**2 ] CSS**

**2.1 Basic of CSS**

2.2.1 CSS introduction

* CSS stands for Cascading Style Sheets.
* To style web page.
* External stylesheets are saved using .css .
* CSS describes how HTML elements are to be displayed on screen, paper, or in other media.
* Why Use CSS? : CSS is used to define styles for your web pages, including the design, layout and variations in display for different devices and screen sizes.
* With an external stylesheet file, you can change the look of an entire website by changing just one file.

2.2.2 External style sheet, Internal style sheet, Inline style sheet

External style sheet :

* Can change look of entire website by changing just one file.
* HTML page must include a reference to external style sheet using <link> element , inside head section.

Ex: <link rel="stylesheet" href="style.css">

Internal style sheet :

* Used to unique style on single HTML page.
* Defined inside the <style> element, inside Head section.

Ex:

<head>  
<style>  
body {  
  background-color: red;  
}  
</style>  
</head>

Inline style sheet :

* Used to apply a unique style for a single element.
* Using style attribute.
* Style attribute can contain any CSS property.

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

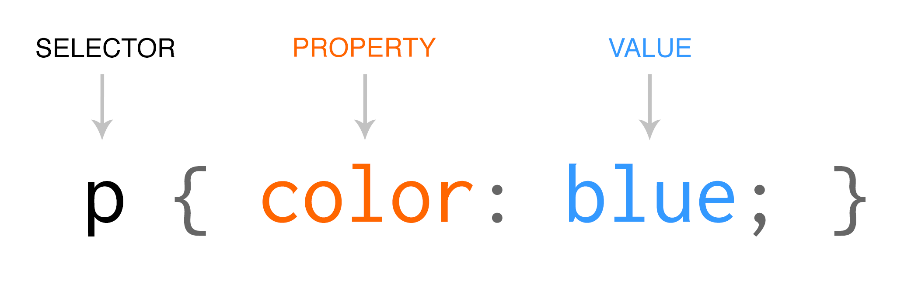
All the styles in a page will "cascade" into a new "virtual" style sheet by the following rules, where number one has the highest priority:

1. Inline style
2. External and internal style sheets (whoever comes last in head section)
3. Browser default

2.2.3 CSS Syntax

A CSS rule consists of a selector and a declaration block.

* Selector : Points the HTML element you want to style.
* Declaration block : Contains one or more declarations separated by semicolons. Each declaration includes a CSS property name and a value, separated by a colon.



2.2.4 CSS Selector

* Selectors are patterns used to select the element(s) you want to style.

|  |  |  |
| --- | --- | --- |
| Selector | Example | Description |
| .class | .name | Selects all elements with class="name" |
| #id | #fname | Selects the element with id="fname" |
| \* | \* | Selects all elements |
| element | p | Selects all <p> elements |
| element.class | p.name | Selects all <p> elements with class="name" |
| element,element | div,p | Selects all <div> elements and all <p> elements |
| [attribute] | [target] | Selects all elements with a target attribute |
| [attribute=value] | [target="\_blank"] | Selects all elements with target="\_blank" |
| :active | a:active | Selects the active link |
| ::after | p::after | Insert something after the content of each <p> element |
| ::before | p::before | Insert something before the content of each <p> element |
| :checked | input:checked | Selects every checked <input> element |
| :default | input:default | Selects the default <input> element |
| :disabled | input:disabled | Selects every disabled <input> element |
| :empty | p:empty | Selects every <p> element that has no children (including text nodes) |
| :first-child | p:first-child | Selects every <p> element that is the first child of its parent |
| :focus | input:focus | Selects the input element which has focus |
| :hover | a:hover | Selects links on mouse over |
| :link | a:link | Selects all unvisited links |
| :read-only | input:read-only | Selects input elements with the "readonly" attribute specified |
| :read-write | input:read-write | Selects input elements with the "readonly" attribute NOT specified |
| :valid | input:valid | Selects all input elements with a valid value |
| :visited | a:visited | Selects all visited links |

2.2.5 CSS basic property

**Backgrounds :** used to add background effects for elements.

* background-color : Specifies background color.

Ex:

body {

background-color: lightblue;

}

* background-image : Specifies an image. By default, the image is repeated so it covers the entire element.

Ex :

body {  
  background-image: url("paper.gif");  
}

* background-repeat : By default, the background-image property repeats an image both horizontally and vertically.

For only horizontally : background-repeat: repeat-x;

For only vertically : background-repeat: repeat-y;

For only ones : background-repeat: no-repeat;

* background-position : used to specify the position of the background image.

Ex: background-position: right top;

* background-attachment : specifies whether the background image should scroll or be fixed.

For fix : background-attachment: fixed;

For scroll : background-attachment: scroll;

* background – shorthand : To specity all background properties in one single property.

Ex :

Normal :

body {  
  background-color: #ffffff;  
  background-image: url("img\_tree.png");  
  background-repeat: no-repeat;  
  background-position: right top;  
}

Shorthand :

body {  
  background: #ffffff url("img\_tree.png") no-repeat right top;  
}

order : color image repeat attachment position

**Borders** :

Allow you to specify the style, width, and color of an element's border.

* border-style : To specifies what kind of border to display.

dotted - Defines a dotted border

dashed - Defines a dashed border

solid - Defines a solid border

double - Defines a double border

none - Defines no border

hidden - Defines a hidden border

Ex : p {border-style: dotted;}

* border-Width : specifies the width of the four borders.

For all sides

Ex : p {border-style: dotted; border-width:2px;}

Top and bottom and sides have different width

Ex: p {border-style: solid; border-width: 5px 20px;}

All 4 sides have different width

Ex: p{border-style: solid; border-width: 25px 10px 4px 35px;}

* border-color : used to set the color of the four borders

Ex : p {border-style: solid; border-color: red;}

Specific Side Colors

Ex : p {border-style: solid; border-color: red green blue yellow;}

* CSS Border - Shorthand Property

Ex : p {

border: 5px solid red;

}

Width – style – color

* border-radius : used to add rounded borders to an element

Ex : p {  
  border: 2px solid red;  
  border-radius: 5px;  
}

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

* individual sides :

margin-top

margin-right

margin-bottom

margin-left

Ex : p { margin-top: 100px; margin-bottom: 100px; margin-right: 150px; margin-left: 80px; }

* Shorthand property :

Top – right – bottom – left

Ex: p {margin: 25px 50px 75px 100px;}

* The auto value **:** You can set the margin property to auto to horizontally center the element within its container.

Ex : div {margin: auto;}

* The inherit Value : Margin inherited from parent element.

Ex :

div {

margin: 50px;

}

p {

margin: inherit;

}

**Padding :**used to create space around an element's content, inside of any defined borders.

* Individual Sides :

padding-top

padding-right

padding-bottom

padding-left

Ex : div { padding-top: 50px; padding-right: 30px; padding-bottom: 50px; padding-left: 80px; }

* Shorthand Property :

Top – right – bottom – left

Ex : div { padding: 25px 50px 75px 100px; }

**Text** : CSS has a lot of properties for formatting text.

* color : used to set the color of the text.

Ex : body { color: blue; }

* text-align : used to set the horizontal alignment of a text.

Left, right, center, justify

Ex : h1 { text-align: center; }

* Text Decoration (shorthand property) :

text-decoration-line (required)

text-decoration-color (optional)

text-decoration-style (optional)

text-decoration-thickness (optional)

Ex : p{  text-decoration: underline red double 5px; }

* text-transform : used to specify uppercase and lowercase letters in a text.

Ex : p{ text-transform: uppercase; }

* text-shadow : adds shadow to text.

Ex : h1 { text-shadow: 2px 2px red};

**Fonts** :

* font-family : to specify the font of a text.

Ex : .p1 { font-family: "Times New Roman", Times, serif; }

Note :  If the font name is more than one word, it must be in quotation marks.

* Font-style : mostly used to specify italic text.

Ex : p { font-style : italic; }

* font-weight : specifies the weight of a font

Ex : p { font-weight: bold; }

* font-size : sets the size of the text

Ex : h1 { font-size: 40px; }

* Shorthand property :

font-style

font-weight

font-size/line-height

font-family

Ex : p { font: italic bold 12px/30px Georgia, serif; }

**Display** : used to specify how an element is shown on a web page.

|  |  |
| --- | --- |
| **Value** | **Description** |
| inline | Displays an element as an inline element |
| block | Displays an element as a block element |
| none | The element is completely removed |
| initial | Sets this property to its default value |
| inherit | Inherits this property from its parent element |

Ex : li { display: inline; }

**Position :** specifies the type of positioning method used for an element (static, relative, fixed, absolute or sticky).

* static : HTML elements are positioned static by default.

Ex : div { position : static; }

* relative : used to position 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.

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

* 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 { position: fixed; bottom: 0; right: 0; }

* absolute : is positioned relative to the nearest positioned ancestor.

However; if an absolute positioned element has no positioned ancestors, it uses the document body, and moves along with page scrolling.

* 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

Ex : div { position: sticky; top: 0; }

Note**:**You must specify at least one of top, right, bottom or left

**z-index** : specifies the stack order of an element.

Ex : img { position: absolute; left: 0px; top: 0px; z-index: -1; }

Here img will be placed behind text.

Note : z-index only works on positioned elements

**overflow** : controls what happens to content that is too big to fit into an area.

Note**:** The overflow property only works for block elements with a specified height.

* 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;

}

**The !important Rule** : used to add more importance to a property/value than normal.

It will override ALL previous styling rules for that specific property on that element!

The only way to override an !important rule is to include another !important

Ex :

p {

background-color: red !important;

}

**Flexbox** :

* Flex container : container becomes  flexible by setting the display property to flex.

Ex : .container { display: flex };

* flex-direction : to define direction of items.

column : stacks the flex items vertically (top to bottom)

column-reverse :  stacks the flex items vertically (bottom to top)

row : stacks the flex items horizontally (from left to right) (default)

row-reverse : stacks the flex items horizontally (right to left)

Ex : .container { display: flex; flex-direction: column-reverse; }

* flex-wrap : specifies whether the flex items should wrap or not.

wrap : for wrap if necessary

nowrap : for no wrap (default)

wrap-reverse : for wrap if necessary but in reverse order

Ex : .container { display: flex; flex-wrap: wrap; }

* justify-content : used to align the flex items

center : aligns items at center of container

flex-start : aligns items at beginning of container (default)

flex-end : aligns items at the end of container

space-around :  items with space before, between, and after the lines

space-between : items with space between the lines

space-evenly : items will have equal space around them

Ex : .container { display: flex; justify-content: space-evenly; }

* align-items : used to align the flex items

center : middle of container

flex-start : top of container

flex-end : bottom of container

stretch : stretches the flex items to fill the container (default)

baseline : aligns the flex items such as their baselines aligns

Ex : .container { display: flex; align-items : center; }