements should look, where they should be placed and whether they should be displayed or not.
- Types of CSS

1. Inline
2. Internal
3. External

Inline CSS

- Use style attribute of an element to decorate it
- Simplest way to add decoration
- It is very difficult to manage because it can target only one element at a time
- It is discouraged to use inline CSS
- E.g. `<p style=”color:red;”>test</p>`

Internal CSS

- Use style tag in header section
- It can target multiple elements at a time in the given page
- It can target only one page at a time
- E.g.

```
p {
color: red;
}
```

External CSS

- Use external CSS file to hold all the rules
- Link the external CSS with link tag in header section

- E.g.

```
<link rel=”stylesheet” href=”mystyles.css”>
```

Terminology

- Rule or Ruleset:
  - Pair of CSS selector and declaration block
- Declaration block:
  - Collection of declarations
- Declaration:
  - Pair of CSS property and its value separated by colon(😃 and terminated by semi-colon(😉
- Selector:
  - Used to select one or more elements from the page

Units

- px: pixel (Picture Element)
- deg: degree
- s: Seconds
- %: with respect to its parent

CSS Selector

- used to select one or more elements from given page
- Types of selector

1. Element selector
   - Also called as type selector
   - Targets only similar type of element
   - E.g. : only paragraph will have red color

```
p { color: red; }
```

2. Multiple element selector (,)
   - Multiple type selector
   - Select multiple type of elements
   - E.g. : paragraph, h2 and h3 will have color green

```
p, h2, h3 { color: green; }
```

3. ID selector (#)
   - Select only element matching the given Id

   - Id can appear in a page only once
   - E.g. :

```
/* select only paragraph having id para1  */
p#para1 { color:red; }

/* select any element having id para1 */
#para1 { color: green; }
```

4. Class selector (.)
   - Select only element matching the given class
   - E.g.

```
/* select only paragraph having class para1 */
p.para1 { color:red; }

/* select any element having class para1 */
.para1 { color: green; }
```

5. Descendant selector (white-space)
   - Used to select child elements at any level
   - E.g.

```
/* paragraph at any level inside div will have color red */
div p { color: red; }
```

6. Child selector (>)
   - Used to select child elements which are at first level
   - Used to select only direct child elements
   - E.g.

```
/* paragraph at first level (direct) inside div will have color red */
div > p { color: red; }
```

7. Attribute selector ([])
   - Used to select an element based on the given attribute value
   - E.g.

```
/* input of type submit will have color red */
input[type=”submit”] { color: red; }
```

8. Universal selector (*)
    - Used to select all elements
    - E.g.

```
/* all elements will have font family as arial */
* { font-family: arial; }
```

9. Pseudo selector (😃
    - Used to apply CSS rules in specific condition
    - The conditions are also known as pseudo classes
    - E.g. hover, nth-child, active, focus, visited
    - E.g.

```
/* when mouse gets over on div, the color will change to red */
div:hover { color: red; }
```

CSS Box Model

- Every element in html is rendered as a box (rectangle)
- Their are 3 Properties
    1. Border
    2. Padding
    3. Margin

CSS Position

- Used to control the position
- Values are

1. static:
    - by default static is used
    - ignores top, bottom, left and right
2. relative:
    - element is aligned with respective with top,bottom,left and right
3. absolute:
    - It is absolute with the current displayed window and moves up as window scrolls
4. fixed:
    - It is fixed at the position. Even if window scrolls the element will not move from the position

CSS Display

- Used to control the display behavior of an element
- Values are

1. block:

   - considers width and height and displays elements on new line
2. inline:
   - ignores the width and height and displays in same line
3. none:
   - hides the element
4. inline-block:
   - considers width and height and displays elements on same line

CSS Float

- The float property in CSS is used to position elements to the left or right of a container, allowing text or other elements to wrap around them.
- values
  1. right:
  2. left:

CSS Flex

- Flexbox (Flexible Box Layout) is a powerful, one-dimensional layout system in CSS designed for organizing elements in rows or columns.
- It simplifies alignment, spacing, and distribution of elements, making it ideal for responsive design.
- To use Flexbox, apply display: flex; to a container.
- This makes all child elements (flex items) automatically adjust according to the rules of Flexbox.
- Flex Container and Flex Items

1. Flex Container: The parent element that holds flex items.
2. Flex Items: The child elements inside the container.

- Below properties apply to the container (display: flex;).

1. flex-direction

1. row: Default. Items are placed left to right.
2. row-reverse: Items are placed right to left.
3. column: Items are placed top to bottom.
4. column-reverse: Items are placed bottom to top.

2. justify-content

1. flex-start: Default. Items align to the start (left).
2. flex-end: Items align to the end (right).
3. center: Items are centered.
4. space-between: First item at start, last item at end, space between them.

CSS Grid

- CSS grid layout introduces a two-dimensional grid system to CSS.
- Grids can be used to lay out major page areas or small user interface elements.

- A grid is a set of intersecting horizontal and vertical lines defining rows and columns. Elements can be placed onto the grid within these column and row lines.
- we can create a grid with fixed track sizes – using pixels for example.
- This sets the grid to the specified pixel which fits to the layout you desire.
- we can also create a grid using flexible sizes with percentages or with the fr unit designed for this purpose.
- To use Grid, apply display: grid; to a container.
- This makes all child elements (grid items) automatically adjust according to the rules of grid.
- Grid Container and Grid Items

1. Grid Container: The parent element that holds grid items.
2. Grid Items: The child elements inside the container.

- Below properties apply to the container (display: grid;).

1. grid-template-columns

1. grid-template-columns : 200px 200px
   - It will create the 2 columns in the container with 200 px each
2. grid-template-columns : 1fr 1fr
   - It will create the 2 columns in the container with using 1 fraction part of the screen

1. grid-template-rows

1. grid-template-rows : 200px 200px
   - It will create the 2 rows in the container with 200 px each

CSS3 Advanced Properties

1. border-radius

   - It is a CSS property that rounds the corners of an element's outer border edge.
   - It can set a single radius to make circular corners

2. text-shadow

   - It adds shadows to text.
   - It accepts a comma-separated list of shadows to be applied to the text and any of its decorations.
   - Each shadow is described by some combination of X Y offsets from the element, blur radius, and color.

3. box-shadow

   - It adds shadow effects around an element's frame.
   - we can set multiple effects separated by commas.
   - A box shadow is described by X and Y offsets relative to the element, blur and spread radius, and color.

4. linear-gradient and radial-gradient

  - To create the most basic type of gradient, all we need is to specify two colors.
  - These are called color stops.
  - we must have at least two, but you can have as many as you want.
  - A linear gradient creates a band of colors that progress in a straight line.

5. Transform

   - It lets you rotate, scale, or translate an element.
   - It modifies the coordinate space of the CSS

6. transition-delay

   - It specifies the duration to wait before starting a property's transition effect when its value changes.

7. column-counts

   - It breaks an element's content into the specified number of columns.
   - Is a strictly positive integer describing the ideal number of columns into which the content of the element will be flowed

8. CSS Media Queries

   - Media queries are a key component of responsive design.
   - They enable conditional setting of CSS styles depending on the presence or value of device characteristics.
   - It's common to use a media query based on viewport size to set appropriate layouts on devices with different screen sizes — for example three columns on a wide screen or a single column on a narrow screen.
   - Other common examples include increasing the font size and hiding navigation menus when printing a page, adjusting the padding between paragraphs when a page is viewed in portrait or landscape mode, or increasing the size of buttons to provide a larger hit area on touchscreens.