**CSS3**

* Cascading Stylesheet Style
* Language which is used to make the website attractive
* Used to add decorations in html files (documents)
* Sequence of styles [CSS precedence]
  + Browser default CSS (black)
  + External CSS (red)
  + Internal CSS (green)
  + Inline CSS (blue)

**Terminology**

* CSS property: pre-defined properties provided by CSS
* **Declaration**:
  + Pair of css property and its value
  + Colon(:) is used to separate the property and its value
  + To terminate a declaration use semi-colon(;)
* **Declaration block**:
  + Collection of declarations
  + Use {} to create a block
* **Selector**:
  + Used to select a type of element(s)
* **Rule/Ruleset**:
  + Pair of selector and a declaration block

**Units**

* px: pixels
* %: with respect to its **parent**
* em: emphasis
  + by default the em is considered as 1 (browser default)
* deg: degree
* s: seconds

**CSS Types**

* Inline CSS
  + Use style attribute of a tag
  + Disadvantages:
    - Needs to be repeated with every tag having same decoration
    - Very difficult to manage (modify)
  + This is discouraged
  + E.g.

<p style=”color:red;”>test</p>

* Internal CSS
  + Use style tag in Head section
  + Advantage:
    - Simpler than inline
    - Easy to manage
  + Disadvantage:
    - Repeated in multiple pages of a website
  + E.g.

<style>

p {

color:red;

}

</style>

* External CSS
  + CSS rules whould be written outside the page (in an external file with .css extension)
  + To attach/link external css with a page use link tag in Head section

<link rel=”stylesheet” href=”<css file name/path>”>

* + Advantages:
    - Single external css file can be used across multiple pages
    - Easy to manage multiple pages at a time
* Browser Default CSS
  + Available in every browser
  + Not for the developer

**Selector Types**

* **Type / Element Selector** 
  + Used to select similar type of element(s)
  + E.g.

**p** {

color: red;

}

* + - only paragraphs will have red color
* **Multiple type/element selector (,)**
  + Used to select multiple type of elements
  + Comma (,) is used to create multiple type selector
  + E.g.

**p, div** {

color: red;

}

* + - both paragraph(s) and division(s) will have red color
* **Id Selector (#)**
  + Used to select an element having specified id
  + Hash(#) is used to create an id selector
  + E.g.

**div#div1** {

color:red

}

* + - only div having id div1 will have color red

**#product1** {

color:red

}

* + - any element having id product1 will have color red
* **Class Selector (.)**
  + Used to select element(s) having same class
  + Dot (.) is used to create a class selector
  + E.g.

**div.div1** {

color:red

}

* + - only div having class div1 will have color red

**.product1** {

color:red

}

* + - any element having class product1 will have color red
* **Descendant selector (white-space)**
  + Used when elements have relationships
  + Used to select child elements at any level (child, grand child, grand grand child ….)
  + Space is used to create descendant selector
    - E.g.

**body p** {

color:red;

}

* + - * every element inside body will have red color
* **Child selector (>)**
  + Used when elements have relationships
  + Used to select child elements at first level (direct child element(s))
  + > is used to create child selector
    - E.g.

**body > p** {

color:red;

}

* + - * Paragraph(s) declared under body will have red color
* **Universal selector (\*)**
  + Used to select All type of elements in a page(s)
  + Use \* to create universal selector
  + E.g.

**\*** {

font-family: Arial;

}

* + - all element(s) in the page will have font family set to Arial
* **Attribute selector** 
  + Used to select element(s) based on the attribute
  + Use [] to write the criteria
  + E.g.

**input[type=”submit”]** {

Color: red;

}

* + - only input having type = “submit” will have color set to red
* **Pseudo selector**

**CSS Box Model**

* Every element in html is rendered as a box
* Properties
  + Border
  + Padding: Gap inside/within the border
  + Margin:
    - Gap outside the border
    - Value: auto

**CSS Display**

* Used to control the display behavior
* Values
  + Block: new line character will be added at the end of the contents
  + Inline:
    - element(s) will be rendered on the same line
    - width and height will ignored
  + Inline-block
    - element(s) will be rendered on the same line
    - width and height will applied
  + Static: default
  + None: hide the element

**CSS Position**

* Used to decide the position of the element
* Values
  + Static:
    - default value
    - top, left, right and bottom will be ignored
  + Relative
    - Relative its static/default position
    - top, left, right and bottom will be applied by using its original (static) position
  + Absolute
    - Top, left, right and bottom will be applied by using browser’s origin
    - Gets scrolled with page
  + Fixed
    - Top, left, right and bottom will be applied by using browser’s origin
    - Never gets scrolled

**CSS Float**

* Used to decide the position (left and right)
* To clear/cancel the effect of floating use clear property

**CSS3 properties**

* Shadow:
  + Values:
    - Vertical
    - Horizontal
    - Blur
    - Color
  + Types
    - Text:
      * text-shadow : 2px 2px 5px red;
    - Box
      * box-shadow : 2px 2px 5px red;
* Border radius
  + Used to add rounded corners to any element
  + E.g. border-radius: 10px;
  + Trick:
    - Apply ½ of width to border radius to convert square element into circle shape
* Transform:
  + Used to transform an element
  + Types
    - Rotate: rotate element
      * transform: rotate(45deg);
    - Scale: scale element (zoom)
      * transform: scale(2);
    - Translate: move position
      * transform: translate(10px, 10px);
* Transition:
  + Used to animation (duration in seconds)
  + E.g. transition: all 2s;
* Gradients
  + Used to add multiple colors (blended)
  + Types
    - Linear
      * E.g. background: linear-gradient(red, yellow);
    - Radial
      * E.g. background: radial-gradient(red, yellow);
* Columns
  + Used to distribute the element contents in multiple columns
  + E.g. column-count: 3;
* At (@) rules:
  + Start with @ symbol
  + **Font**:
    - Used to load custom fonts
    - E.g.

**@font-face** {

font-family: <family name>;

src: url(‘<path>’);

}

p {

font-family: <family name>;

}

* + **Media Query**
    - Used to create responsive website
    - A website is having an ability to optimize output according to the device width
      * Desktop
      * Tablet
      * Mobile
    - E.g.

@media screen and (max-height:768px) {

h1 {

color: red;

}

}

* + - * will have h1 with color red only on mobile devices

**Bootstrap**

* framework (having HTML, CSS and JS) used to design responsive web sites
* free and open source
* large community
* developed by Twitter
* steps to use bootstrap
  + download the bootstraps files (getboostrap.com) – v3.3.7
  + copy css, fonts and js folders into the application
  + load the css files
    - <link rel=”stylesheet” href=”css/boostrap.css”>
    - <link rel=”stylesheet” href=”css/boostrap-theme.css”>
* Containers
  + container: aligns contents to the center of the page
  + container-fluid: uses the page width
* Grid System:
  + Every element is divided into 12 equal columns
  + Classes
    - col-lg-\* : larger displays (>= 1200 px)
    - col-md-\*: medium displays (>= 992px and < 1200px)
    - col-sm-\*: smaller displays (>768px and < 992px)
    - col-xs-\*: extra small displays (< 768px)