CSS

- Cascading Style Sheet
- used to decorate the web page
    - mangae the shapes
    - mangae the sizes
    - mangae the colors
    - mangae the animation
    - mangae the mobile friendlyness
- not used for
    - adding programming logic in the website
    - designing web pages

Ways to add CSS in html document

- **browser default css**
    - by default, css which is provided by every browser
    - the browser default css will be browser specific
    - which is responsible for displaying the default tags
        - h1 will be rendered using biggest font size
        - ul and li will be rendered one after another in vertical orientation
    - generally, it is **not adivsible** to modify the default CSS
- **inline css**
    - adding the css rules inside the target tag using style attribute
    - not encouraged to use the inline style
    - limitation
        - needs to be repeated with every tag that requires modification
        - very difficult to manage / update the code
    - e.g.
        - <h1 style="color: red;">this is header1</h1>
- **internal css**
    - which is added internally to the page
    - must be added using style tag in head section
    - e.g.

```
<style>
div {
color: red;
}
</style>
```

- **external css**
    - which is added outside the page

- linked with the page using
- e.g.

```
<link rel="stylesheet" href="styles.css">
```

termonologies

- **css property**
  - used to modify the visual properties of a tag
  - e.g.
    - color, font-family, font-size, border etc.
- **css value**
  - value of the property to be modified
  - e.g.
    - red is a value
    - 20px is value
- **css declaration**
  - used to modify the visual appearance of the tag/UI
  - pair of css property and its value
  - property and its value get separated by colon (:)
  - multiple declarations are separated by semi-colon (;)
  - only one declarration need not require to be terminated with ; (semi-colon is optional)
  - e.g.

```
color: red;
font-size: 20px;
```

- **css declaration block**
  - collection of multiple declarations
  - starts with { and ends with }
  - e.g.

```
{
color: red;
font-size: 20px;
}
```

- **css selector**
  - used to select the target elements (tags)
  - e.g.

```
div {
color: red;
```

```
}
/* all divisions will be decorated with red color */
```

- **css rule**
  - also known as css ruleset
  - pair of css selector and css declaration block
  - e.g.

```
div {
color: red;
font-size: 20px;
}
```

css units

- px
  - stands for pixels
  - pixel: picture element
- percentage (%)
- em/rem
- degree

CSS Selectors types

- **type selector**
  - used to select similar type of elements
  - also known as element selector
  - e.g.

```
div {
color: red;
}
/* div selector will select only div tags */
```

- **multiple type selector**
  - also known as a combinator selector
  - uses punctuation symbol comma (,)
  - used to select multiple types of elements
  - e.g.

```
div, p, span {
font-size: 20px
}
/* all divisions, paragraphs and spans will be decorated with font size set to 20px */
```

- **id selector**
  - used to target an element based on the id attribute value

- uses punctuation symbol hash (#)
- e.g.

```
#div-3 { color: red; }
/* any element having an id div-3 will get decorated with red color */
```

div#div-3 { color: red; }
/* only div element having an id div-3 will get decorated with red color */

- **class selector**
  - used to target element(s) based on the class attribute
  - uses punctuation symbol dot (.)
  - e.g.

```
.div-3 { color: red; }
/* any element having a class div-3 will get decorated with red color */
```

div.div-3 { color: red; }
/* only div element having class div-3 will get decorated with red color */

- **universal selector**
  - used to apply rules on every possible element in the page
  - uses punctuation symbol star (*)
  - e.g.

```
    { font-family: arial }
    /* all elements will use the font as arial */
```

- **attribute selector**
  - used to select element(s) based on the value of an attribute
  - uses punctuation symbol square bracket []
  - e.g.

```
input[type="submit"] {
background-color: green;
}
/* only input having type = submit will get green backgound */
```

- **descendent selector**
  - used to select the element(s) based on the parent-child relationship
  - selects all the element(s) which are descedent [appear at any level: child, grand-child ..] of parent
  - e.g.

```
<style>
div p { color: green }
```

```
/* all paras will turn to green [a all paras are descendent of
div] */
```

</style>

para 1 inside div
para 2 inside div

 para 1 inside li
 para 2 inside li

- **child selector**
  - used to select the element(s) based on the parent-child relationship
  - selects all the element(s) which are direct child element(s) of parent
  - e.g.

> <style>
> div > p { color: green }

```
/* paras which are direct child elements of div will turn to
green*/
```

</style>

para 1 inside div
para 2 inside div

 para 1 inside li
 para 2 inside li

- **general sibling selector**
  - used to select element(s) based on the levels they appear on
  - uses punctuation symbol tild (~)
  - e.g.

> p ~ span {
> color: red;

```
}
/* select spans appearing on the same level as that of para and after paragraph */
```

- **adjacent sibling selector**
  - used to select element(s) based on the levels they appear on
  - uses punctuation symbol plus (+)
  - e.g.

```
p + span {
color: red;
}
/* select spans appearing on the same level immediately after paragraph */
```

- **pseudo selector**
  - **pseudo class**
    - keyword added to a selector that specifies a special state of the selected element(s)
    - e.g.

```
div:hover {
color: green;
}
/* div will be decorated with green color only when mouse goes on top of it */
```

  - **pseudo element**
    - is a keyword added to a selector that lets you style a specific part of the selected element(s)
    - e.g.

```
p::first-letter {
color: red;
}
```

/* only the first character will be decorated with red color */

CSS Box Model

- every element in CSS is rendered as a box with following properties
  - border
    - the bounding box for the element
    - has following properties
      - style
      - width
      - color
      - radius
  - padding
    - gap between the border and content

    - margin
        - gap outside the border

CSS Positions

- **static**
    - default position
    - decided by the code structure
    - top, left, bottom and right properties will be ignored
    - e.g.

```
button {
  position: static;
}
```

- **relative**
    - the new position (by setting top, bottom, left and right) with respect to the default position
    - e.g.

```
button {
  position: static;
  top: 10px;
  left: 10px;
}
```

- **animation**
    - transform
    - transition