Cascading Style Sheets (CSS)

Introduction to CSS

What is CSS?

- CSS stands for Cascading Style Sheets.
- CSS is the language we use to style an HTML document.
- CSS describes how HTML elements should be displayed.

Developed by Håkon Wium Lie and Bert Bos in 1994

Introduction to CSS: Types of CSS

CSS can be added to HTML documents in 3 ways:

(1) Inline

(2) Internal

(3) External (Most Popular)

Introduction to CSS: Syntax

- CSS Syntax

```
selector {
        property1: value1;
        property2: value2;
    }
```

A Selector can be a HTML Tag, id or class

Introduction to CSS: selectors

1. Simple selectors (select elements based on name (p), id (#), class(.))
2. Combinator selectors (descendant (space), child (>))
3. Pseudo-class selectors (Anchor Pseudo-classes (a:hover), Div Pseudo-classes (div:hover))
4. Pseudo-elements selectors (p::first-line, p::first-letter)
5. Attribute selectors (input[type="text"], input[type="button"])

CSS Colors

CSS Colors

In CSS, colors are specified by using a predefined color name, or with a RGB, HEX, HSL, RGBA, HSLA value.

CSS Color Names

In CSS, a color can be specified by using a predefined color name:

Tomato	Orange	DodgerBlue	MediumSeaGreen
Gray	SlateBlue	Violet	LightGray

CSS Text Styling: Font properties and text formatting

CSS Text

- Text Color (`color:, background-color`)
- Text Alignment (`text-align:center;`)
- Text Decoration (`text-decoration: underline red double;`)
- Text Transformation (`text-transform: uppercase;`)
- Text Spacing (`text-indent: 50px;, letter-spacing: 5px;,  line-height: 1.8;`

CSS Text Color

You can set the color of text:

Hello World

Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat.

Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat.

CSS Border Color

You can set the color of borders:

Hello World

Hello World

Hello World

CSS Color Values

In CSS, colors can also be specified using RGB values, HEX values, HSL values, RGBA values, and HSLA values:

Same as color name "Tomato":

rgb(255, 99, 71)

#ff6347

hsl(9, 100%, 64%)

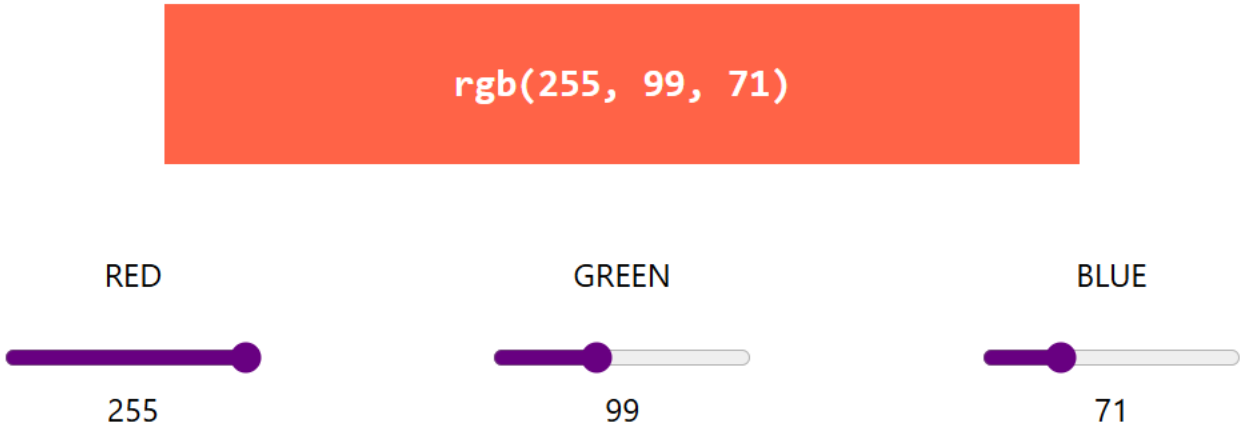
Same as color name "Tomato", but 50% transparent:

rgba(255, 99, 71, 0.5)

hsla(9, 100%, 64%, 0.5)

CSS RGB Colors

- Each parameter (red, green, and blue) defines the intensity of the color between 0 and 255.
- For example, rgb(255, 0, 0) is displayed as red, because red is set to its highest value (255) and the others are set to 0.
- To display black, set all color parameters to 0, like this: rgb(0, 0, 0).
- To display white, set all color parameters to 255, like this: rgb(255, 255, 255).
- Experiment by mixing the RGB values below:



RGBA Value

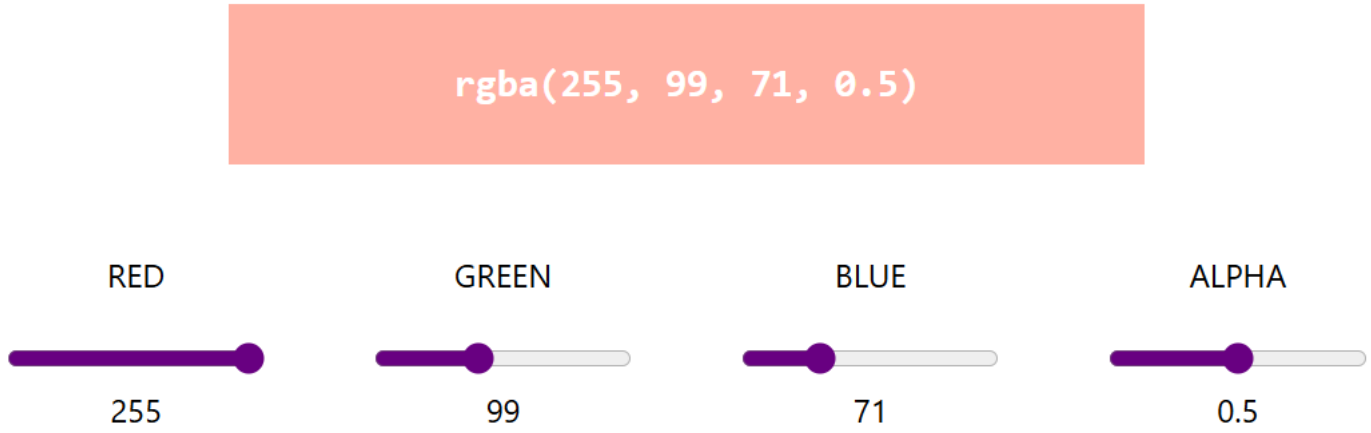
RGBA color values are an extension of RGB color values with an alpha channel - which specifies the opacity for a color.

An RGBA color value is specified with:

rgba(*red, green, blue, alpha*)

The alpha parameter is a number between 0.0 (fully transparent) and 1.0 (not transparent at all):

Experiment by mixing the RGBA values below:



CSS HEX Colors

HEX Value

A hexadecimal color is specified with: #RRGGBB.

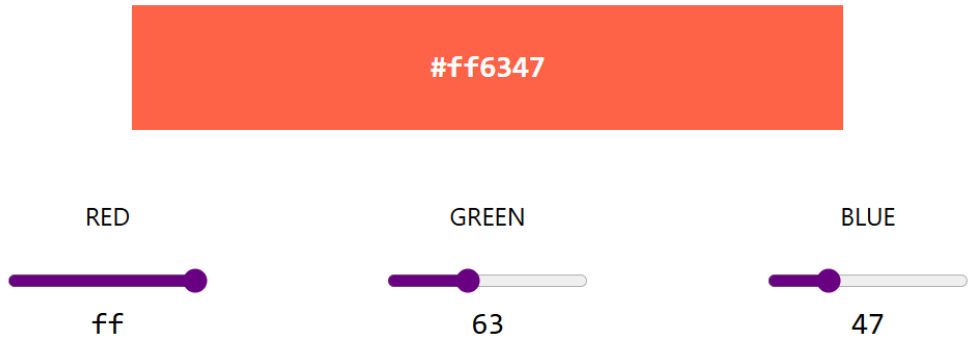
In CSS, a color can be specified using a hexadecimal value in the form:

#rrggbb

Where rr (red), gg (green) and bb (blue) are hexadecimal values between 00 and ff (same as decimal 0-255).

For example, #ff0000 is displayed as red, because red is set to its highest value (ff) and the others are set to the lowest value (00).

- To display black, set all values to 00, like this: #000000.
- To display white, set all values to ff, like this: #ffffff.



3 Digit HEX Value

Sometimes you will see a 3-digit hex code in the CSS source.

The 3-digit hex code is a shorthand for some 6-digit hex codes.

The 3-digit hex code has the following form:

#rgb

Where r, g, and b represent the red, green, and blue components with values between 0 and f.

The 3-digit hex code can only be used when both the values (RR, GG, and BB) are the same for each component. So, if we have #ff00cc, it can be written like this: #f0c.

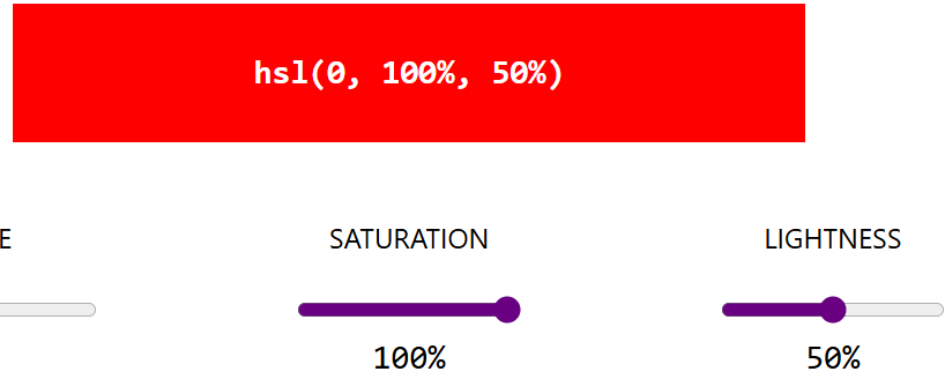
CSS HSL Colors

HSL Value

HSL stands for hue, saturation, and lightness.

In CSS, a color can be specified using hue, saturation, and lightness (HSL) in the form:

hsl(*hue*, *saturation*, *lightness*)



Hue is a degree on the color wheel (from 0 to 360):

- 0 (or 360) is red
- 120 is green
- 240 is blue

Saturation is a percentage value: 0% means a shade of gray, and 100% is the full color.

Lightness is also a percentage; 0% is black, 50% is neither light or dark, 100% is white.

Experiment by mixing the HSL values below:

CSS HSL Colors

Saturation

- Saturation can be described as the intensity of a color.
- 100% is pure color, no shades of gray.
- 50% is 50% gray, but you can still see the color.
- 0% is completely gray; you can no longer see the color.

Lightness

- The lightness of a color can be described as how much light you want to give the color, where 0% means no light (black), 50% means 50% light (neither dark nor light) and 100% means full lightness (white).

HSLA Value

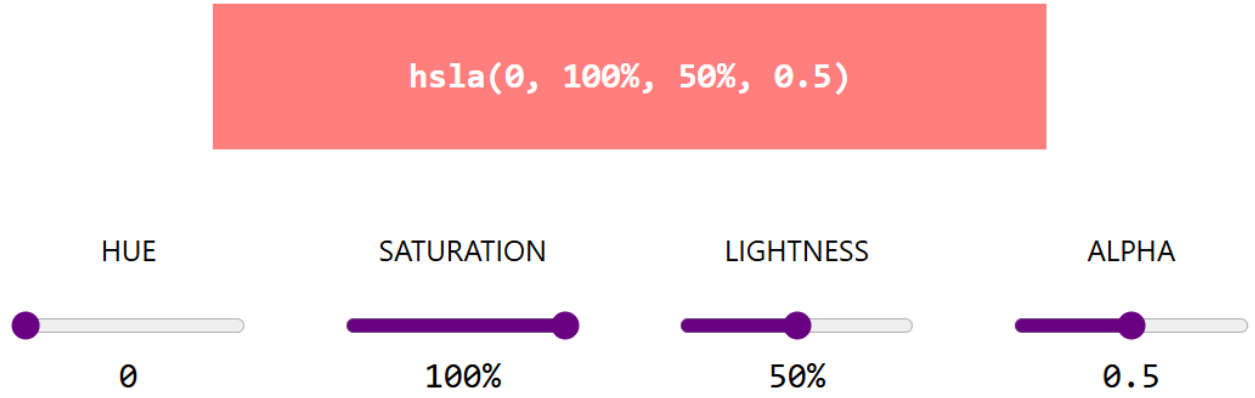
HSLA color values are an extension of HSL color values with an alpha channel - which specifies the opacity for a color.

An HSLA color value is specified with:

hsla(*hue, saturation, lightness, alpha*)

The alpha parameter is a number between 0.0 (fully transparent) and 1.0 (not transparent at all):

Experiment by mixing the HSLA values below:



CSS Fonts

Font Selection is Important

- Choosing the right font has a huge impact on how the readers experience a website.
- The right font can create a strong identity for your brand.
- Choosing a font that is easy to read is important. It is also important to choose a good color and size for your font.

The CSS font-family Property

The CSS `font-family` property specifies the font for an element.

Tip: The `font-family` property should hold several font names as a "fallback" system.

If the browser does not support the first font, it tries the next font. The font names should be separated with a comma.

CSS 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.

All the different font names belong to one of the generic font families.

Difference Between Serif and Sans-serif Fonts

Generic Font Family	Examples of Font Names
Serif	Times New Roman Georgia Garamond
Sans-serif	Arial Verdana Helvetica
Monospace	Courier New Lucida Console Monaco
Cursive	Brush Script MT Lucida Handwriting
Fantasy	Copperplate Papyrus

CSS Web Safe Fonts

What are Web Safe Fonts?

Web safe fonts are fonts that are universally installed across all browsers and devices.

Fallback Fonts

However, there are no 100% completely web safe fonts. There is always a chance that a font is not found or is not installed properly.

Therefore, it is very important to always use fallback fonts.

This means that you should add a list of similar "backup fonts" in the `font-family` property. If the first font does not work, the browser will try the next one, and the next one, and so on. Always end the list with a generic font family name.

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)

CSS Font Style

CSS Font Style

The CSS `font-style` property specifies the font style for a text..

This property can have one of the following 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)

CSS Font Weight

The CSS `font-weight` property specifies how thick or thin characters in text should be displayed.

This property can have one of the following values:

- `normal` - This is default. Defines normal characters
- `bold` - Defines thick characters
- `bolder` - Defines thicker characters
- `lighter` - Defines lighter characters
- `100-900` - Defines from thin to thick characters. 400 is the same as normal, and 700 is the same as bold

CSS Font Variant

The CSS `font-variant` property specifies whether or not a text should be displayed in a small-caps font.

In a small-caps font, all lowercase letters are converted to uppercase letters. However, the converted uppercase letters appears in a smaller font size than the original uppercase letters in the text.

CSS Font Size

The CSS `font-size` property is used to specify the size of the text/font.

Being able to manage the text size is very 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.

The `font-size` can be set to an absolute size or to a relative size.

CSS Font Size

Absolute sizes:

- `px`: Pixels offers fixed and precise control over the font size.
- `xx-small`, `x-small`, `small`, `medium`, `large`, `x-large`, `xx-large`. These keywords has a predefined set of sizes in browsers.

Relative sizes:

- `em`: This unit is relative to the font size of the parent element.
- `rem`: This unit is relative to the font size of the root HTML element.
- `%`: This unit is relative to the font size of the parent element
- `smaller` and `larger`: These units adjust the font size relative to the parent element.

Set Font Size With Em

The `em` unit is relative to the font size of the parent element. So, if the parent element has a font size of 16px, then 2.5em would result in 40px.

In the following example, the text size in `em` is the same as the previous example in pixels. However, the `em` unit allows the user to adjust the text size in the browser settings.

Set Font Size With Rem

The `rem` unit is relative to the font size of the root HTML element (<html>).

Unlike `em`, which is relative to the font-size of its parent element, `rem` always refers to the font-size of the <html> element, regardless of its position in the document tree. This makes `rem` very useful for creating scalable and responsive designs. By changing the font-size of the <html> element, all elements sized with `rem` units will scale proportionally throughout the entire page.

The default font-size of the <html> element in most browsers, is 16px. So, by default, 1rem equals 16px unless explicitly overridden in the CSS.

The vw Unit

The font size can also be set with the `vw` unit, which means the "viewport width".

The `vw` unit is a relative unit that represents a percentage of the width of the viewport.

1vw = 1% of the current width of the browser's viewport. So, if the viewport is 500px wide, 1vw is 5px.

This way the text size will follow the size of the browser window:

CSS Google Fonts

Google Fonts

If you do not want to use any of the standard fonts in HTML, you can use Google Fonts.

Google Fonts are free to use, and have more than 1000 fonts to choose from.

How To Use Google Fonts

Just add a special style sheet link in the <head> section and then refer to the font in the CSS.

Use Multiple Google Fonts

To use multiple Google fonts, just separate the font names with a pipe character (|), like this:

Styling Google Fonts

Of course, you can style Google Fonts as you like, with CSS!

Enabling Font Effects

Google has also enabled different font effects that you can use.

First add `effect=`*`effectname`* to the Google API, then add a special class name to the element that is going to use the special effect. The class name always starts with `font-effect-` and ends with the *`effectname`*.

To request multiple font effects, just separate the effect names with a pipe character (`|`), like this:

CSS Great Font Pairings

Great font pairings are essential to great design!

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.

Font Pairing Rules

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.

CSS Font Property

The CSS Font Shorthand Property

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

The CSS `font` property is a shorthand property for:

- `font-style`
- `font-variant`
- `font-weight`
- `font-stretch`
- `font-size`
- `line-height`
- `font-family`

CSS Font Property

Rules for the CSS Font Shorthand Property

1. The `font-size` and `font-family` values are required. If one of the other values is missing, their default value are used.
2. If defined, `font-style`, `font-variant` and `font-weight` must precede `font-size`.
3. If defined, `line-height` must immediately follow `font-size`, preceded by "/", like this: 15px/30px.
4. `font-family` must be the last value specified.

CSS Text Styling

CSS Fonts

- font-family (`font-family: "Times New Roman“;`)
- font-style (`font-style: italic;`)
- Font Size (`font-size: 100%; font-size: 40px; font: 20px Arial, sans-serif;`)

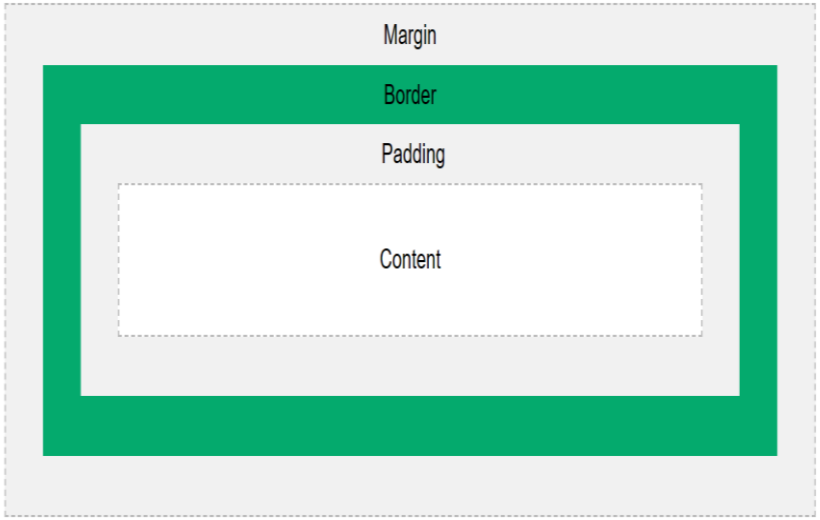
$$1\ \text{px} = 0.026458333333333\ \text{cm}$$

CSS Box Model: Margin, padding, and border

- In CSS, the term "box model" is used when talking about web design and layout.
- The CSS box model is essentially a box that wraps around every HTML element.
- Every box consists of four parts: content, padding, borders and margins.

CSS Box Model: Margin, padding, and border

- **The CSS Box Model**



- **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

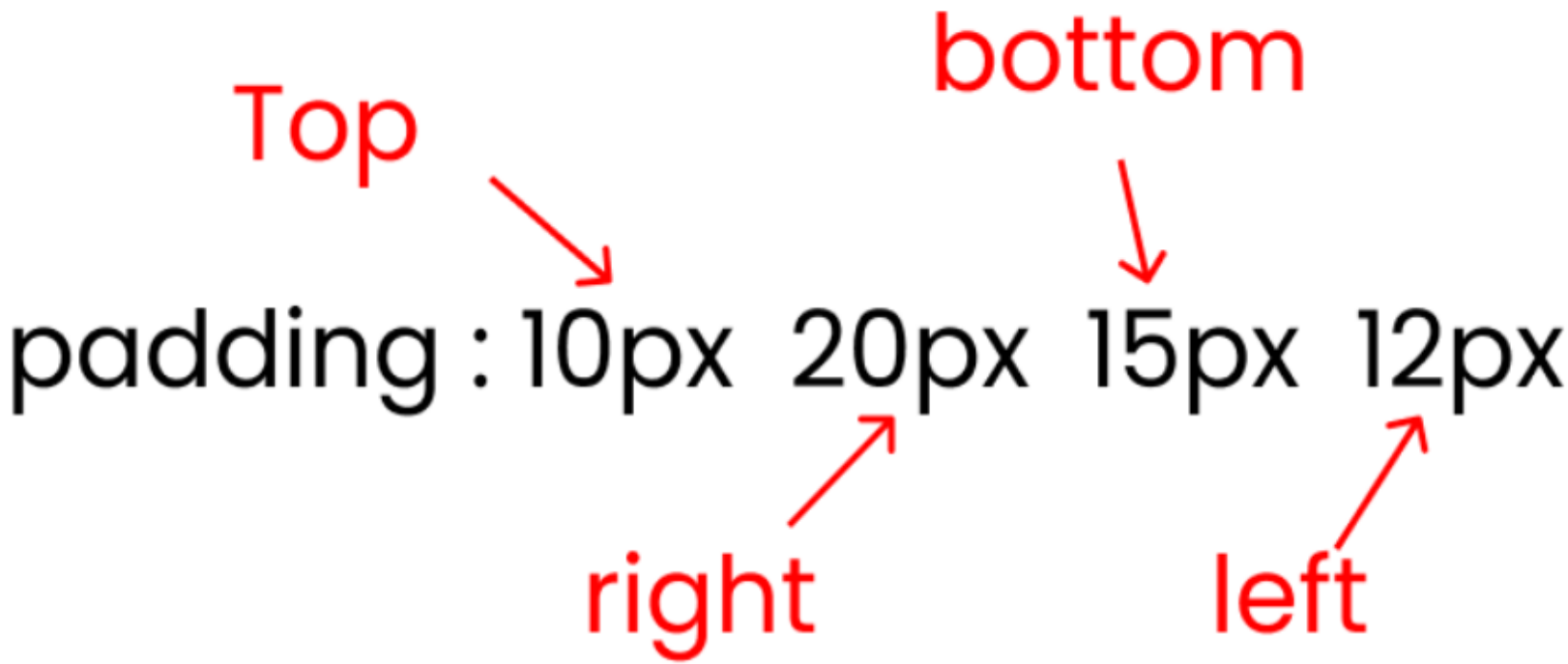
CSS Box Model: Margin, padding, and border

Width and Height of an Element

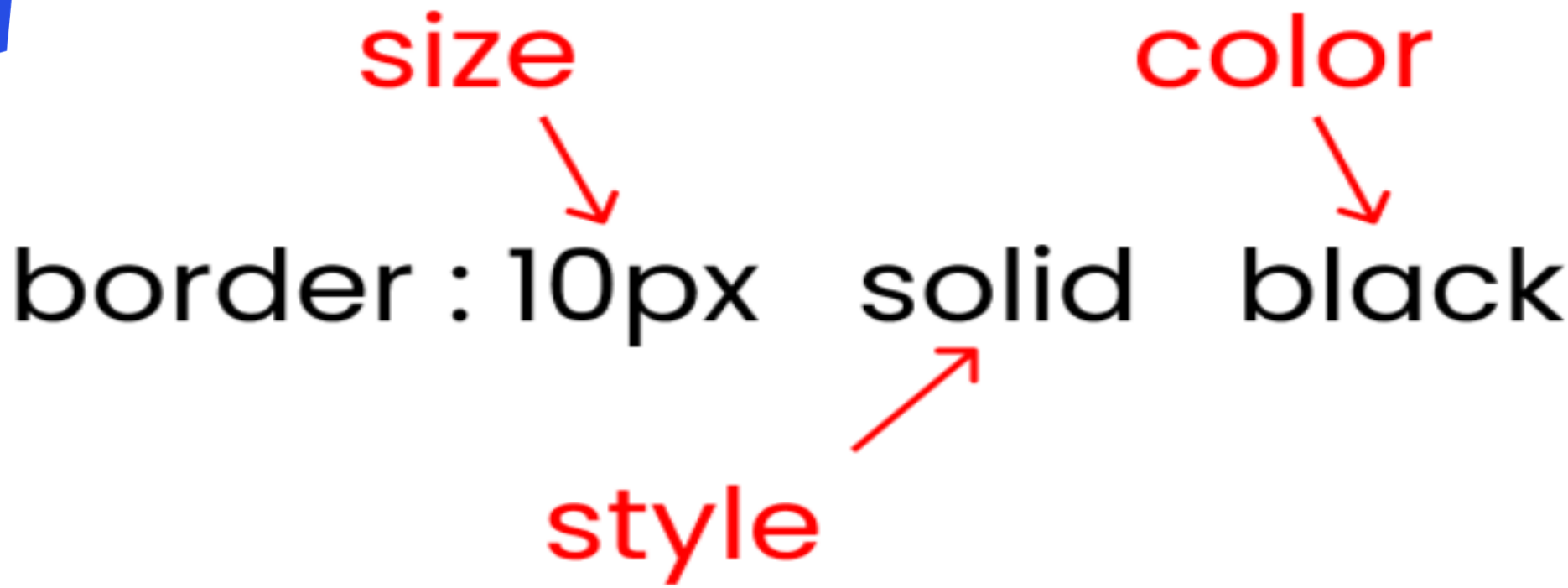
In order to set the width and height of an element correctly in all browsers, you need to know how the box model work

Navigation bar without padding

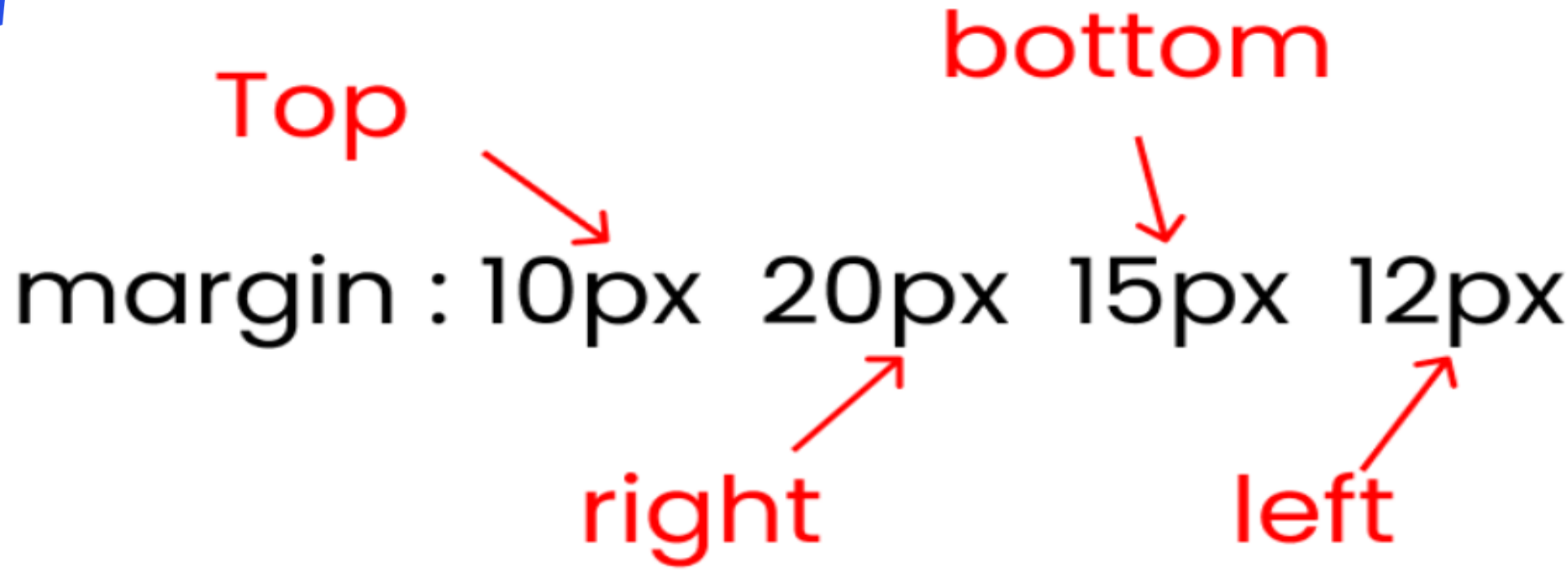
Navigation bar with padding



Shorthand of padding property



Border property syntax



Shorthand of the margin property

Positioning and float

Positioning and float

```
img {
  float: right;
}


.img-container {
  float: left;
  width: 33.33%; /* three containers (use 25% for four)
  padding: 5px; /* if you want space between the images
*/
}
```

Flexbox: Creating flexible layouts

CSS Flexbox

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

```
<div class="flex-container">
  <div>1</div>
  <div>2</div>
  <div>3</div>
</div>
```

Grid: Building grid-based layouts

```css
.grid-container {
  display: grid;
  row-gap: 50px;
}
```

```html
<div class="grid-container">
  <div class="grid-item">1</div>
  <div class="grid-item">2</div>
  <div class="grid-item">3</div>
  <div class="grid-item">4</div>
</div>
```

Responsive Layout: Media queries

Media Query?

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

Responsive Layout: mobile-friendly layouts

```
/* For mobile phones: */
[class*="col-"] {
  width: 100%;
}
```

```
@media only screen and (min-width: 600px) {
  /* For tablets: */
  .col-s-1 {width: 8.33%;}
  .col-s-2 {width: 16.66%;}
  .col-s-3 {width: 25%;}
  .col-s-4 {width: 33.33%;}
  .col-s-5 {width: 41.66%;}
  .col-s-6 {width: 50%;}
  .col-s-7 {width: 58.33%;}
  .col-s-8 {width: 66.66%;}
  .col-s-9 {width: 75%;}
  .col-s-10 {width: 83.33%;}
  .col-s-11 {width: 91.66%;}
  .col-s-12 {width: 100%;}
}
```

```
@media only screen and (min-width: 768px) {
  /* For desktop: */
  .col-1 {width: 8.33%;}
  .col-2 {width: 16.66%;}
  .col-3 {width: 25%;}
  .col-4 {width: 33.33%;}
  .col-5 {width: 41.66%;}
  .col-6 {width: 50%;}
  .col-7 {width: 58.33%;}
  .col-8 {width: 66.66%;}
  .col-9 {width: 75%;}
  .col-10 {width: 83.33%;}
  .col-11 {width: 91.66%;}
  .col-12 {width: 100%;}
}
```

CSS Transitions

CSS Transitions

```
div:hover {
        width: 300px;
        height: 100px;
        }


div {
        width: 100px;
        height: 100px;
        background: red;
        transition: width 2s, height 4s;
     }
```

Thank You