# There are three ways to add CSS to HTML:

1. **Inline** - by using the style attribute inside HTML elements
2. **Internal -** by using <style> element in the <head> section
3. **External** – by using a <link> element to link to an external CSS file (the most common way)

Example:

**Inline**:

<h1 style="color:blue;">A Blue Heading</h1>

**Internal:**

**<head>**

**body{background-color: white}**

**</head>**

**External:**

**<link rel =” stylesheet” href =” style.css”>**

# HTML id Attribute

Class and Id (https://www.w3schools.com/html/tryit.asp?filename=tryhtml\_id\_class)



@charset "UTF-8"; // In css file, define the character encoding used in the document as utf-8

The four links states are:

* a:link - a normal, unvisited link
* a:visited - a link the user has visited
* a:hover - a link when the user mouses over it
* a:active - a link the moment it is clicked

# Background-image

* background-image: url(url);
  + eg: background-image: url(image.png);
* Tiling a background image:
  + Background-repeat: type; // repeat(default), repeat-x, repeat-y, no-repeat, round, or space.
    - Eg: background-repeat: repeat-x;
* Attaching the background image:
  + Background-attachment: type; //scroll(default), fixed, or local
    - Eg: background-attachment: fixed; /\* watermarks \*/
* Setting the background image Position:
  + Background-position: horizontal vertical;
    - Eg: background-position: 10% 20%; /\* 10% of the width of the element from the lefr edge of the background, and 20% of the element’s height from the background’s top edge \*/
    - Eg: background-position: 30px; /\* 30 pixels from the element’s left edge and down from the top edge \*/
    - Eg: background-position: right bottom; /\* bottom-right corner of the element.
* Clipping an background:
  + Background-clip:type; ; //content-box, padding-box, or border-box
    - Eg: background-clip: content-box;
  + Background-origin:type
    - Eg: background-origin: content-box;
* Sizing an background image:
  + Background-size: width height;
    - Eg: background-size: 300px 200px;
    - Eg: background-size: auto 200px;
* Background property:
  + Background: color url(url) position /size repeat origin clip;
    - Eg: red url(image.png) center center / 10% 10% no-repeat fixed content-box content-box;
* Adding multiple backgrounds:
  + Background: background1, background2, …

# Border (border.html)

* Setting Border Width and Color
  + border-width: top right bottom left;
    - Eg: border-width: 10px 20px; /\*setting with top and bottom border 10px, right and left border 20px. \*/
  + border-side-width: width; // side: top right bottom left
    - eg: border-bottom-width: 10px;
  + border-color: top right bottom left;
    - Eg: border-color: gray black black gray;
  + Border-side-color: color;
    - Eg: border-top-color: blue;
* Setting the Border Design
  + border-style: top right bottom left;
    - Eg: border-style: double solid; /\*setting top and bottom border style double and right and left border style solid. \*/
  + border-side-style: style;
    - Eg: border-top-style: dashed;
  + border: width style color;
    - Eg: border: 2px solid blue;
  + border-side: width style color;
    - Eg border-top: 2px solid blue;
* Creating Rounded Corners
  + border-radius: top-left top-right bottom-right bottom-left
    - Eg: border-radius: 10px 20px; /\*top-left and bottom-right with 10px, and top-right and bottom-left with 20px \*/
  + border-corner-radius: radius; /\*Setting the curvature for only one corner. Corner can be replaced with either top-left, top-right, bottom-right, or bottom-left.
    - Eg: border-top-left:10px;
  + Creating an elongated corner:
    - Eg: border-radius: 50px/20px;
    - Eg: border-radius: 15%;
    - border-corner-radius: horizontal vertical;
      * eg:border-top-left-radius: 20px 15px;
* Applying a border image:
  + Border-image: url(url) slice stretch;

stretch: The slices are stretched to fill each side.

repeat: The slices are tiled to fill each side.

round: The slices are tiled to fill each side; if they don’t fill the sides with an integer number of tiles, the slices are rescaled until they do.

space: The slices are tiled to fill each side; if they don’t fill the sides with an integer number of tiles, extra space is distributed around the tiles.

* + - Eg: border-image: url(frame.png) 5% 10% stretch; /\*
    - Eg: border-image:rul(frame.png) 5 stretch;

# Shadow (Shadow.html)

* Text Shadow:
  + texst-shadow: color offsetX offsetY blur; /\*offsetX and offsetY are the distances of the shadow from the text in the horizontal and vertical directions. \*/
    - eg: text-shadow: red 10px 5px 8px;
  + Text-shadow: shadow1, shadow2, shadow3, …;
* Box Shadow:
  + box-shadow: color offsetX offsetY blur;
    - eg: box-shadow: red 10px 20px 10px; /\*shadow exists 10px on the right and 20px on the bottom \*/
  + box-shadow: shadow1, shadow2, …
    - eg: box-shadow: red 10px 20px 10px , green -10px 20px 10px;
  + box-shadow: inset color offsetX offsetY blur;
    - eg: box-shadow: inset red 10px 10px 20px;
* Reflection:
  + -webkit-box-reflect: direction offset mask-box-image;
    - eg: -webkit-box-reflect: below 10px; /\* reflection is 10 pixels below the inline image \*/

# Gradient

* Linear Gradient
  + Linear-gradient (color1, color2, …);
    - eg: background: linear-gradient(red, yellow, blue);
  + Linear-gradient (direction, color1, color2, ...);
    - eg: background: linear-gradient ( to left, red, white, blue);
    - eg: background: linear-gradient (60deg, red, white, blue);
* Linear Gradient Color Stop
  + Linear-gradient (direction, color-stop1, color-stop2,…);
    - eg: background: linear-gradient(to left, red 25% , white 25% , blue 70%);
* Radial gradient
  + Radial-gradient (shape size at position, color-stop1, color-stop2, … );

/\* shape: ellipse(default), circle

Size: farthest-corner(default), farthest-side, closest-corner, and closest-side \*/

* + - eg: background: radial-gradient (circle closet-corner at center center, red, white, blue);
* Repeating a gradient
  + Repeating-linear-gradient (direction, color1, color2 stop …); /\*always need the stop point for the last color \*/
    - eg: background: repeating-linear-gradient (red, yellow, blue 20%);
  + Repeating-radial-gradient (direction, color1, color2,… );
    - eg: background: repeating-Radial-gradient (circle closest-corner at center center, red 10%, yellow 20%, blue 30%);

# Semi-Transparent object

* opacity: value; /\*from 0 (completely transparent) up to 1 (completely opaque).\*/
  + eg: opacity: 0.7;

# Transforming Page Objects (translate.html)

* transform: effect( params);
  + eg: transform: rotate(30deg);
  + eg: transform: scale(0.8,0.5);
* transform function:
  + translate ( offX, offY)
  + translateX ( offX )
  + translateY ( offY )
  + scale ( X, Y)
  + scaleX (X)
  + scaleY (Y)
  + skew( angleX, angleY)
  + skewX( angleX)
  + skewY( angleY)
  + rotate( angle)
  + matrix(n ,n ,n ,n ,n,n)
* multiple effect:
  + eg: transform: skew(30deg) scale(0.5, 0.5);

# Mobile Web

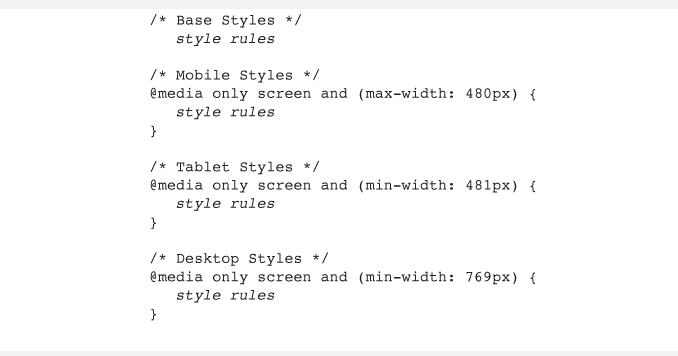
* Media Queries:
  + Media=” devices”
  + Three ways of specifying media devices:
    - <link href =” style.css” media=”print, projection”/ >
    - @ media print, projection {h1 {font-size:2em;}}

@media screen, tv {h1{font-size:2em;}}

* + - @import url(“style.css”) screen, projection;
  + Media Type:
    - * all (default): All output devices
      * braille: Braille tactie feedback devices
      * embossed: paged Braille printers
      * handheld: Mobile devices with small screens and limited bandwidth
      * print: printers
      * projection: projectors
      * screen: computer screens
      * speech: speech and sound synthesizers, and aural browsers
* Media Queries and Device Features.
  + media=” devices and | or (feature : value ) ”;
    - eg: @media screen and (device-width:320px){ h1{front-size:2em;}}
    - eg: @import url(“sstyle.css”) screen and (device-width:320px)

/\*only to the screen width with 320px\*/

* Applying media queries to a style sheet



* Viewports and Device Width
  + Mobile devices have two types of viewports:
    - visual viewport: displaying the web page content that fits within a mobile screen.
    - layout viewport: containing the entire content of the page, some of which may be hidden from the user.
  + <meta name=” viewport” content=” properties” />
    - Eg: <meta name=” viewport” content=” width=device-width, initial-scale=1” />

# Pull Down Menu (pulldownmenu.html)

# Flexible Boxes

* define an element as a flexbox
  + display: flex
  + display: inline-flex
* Setting the flexbox flow
  + flex-direction: direction ;
    - eg: flex-direction: column;
    - direction:
      * row (default): from left to right
      * column: from top and moving downward
      * row-reverse: lay out the items bottom-to-top
      * column-reverse: lay out the items right-to-left
  + flex-wrap: type; /\*nowrap(default), wrap, wrap-reverse \*/
    - eg: flex-wrap: wrap-reverse;
  + flex-flow: direction wrap;
    - eg: flex-flow: row wrap;
* Flex Basis: The size defines the initial size of the item before the browser attempts to fit it to the flexbox.
  + flex-basis: size ;
    - eg: flex-basis: 200px;
* flex growth: the rate at which a flex item grows from its basis size
  + flex-grow:value;
    - eg: flex-grow: 3;
* shrink rate:
  + flex-shrink: value;
    - eg: flex-shrink: 3;
* flex property:
  + flex: grow shrink basis;
    - eg: flex: 3 1 auto;
  + flex: keyword;
    - eg: flex: auto;
    - keyword:
      * auto: equivalent to (flex: 1 1 auto; )
      * initial: equivalent to (flex: 0 0 auto; )
      * none: ( flex: 0 0 auto ; )
      * inherit: use to inhert the flex values of its parent element.
* Applying a flexbox layout