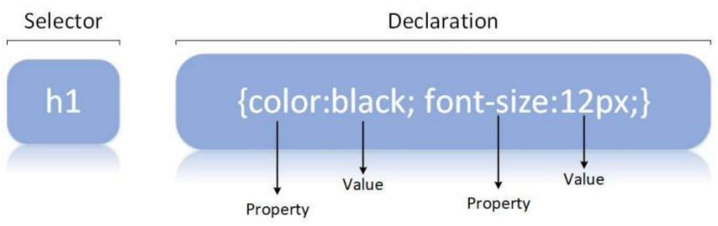
What is CSS?

- CSS is the language we use to style a Web page.
- CSS stands for Cascading Style Sheets
- CSS describes how HTML elements are to be displayed on screen, paper, or in other media

css syntax



CSS selectors are used to "find" (or select) the HTML elements you want to style.

How To Add CSS

There are three ways of inserting a css:

- External CSS
- Internal CSS
- Inline CSS

External CSS

With an external style sheet, you can change the look of an entire website by changing just one file!

```
<link  href="./mystyle.css" rel="stylesheet">
```

Internal CSS

An internal style sheet may be used if one single HTML page has a unique style.

Ex: `<style>`

```
h1 {
  color: red;
}
</style>
```

Inline CSS

An inline style may be used to apply a unique style for a single element

```
Ex: <p style="color:red;">This is a paragraph.</p>
```

CSS Comments

Comments are used to explain the code, and may help when you edit the source code at a later date.

Css comments start with /* & end with */

Ex: `/* This is a syntax for comment */`

CSS Colors

1. Name

Ex: color: blue;

2. Rgb

-- Each parameter (red, green, and blue) defines the intensity of the color between 0 and 255

Ex: rgb(0, 0, 0)

3. Rgba

-- RGBA color values are an extension of RGB color values with an alpha channel - which specifies the opacity for a color.

Ex: rgba(0,0,0,1)

4. HEX

--color can be specified using a hexadecimal value in the form:

-- *#rrggbb* -> Where rr (red), gg (green) and bb (blue) are hexadecimal values varies between 00 and ff

Ex: #ffffff.

CSS Backgrounds

The CSS background properties are used to add background effects for elements.

- `background-color`
- `background-image`
- `background-repeat`
- `background-position`
- `background-attachment`
- `background` (shorthand property)

1. background-color

```
Ex: background-color: blue;
```

2. background-image

```
Ex: background-image: url("./image.png");
```

3. background-repeat

By default, the `background-image` property repeats an image both horizontally and vertically.

```
Ex: background-repeat: repeat-x;  /* Repeat only horizontally */

     background-repeat: repeat-y;  /* Repeat only vertically */

     background-repeat: no-repeat;  /* the bg image is only shown once */
```

4. Background-position

The `background-position` property is used to specify the position of the background image.

```
Ex:  background-position: 100px 300px;

     background-position: center center;
```

5. background-attachment

The `background-attachment` property specifies whether the background image should scroll or be fixed (will not scroll with the rest of the page):

```
Ex: background-attachment: fixed;
```

6. Background (shorthand property)

Ex: `background: #fff url("img_tree.png") no-repeat right top;`

CSS Margins

Margins are used to create space around elements, outside of any defined borders.

- `margin-top`
- `margin-right`
- `margin-bottom`
- `margin-left`
- `Margin` (shorthand property)

Ex: **margin: 25px 20px 35px 10px;**

CSS Padding

The CSS `padding` properties are used to generate space around an element's content, inside of any defined borders

- `padding-top`
- `padding-right`
- `padding-bottom`
- `padding-left`
- `Padding` (shorthand property)

Ex: **padding: 25px 20px 35px 10px;**

CSS Borders

- `border-width --> border-width: 5px;`
- `border-style --> border-style: dotted;`
- `border-color --> border-color: red;`
- `Border` (shorthand property) --> `border: 5px solid red;`
- `Border-radius --> border-radius: 5px;`

CSS Box Model

All HTML elements can be considered as boxes.
It consists of: margins, borders, padding, and the actual content.

- **Content** - The content of the box, where text and images appear
- **Padding** - Clears an area around the content. The padding is transparent
- **Border** - A border that goes around the padding and content
- **Margin** - Clears an area outside the border. The margin is transparent

CSS Outline

An outline is a line drawn outside the element's border.

- `Outline-width -->  Outline-width: 5px;`
- `Outline-style  -->  Outline-style: dotted;`
- `Outline-color  -->    Outline-color: red;`
- `Outline` (shorthand property) --> `Outline: 5px solid red;`
- `Outline-offset  --> a space b/w outline & border`

CSS Height, Width

The CSS `height` and `width` properties are used to set the height and width of an element.

Ex:
```
h1 {
  height: 200px;
  width: 50%;
  background-color: blue;
}
```

CSS Text

1. Text color

Ex: color: red;

2. Text Alignment

The `text-align` property is used to set the horizontal alignment of a text.

A text can be left or right aligned, center

```
text-align: left;
text-align: right;
text-align: center;

Ex: h1 {
 text-align: left;
}
```

3. Text Decoration

- `text-decoration-line` --> `text-decoration-line: underline;`
- `text-decoration-color` --> `text-decoration-color: red;`
- `text-decoration-style` --> `text-decoration-style: dotted;`
- `text-decoration-thickness` --> `text-decoration-thickness: 5px;`
- `text-decoration` (shorthand) --> `text-decoration: underline red double 5px;`

4. Text Transformation

The `text-transform` property is used to specify uppercase and lowercase letters in a text.

```
text-transform: uppercase;

text-transform: lowercase;

text-transform: capitalize;
```

5.Text Spacing

- `text-indent  -->  text-indent: 50px;`
- `letter-spacing -->  letter-spacing: 5px;`
- `line-height  -->  line-height: 1.8;`
- `word-spacing --> word-spacing: 10px;`

CSS Fonts

1. Font-family

```
font-family: "Times New Roman", Times, serif;
```

2. Font-style

```
font-style: italic;
```

3. Font-weight

```
font-weight: bold;
```

4. Font variant

```
font-variant: small-caps;
```

5. Font-size

The `font-size` property sets the size of the text.

```
font-size: 16px;
```

CSS Tables

```
table, th, td {
  border: 1px solid;
}

table {
  border-collapse: collapse;
}
```

CSS Overflow

The CSS `overflow` property controls what happens to content that is too big to fit into an area.

The `overflow` property has the following values:

- `visible` - Default. The overflow is not clipped. The content renders outside the element's box
- `hidden` - The overflow is clipped, and the rest of the content will be invisible
- `scroll` - The overflow is clipped, and a scrollbar is added to see the rest of the content
- `auto` - Similar to `scroll`, but it adds scrollbars only when necessary

```
Ex: div {
  overflow: scroll;
}
 div {
  overflow: auto;
}
div {
  overflow: hidden;
}
```

display Property

The `display` property specifies if/how an element is displayed.

1. Display: Inline
2. Display: Block
3. Display: None
4. `visibility: hidden;`
5. `display: inline-block;`

Pseudo-classes

A pseudo-class is used to define a special state of an element.

```
selector:pseudo-class {
  property: value;
}

Ex:

a:link {
  color: #FF0000;  /* unvisited link */
}

/* visited link */
a:visited {
  color: #00FF00;
}

/* mouse over link */
a:hover {
  color: #FF00FF;
}

/* selected link */
a:active {
  color: #0000FF;
}

.mainblock p :first-child {
  color: red;
}

.mainblock p :last-child {
  color: blue;
}

.mainblock p :nth-child(4) {
  color: green;
}
```

CSS Forms

- `input[type=text]` - will only select text fields
- `input[type=password]` - will only select password fields
- `input[type=number]` - will only select number fields

CSS Icons

Import 2 methods:

```
1. <link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/6.4.2/css/all.min.css">

2. Download - <link rel="stylesheet"
href="../fontawesome/css/al.css">
```

```
Ex:  <i class="fa-solid fa-phone-flip"></i>
```

position Property

The `position` property specifies the type of positioning method used for an element.

There are five different position values:

- `static`
- `relative`
- `Absolute`
- `fixed`
- `sticky`

position: static;

HTML elements are positioned static by default.

Static positioned elements are not affected by the top, bottom, left, and right properties.

position: relative;

An element with `position: relative;` is positioned relative to its normal position.

Setting the top, right, bottom, and left properties of a relatively-positioned element will cause it to be adjusted away from its normal position. Other content will not be adjusted to fit into any gap left by the element.

Ex:
```
div.relative {
  position: relative;
  left: 30px;
}
```

position: fixed;

An element with `position: fixed;` is positioned relative to the viewport, which means it always stays in the same place even if the page is scrolled. The top, right, bottom, and left properties are used to position the element.

Ex:
```
div.fixed {
  position: fixed;
  bottom: 0;
  right: 0;
}
```

position: absolute;

An element with `position: absolute;` is positioned relative to the nearest positioned ancestor (instead of positioned relative to the viewport, like fixed).

Ex: div.absolute {

position: absolute;

top: 80px;

}

position: sticky;

An element with `position: sticky;` is positioned based on the user's scroll position.

A sticky element toggles between `relative` and `fixed`, depending on the scroll position. It is positioned relative until a given offset position is met in the viewport - then it "sticks" in place (like position:fixed).

```
div.sticky {
  position: sticky;
  top: 0;
}
```

The z-index Property

When elements are positioned, they can overlap other elements.

The `z-index` property specifies the stack order of an element (which element should be placed in front of, or behind, the others).

Note: `z-index` only works on positioned elements (position: absolute, position: relative, position: fixed, or position: sticky) and flex items (elements that are direct children of display: flex elements).

Ex:

```
 <img src="img_tree.png">

<p>Because the image has a z-index of -1, it will be placed behind the text.</p>

img { position: absolute;

  left: 0px;

  z-index: -1;

}
```

The float Property

The `float` property is used for positioning and formatting content e.g. let an image float left to the text in a container.

The `float` property can have one of the following values:

- `none` - The element does not float (will be displayed just where it occurs in the text). This is default
- `left` - The element floats to the left of its container
- `right` - The element floats to the right of its container

Ex: `float: right;`

CSS Flexbox

The Flexible Box Layout Module, makes it easier to design flexible responsive layout structure without using float or positioning.

- `flex-direction` - property defines in which direction the container wants to stack the flex items.
- `flex-wrap` - property specifies whether the flex items should wrap or not.
- `justify-content` - property is used to align the flex items:
- `align-items` - property is used to align the flex items.

CSS Media Query

Media query is a CSS technique introduced in CSS3.

It uses the `@media` rule to include a block of CSS properties only if a certain condition is true.

```css
@media only screen and (max-width: 600px) {
  body {
    background-color: lightblue;
  }
}

@media only screen and (min-width: 600px) {
  body {
    background-color: lightblue;
  }
}
```

```css
@media only screen and (max-width: 600px) and (min-width: 400px)  { }
```

CSS Box Shadow & Linear Gradiant

Ex:

```css
box-shadow: 10px 10px 8px 10px #888888;
 background-image: linear-gradient(red, yellow); /* Direction - Top to Bottom ( default) */
 background-image: linear-gradient(to right, red , yellow); /* Direction - left to right */
 background-image: linear-gradient(0deg, red, yellow);   /* Direction -  (to top) */
 background-image: linear-gradient(90deg, red, yellow); /* Direction -  (to left) */
 background-image: linear-gradient(180deg, red, yellow); /* Direction -  (to bottom) */
 background-image: linear-gradient(270deg, red, yellow); /* Direction -  (to right) */
```

CSS Transition

CSS transitions allows you to change property values smoothly, over a given duration.

- `transition-delay`
- `transition-duration`
- `transition-property`
- `transition-timing-function`
- `Transition`

transition	A shorthand property for setting the four transition properties into a single propert
transition-delay	Specifies a delay (in seconds) for the transition effect
transition-duration	Specifies how many seconds or milliseconds a transition effect takes to complete
transition-property	Specifies the name of the CSS property the transition effect is for
transition-timing-function	Specifies the speed curve of the transition effect

The `transition-timing-function` property specifies the speed curve of the transition effect.

- `ease` - specifies a transition effect with a slow start, then fast, then end slowly (this is default)
- `linear` - specifies a transition effect with the same speed from start to end
- `ease-in` - specifies a transition effect with a slow start
- `ease-out` - specifies a transition effect with a slow end
- `ease-in-out` - specifies a transition effect with a slow start and end
- `cubic-bezier(n,n,n,n)` - lets you define your own values in a cubic-bezier function

CSS Transform

`transform` property you can use the following 2D transformation methods:

- `rotate()`
- `scale()`
- `skew()`
- `translatex()`
- `translatey()`
- `translate()`

CSS Animations

- `@keyframes`
- `animation-name`
- `animation-duration`
- `animation-timing-function`
- `animation-delay`
- `animation-iteration-count`
- `animation-direction`
- `animation-fill-mode`
- `Animation`

Property	Description
@keyframes	Specifies the animation code
animation	A shorthand property for setting all the animation properties
animation-delay	Specifies a delay for the start of an animation
animation-direction	Specifies whether an animation should be played forwards, backwards or in alternate cycles
animation-duration	Specifies how long time an animation should take to complete one cycle
animation-fill-mode	Specifies a style for the element when the animation is not playing (before it starts, after it ends, or both)
animation-iteration-count	Specifies the number of times an animation should be played
animation-name	Specifies the name of the @keyframes animation
animation-timing-function	Specifies the speed curve of the animation

The `animation-direction` property specifies whether an animation should be played forwards, backwards or in alternate cycles.

The animation-direction property can have the following values:

- `normal` - The animation is played as normal (forwards). This is default
- `reverse` - The animation is played in reverse direction (backwards)
- `alternate` - The animation is played forwards first, then backwards
- `alternate-reverse` - The animation is played backwards first, then forwards

The `animation-timing-function` property specifies the speed curve of the animation.

The animation-timing-function property can have the following values:

- `ease` - Specifies an animation with a slow start, then fast, then end slowly (this is default)
- `linear` - Specifies an animation with the same speed from start to end
- `ease-in` - Specifies an animation with a slow start
- `ease-out` - Specifies an animation with a slow end
- `ease-in-out` - Specifies an animation with a slow start and end

The `animation-fill-mode` property specifies a style for the target element when the animation is not playing (before it starts, after it ends, or both).

The animation-fill-mode property can have the following values:

- `none` - Default value. Animation will not apply any styles to the element before or after it is executing
- `forwards` - The element will retain the style values that is set by the last keyframe (depends on animation-direction and animation-iteration-count)
- `backwards` - The element will get the style values that is set by the first keyframe (depends on animation-direction), and retain this during the animation-delay period
- `both` - The animation will follow the rules for both forwards and backwards, extending the animation properties in both directions

Ex:

```
div {
  width: 100px;
  height: 100px;
  background-color: red;
  position: relative;
  animation-name: example;
  animation-duration: 4s;
  animation-iteration-count: 2;
  animation-direction: alternate-reverse;
}

@keyframes example {
  0%   {background-color:red; left:0px; top:0px;}
  25%  {background-color:yellow; left:200px; top:0px;}
  50%  {background-color:blue; left:200px; top:200px;}
  75%  {background-color:green; left:0px; top:200px;}
  100% {background-color:red; left:0px; top:0px;}
}
@keyframes example {
  from {background-color: red;}
  to {background-color: yellow;}
}

  animation: example 5s linear 2s infinite alternate;
```

BOOSTRAP

Bootstrap is a popular open-source front-end framework used for web development.

It provides a collection of HTML, CSS, and JavaScript tools, templates, and components that help developers create responsive and visually appealing websites and web applications quickly and efficiently.

- Tables
- Alerts
- Buttons
- Spinners
- List Groups
- Cards
- Dropdowns
- Collapse
- Navbar
- Forms
- Model
- Accordian
- breadcrumb
- carousel
- Containers
- Fluid Containers
- Images with .img-fluid
- Pagination
- Text alignment
- Toasts
- Tooltip
- Display -> d-block, d-inline, d-none
- flex
- grid

Flex

```
<div class="d-flex">.I'm a flexbox container!</div>
<div class="d-flex justify-content-center">...</div>
<div class="d-flex justify-content-start">...</div>
<div class="d-flex justify-content-end">...</div>
<div class="d-flex justify-content-between">...</div>
<div class="d-flex justify-content-around">...</div>
<div class="d-flex justify-content-evenly">...</div>
<div class="d-flex align-items-start">...</div>
<div class="d-flex align-items-end">...</div>
<div class="d-flex align-items-center">...</div>
<div class="d-flex flex-wrap">  ...</div>
<div class="d-flex flex-nowrap">  ...</div>
<div class="d-flex flex-row-reverse">...</div>
```

Grid

```
  <div class="row">
   <div class="col-4 col-lg-2">      1 of 3    </div>
<div class="col-4 col-lg-2">      Variable width content    </div>
 <div class="col-4 col-lg-2">      3 of 3    </div>  </div>
</div>
  Col-sm: : width - 767px,
 Col-md: width - 991px,
  Col-lg: : width -575px,
```