CSS – basics

**External CSS**

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

Each HTML page must include a reference to the external style sheet file inside the <link> element, inside the head section.

Ex :

<!DOCTYPE html>  
<html>  
<head>  
<link rel="stylesheet" href="mystyle.css">  
</head>  
<body>  
  
<h1>This is a heading</h1>  
<p>This is a paragraph.</p>  
  
</body>  
</html>

**Internal CSS**

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

The internal style is defined inside the <style> element, inside the head section.

Ex:

<!DOCTYPE html>  
<html>  
<head>  
<style>  
body {  
  background-color: linen;  
}  
  
h1 {  
  color: maroon;  
  margin-left: 40px;  
}  
</style>  
</head>  
<body>  
  
<h1>This is a heading</h1>  
<p>This is a paragraph.</p>  
  
</body>  
</html>

**Inline CSS**

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

To use inline styles, add the style attribute to the relevant element. The style attribute can contain any CSS property.

Ex:

<!DOCTYPE html>  
<html>  
<body>  
  
<h1 style="color:blue;text-align:center;">This is a heading</h1>  
<p style="color:red;">This is a paragraph.</p>  
  
</body>  
</html>

What style will be used when there is more than one style specified for an HTML element?

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 (inside an HTML element)
2. External and internal style sheets (in the head section)
3. Browser default

So, an inline style has the highest priority, and will override external and internal styles and browser defaults

Conclusion: The last one defined is keept

Colors are specified using predefined color names, or RGB, HEX, HSL, RGBA, HSLA values.

The background-color property specifies the background color of an element. The opacity property specifies the opacity/transparency of an element. It can take a value from 0.0 - 1.0. The lower value, the more transparent:

The background-image property specifies an image to use as the background of an element.

CSS Border Style

The border-style property specifies what kind of border to display.

The following values are allowed:

* dotted - Defines a dotted border
* dashed - Defines a dashed border
* solid - Defines a solid border
* double - Defines a double border
* groove - Defines a 3D grooved border. The effect depends on the border-color value
* ridge - Defines a 3D ridged border. The effect depends on the border-color value
* inset - Defines a 3D inset border. The effect depends on the border-color value
* outset - Defines a 3D outset border. The effect depends on the border-color value
* none - Defines no border
* hidden - Defines a hidden border

If the border-style property has four values:

* **border-style: dotted solid double dashed;**
  + top border is dotted
  + right border is solid
  + bottom border is double
  + left border is dashed

If the border-style property has three values:

* **border-style: dotted solid double;**
  + top border is dotted
  + right and left borders are solid
  + bottom border is double

If the border-style property has two values:

* **border-style: dotted solid;**
  + top and bottom borders are dotted
  + right and left borders are solid

If the border-style property has one value:

* **border-style: dotted;**
  + all four borders are dotted

The border-radius property is used to add rounded borders to an element.

Ex:

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

The CSS box model is essentially a box that wraps around every HTML element. It consists of: content, padding, borders and margins. The image below illustrates the box model:

A screenshot of a computer

Description automatically generated

Explanation of the different parts:

* **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 Style**

The outline-style property specifies the style of the outline, and can have one of the following values:

* dotted - Defines a dotted outline
* dashed - Defines a dashed outline
* solid - Defines a solid outline
* double - Defines a double outline
* groove - Defines a 3D grooved outline
* ridge - Defines a 3D ridged outline
* inset - Defines a 3D inset outline
* outset - Defines a 3D outset outline
* none - Defines no outline
* hidden - Defines a hidden outline

**CSS Outline Width**

The outline-width property specifies the width of the outline, and can have one of the following values:

* thin (typically 1px)
* medium (typically 3px)
* thick (typically 5px)
* A specific size (in px, pt, cm, em, etc)

**Text Alignment**

* The text-align property is used to set the horizontal alignment of a text.
* A text can be left or right aligned, centered, or justified.
* When the text-align property is set to "justify", each line is stretched so that every line has equal width, and the left and right margins are straight

**Text Align Last**

* The text-align-last property specifies how to align the last line of a text.

**Text Direction**

* The direction and unicode-bidi properties can be used to change the text direction of an element:

**Vertical Alignment**

The vertical-align property sets the vertical alignment of an element.

Add a Decoration Line to Text

The text-decoration-line property is used to add a decoration line to text.

**Tip:** You can combine more than one value, like overline and underline to display lines both over and under a text.

Ex: text-decoration-line: overline, line-through, underline

**Specify a Color for the Decoration Line**

The text-decoration-color property is used to set the color of the decoration line.

All links in HTML are underlined by default. Sometimes you see that links are styled with no underline. The text-decoration: none; is used to remove the underline from links, like this:

**Text Transformation**

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

Ex: text-transform: uppercase, lowercase, capitalize

**Text Indentation**

The text-indent property is used to specify the indentation of the first line of a text:

Ex: text-indent: 50px;

**Letter Spacing**

The letter-spacing property is used to specify the space between the characters in a text.

Ex:

h1 {  
  letter-spacing: 5px;  
}

**Line Height**

The line-height property is used to specify the space between lines:

p.small {  
  line-height: 0.8;  
}

**Word Spacing**

The word-spacing property is used to specify the space between the words in a text.

Ex:

p.one {  
  word-spacing: 10px;  
}

**Text Shadow**

The text-shadow property adds shadow to text.

In its simplest use, you only specify the horizontal shadow (2px) and the vertical shadow (2px):

Ex:

h1 {  
  text-shadow: 2px 2px;  
}

**Generic Font Families**

In CSS there are five generic font families:

1. **Serif** fonts have a small stroke at the edges of each letter. They create a sense of formality and elegance.
2. **Sans-serif** fonts have clean lines (no small strokes attached). They create a modern and minimalistic look.
3. **Monospace** fonts - here all the letters have the same fixed width. They create a mechanical look.
4. **Cursive** fonts imitate human handwriting.
5. **Fantasy** fonts are decorative/playful fonts.

A screenshot of a computer

Description automatically generated

**The CSS font-family Property**

In CSS, we use the font-family property to specify the font of a text.

**Note**: If the font name is more than one word, it must be in quotation marks, like: "Times New Roman".

**Tip:** The font-family property should hold several font names as a "fallback" system, to ensure maximum compatibility between browsers/operating systems. Start with the font you want, and end with a generic family (to let the browser pick a similar font in the generic family, if no other fonts are available). The font names should be separated with comma.

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

**Best Web Safe Fonts for HTML and CSS**

The following list are the best web safe fonts for HTML and CSS:

* Arial (sans-serif)
* Verdana (sans-serif)
* Tahoma (sans-serif)
* Trebuchet MS (sans-serif)
* Times New Roman (serif)
* Georgia (serif)
* Garamond (serif)
* Courier New (monospace)
* Brush Script MT (cursive)

**Font Style**

The font-style property is mostly used to specify italic text.

This property has three values:

* normal - The text is shown normally
* italic - The text is shown in italics
* oblique - The text is "leaning" (oblique is very similar to italic, but less supported)

Being able to manage the text size is important in web design. However, you should not use font size adjustments to make paragraphs look like headings, or headings look like paragraphs.

Always use the proper HTML tags, like <h1> - <h6> for headings and <p> for paragraphs.

Responsive Font Size

The text size can be set with a vw unit, which means the "viewport width".

**Font Pairing Rules**

Here are some basic rules to create great font pairings:

1. Complement

It is always safe to find font pairings that complement one another.

A great font combination should harmonize, without being too similar or too different.

2. Use Font Superfamilies

A font superfamily is a set of fonts designed to work well together. So, using different fonts within the same superfamily is safe.

For example, the Lucida superfamily contains the following fonts: Lucida Sans, Lucida Serif, Lucida Typewriter Sans, Lucida Typewriter Serif and Lucida Math.

3. Contrast is King

Two fonts that are too similar will often conflict. However, contrasts, done the right way, brings out the best in each font.

Example: Combining serif with sans serif is a well known combination.

A strong superfamily includes both serif and sans serif variations of the same font (e.g. Lucida and Lucida Sans).

4. Choose Only One Boss

One font should be the boss. This establishes a hierarchy for the fonts on your page. This can be achieved by varying the size, weight and color.

Ex: Georgia and Verdana ; Helvetica and Garamond ; Merriweather and Open Sans; Ubuntu and Lora; Abril Fatface and Poppins; Cinzel and Fauna One; Fjalla One and Libre Baskerville; Space Mono and Muli; Spectral and Rubik; Oswald and Noto Sans

## **The CSS Font Property**

To shorten the code, it is also possible to specify all the individual font properties in one property.

The font property is a shorthand property for:

* font-style
* font-variant
* font-weight
* font-size/line-height
* font-family

**Note:** The font-size and font-family values are required. If one of the other values is missing, their default value are used.

## **How To Add Icons**

The simplest way to add an icon to your HTML page, is with an icon library, such as Font Awesome.

Add the name of the specified icon class to any inline HTML element (like <i> or <span>).

All the icons in the icon libraries below, are scalable vectors that can be customized with CSS (size, color, shadow, etc.)

Ex:

<!DOCTYPE html>  
<html>  
<head>  
<script src="https://kit.fontawesome.com/a076d05399.js" crossorigin="anonymous"></script>  
</head>  
<body>  
  
<i class="fas fa-cloud"></i>  
<i class="fas fa-heart"></i>  
<i class="fas fa-car"></i>  
<i class="fas fa-file"></i>  
<i class="fas fa-bars"></i>  
  
</body>  
</html>

## **Bootstrap Icons**

To use the Bootstrap glyphicons, add the following line inside the <head> section of your HTML page:

<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css">

Ex:  
<!DOCTYPE html>  
<html>  
<head>  
<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css">  
</head>  
<body>  
  
<i class="glyphicon glyphicon-cloud"></i>  
<i class="glyphicon glyphicon-remove"></i>  
<i class="glyphicon glyphicon-user"></i>  
<i class="glyphicon glyphicon-envelope"></i>  
<i class="glyphicon glyphicon-thumbs-up"></i>  
  
</body>  
</html>

## **Google Icons**

To use the Google icons, add the following line inside the <head> section of your HTML page:

<link rel="stylesheet" href="https://fonts.googleapis.com/icon?family=Material+Icons">

Ex:

<!DOCTYPE html>  
<html>  
<head>  
<link rel="stylesheet" href="https://fonts.googleapis.com/icon?family=Material+Icons">  
</head>  
<body>  
  
<i class="material-icons">cloud</i>  
<i class="material-icons">favorite</i>  
<i class="material-icons">attachment</i>  
<i class="material-icons">computer</i>  
<i class="material-icons">traffic</i>  
  
</body>  
</html>

## **Styling Links**

Links can be styled with any CSS property (e.g. color, font-family, background, etc.).

In addition, links can be styled differently depending on what **state** they are in.

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

When setting the style for several link states, there are some order rules:

* a:hover MUST come after a:link and a:visited
* a:active MUST come after a:hover

## **Link Buttons**

This example demonstrates a more advanced example where we combine several CSS properties to display links as boxes/buttons:

Ex:

a:link, a:visited {  
  background-color: #f44336;  
  color: white;  
  padding: 14px 25px;  
  text-align: center;  
  text-decoration: none;  
  display: inline-block;  
}  
  
a:hover, a:active {  
  background-color: red;  
}

## **Different List Item Markers**

The list-style-type property specifies the type of list item marker.

The following example shows some of the available list item markers:

Ex : list-style-type: circle; square; upper-roman; lower-alpha;

When using the shorthand property, the order of the property values are:

* list-style-type (if a list-style-image is specified, the value of this property will be displayed if the image for some reason cannot be displayed)
* list-style-position (specifies whether the list-item markers should appear inside or outside the content flow)
* list-style-image (specifies an image as the list item marker)

## **Table Borders**

To specify table borders in CSS, use the border property.

## **Full-Width Table**

The table above might seem small in some cases. If you need a table that should span the entire screen (full-width), add width: 100% to the <table> element:

## **Collapse Table Borders**

The border-collapse property sets whether the table borders should be collapsed into a single border:

## **Table Width and Height**

The width and height of a table are defined by the width and height properties.

The example below sets the width of the table to 100%, and the height of the <th> elements to 70px

## **Horizontal Alignment**

The text-align property sets the horizontal alignment (like left, right, or center) of the content in <th> or <td>.

By default, the content of <th> elements are center-aligned and the content of <td> elements are left-aligned.

## **Vertical Alignment**

The vertical-align property sets the vertical alignment (like top, bottom, or middle) of the content in <th> or <td>.

By default, the vertical alignment of the content in a table is middle (for both <th> and <td> elements)

## **Table Padding**

To control the space between the border and the content in a table, use the padding property on <td> and <th> elements

## **Hoverable Table**

Use the :hover selector on <tr> to highlight table rows on mouse over:

Ex: tr:hover {background-color: coral;}

## **Responsive Table**

A responsive table will display a horizontal scroll bar if the screen is too small to display the full content:

Ex:

<div style="overflow-x:auto;">  
  
<table>  
... table content ...  
</table>  
  
</div>

## **The display Property Values**

The display property has many values:

|  |  |
| --- | --- |
| **Value** | **Description** |
| inline | Displays an element as an inline element |
| block | Displays an element as a block element |
| contents | Makes the container disappear, making the child elements children of the element the next level up in the DOM |
| flex | Displays an element as a block-level flex container |
| grid | Displays an element as a block-level grid container |
| inline-block | Displays an element as an inline-level block container. The element itself is formatted as an inline element, but you can apply height and width values |
| inline-flex | Displays an element as an inline-level flex container |
| inline-grid | Displays an element as an inline-level grid container |
| inline-table | The element is displayed as an inline-level table |
| list-item | Let the element behave like a <li> element |
| run-in | Displays an element as either block or inline, depending on context |
| table | Let the element behave like a <table> element |
| table-caption | Let the element behave like a <caption> element |
| table-column-group | Let the element behave like a <colgroup> element |
| table-header-group | Let the element behave like a <thead> element |
| table-footer-group | Let the element behave like a <tfoot> element |
| table-row-group | Let the element behave like a <tbody> element |
| table-cell | Let the element behave like a <td> element |
| table-column | Let the element behave like a <col> element |
| table-row | Let the element behave like a <tr> element |
| none | The element is completely removed |
| initial | Sets this property to its default value |
| inherit | Inherits this property from its parent element |

## **Hide an Element - display:none or visibility:hidden**

Hiding an element can be done by setting the display property to none. The element will be hidden, and the page will be displayed as if the element is not there; visibility:hidden; also hides an element

## **Using width, max-width and margin: auto;**

As mentioned in the previous chapter; a block-level element always takes up the full width available (stretches out to the left and right as far as it can).

Setting the width of a block-level element will prevent it from stretching out to the edges of its container. Then, you can set the margins to auto, to horizontally center the element within its container. The element will take up the specified width, and the remaining space will be split equally between the two margins:

This <div> element has a width of 500px, and margin set to auto.

**Note:** The problem with the <div> above occurs when the browser window is smaller than the width of the element. The browser then adds a horizontal scrollbar to the page.

Using max-width instead, in this situation, will improve the browser's handling of small windows. This is important when making a site usable on small devices:

This <div> element has a max-width of 500px, and margin set to auto.

**Tip:** Resize the browser window to less than 500px wide, to see the difference between the two divs!

## **The position Property**

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

There are five different position values:

* static
* relative
* fixed
* absolute
* sticky

Elements are then positioned using the top, bottom, left, and right properties. However, these properties will not work unless the position property is set first. They also work differently depending on the position value

## **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).

A close-up of a white background

Description automatically generated

Ex:

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

## **CSS Overflow**

The overflow property specifies whether to clip the content or to add scrollbars when the content of an element is too big to fit in the specified 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

## **overflow-x and overflow-y**

The overflow-x and overflow-y properties specifies whether to change the overflow of content just horizontally or vertically (or both):

overflow-x specifies what to do with the left/right edges of the content.  
overflow-y specifies what to do with the top/bottom edges of the content.

Ex:

div {  
  overflow-x: hidden; /\* Hide horizontal scrollbar \*/  
  overflow-y: scroll; /\* Add vertical scrollbar \*/  
}

## **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:

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

In its simplest use, the float property can be used to wrap text around images.

## **The clear Property**

When we use the float property, and we want the next element below (not on right or left), we will have to use the clear property.

The clear property specifies what should happen with the element that is next to a floating element.

The clear property can have one of the following values:

* none - The element is not pushed below left or right floated elements. This is default
* left - The element is pushed below left floated elements
* right - The element is pushed below right floated elements
* both - The element is pushed below both left and right floated elements
* inherit - The element inherits the clear value from its parent

When clearing floats, you should match the clear to the float: If an element is floated to the left, then you should clear to the left. Your floated element will continue to float, but the cleared element will appear below it on the web page

A close-up of a pineapple

Description automatically generated

## **Navigation Menu**

You can also use float with a list of hyperlinks to create a horizontal menu: