Units

-rem vs em (Relative units)

em
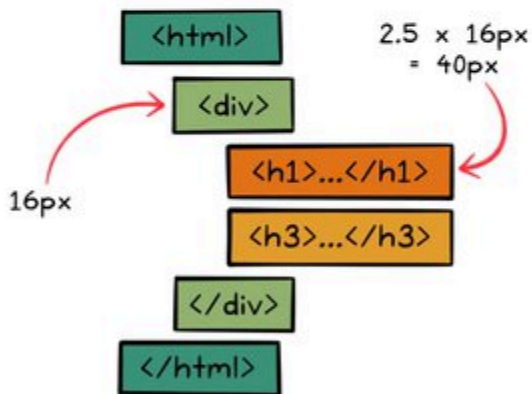
Relative to the font size of the parent element.

`h1 { font-size: 2.5em; }`

```
<html>
  <div>
    <h1>...</h1>
    <h3>...</h3>
  </div>
</html>
```

16px

2.5 x 16px = 40px

rem

Relative to the font size of the root element.

`h1 { font-size: 2.5rem; }`

```
<html>
  <div>
    <h1>...</h1>
    <h3>...</h3>
  </div>
</html>
```

24px

2.5 x 24px = 60px
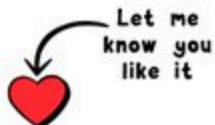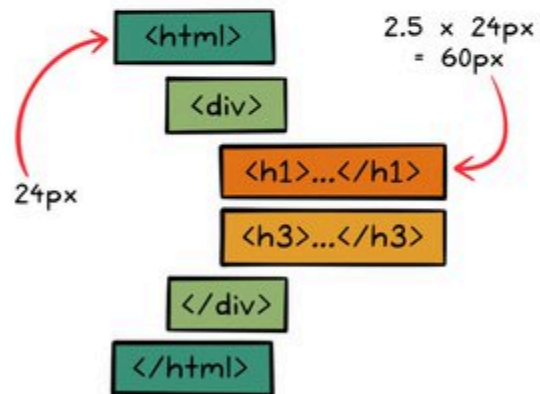
em unit is relative to the CSS font size of its parent element, whereas a rem unit is always relative to the root CSS font size.

1em = 16px (1 * 16)

1rem = 16px

The rem unit in CSS stands for "root em". It is a relative unit of measurement that is relative to the font size of the root element. One rem is equal to the font size of the root element.

em units can be used for much more than just setting font-size however, and they can be used pretty much everywhere units are expected (padding, margin, width, height, max-width,…) When em units are used on other properties than font-size, the value is relative to the element's own font-size.

by using the rem unit, the values of parent elements are ignored, and only the value of the root is taken into consideration.

rem Units vs em Units

Parameter	rem	em
Definition	Relative to the root element (html) font size	Relative to the parent element's font size
Consistency	Consistent, since it always refers to the same base font size	Can cause issues with cascading font size values
Default font-size	16px (default font size of root element)	Based on parent element's font size
Browser support	Widely supported in modern browsers	Universal support
Use case	To size elements relative to the root element's font size for a consistent and scalable design	To size elements relative to the parent element's font size

The font size of an element in relation to its parent may be changed using the em unit. It thus manages to make it simple to sustain the size correlation amongst elements in a responsive design.

One of the major difference between em and rem CSS units is how the browser calculates their px value. It is important to understand this difference in order to determine when each unit should be used.

Applying ems and rems in CSS:

Here are a few other ways in which rem and em can be applied;

When it comes to sizes and spacing, use rem. Media queries should be made using em.

Choose rem units to create elements that function based on user perceptions, and em units to improve elements that rely on their parent elements to deploy. An easily accessible website can be designed with features that can be customized to the requirements of the user by utilizing rem units instead of px units.

.

em vs rem, Which is Better?

There's no better unit really, and it all depends on your personal preferences. Some people like to design everything in rem units for consistency and predictability, while others like to also use em units in places where the influence of nearby parent elements would make sense

-vh vs vw (viewport units)

Differences between vh and vw units in CSS

Aspect	Viewport Height (vh)	Viewport Width (vw)
Definition	Represents 1% of the viewport's height	Represents 1% of the viewport's width
Usage	Used for vertical scaling of elements	Used for horizontal scaling of elements
Common Applications	Setting heights, vertical margins, and paddings	Setting widths, horizontal margins, and paddings
Responsive Design	Ensures elements adjust to changes in viewport height	Ensures elements adjust to changes in viewport width
Examples	`height: 50vh;` sets height to 50% of viewport height	`width: 50vw;` sets width to 50% of viewport width

To size something as tall as the viewport, you can use the vw and vh units.

- vw = 1% of the width of the viewport size.
- vh = 1% of the height of the viewport size.

These units enable responsive design by sizing elements based on the user's viewport.

vh is commonly used for creating responsive designs where elements scale with the height of the viewport. This unit helps maintain consistent vertical spacing relative to the viewport height.

It's used for responsive design, scaling elements dynamically based on the width of the user's viewport.

-pc vs cm (Absolute Units)

Unit	Description
cm	centimeters
mm	millimeters
in	inches (1in = 96px = 2.54cm)
px *	pixels (1px = 1/96th of 1in)
pt	points (1pt = 1/72 of 1in)
pc	picas (1pc = 12 pt)

A centimeter (cm) is an absolute unit of length. 1cm is roughly 37.8 pixels.
A pica (pc) is an absolute unit of length. One pc is equivalent to 16px.

-color hsl(hue, saturation, lightness)

0°   HUE   360°

0%   SATURATION   100%

0%   LIGHTNESS / BRIGHTNESS   100%

HSL is an abbreviation for hue, saturation, and lightness. This color model is built around the RGB color wheel. The color's transparency is represented by an optional alpha component, turning HSL into HSLA. Take a look at the syntax below:

Hue

Hue measures the value of an angle on a color wheel
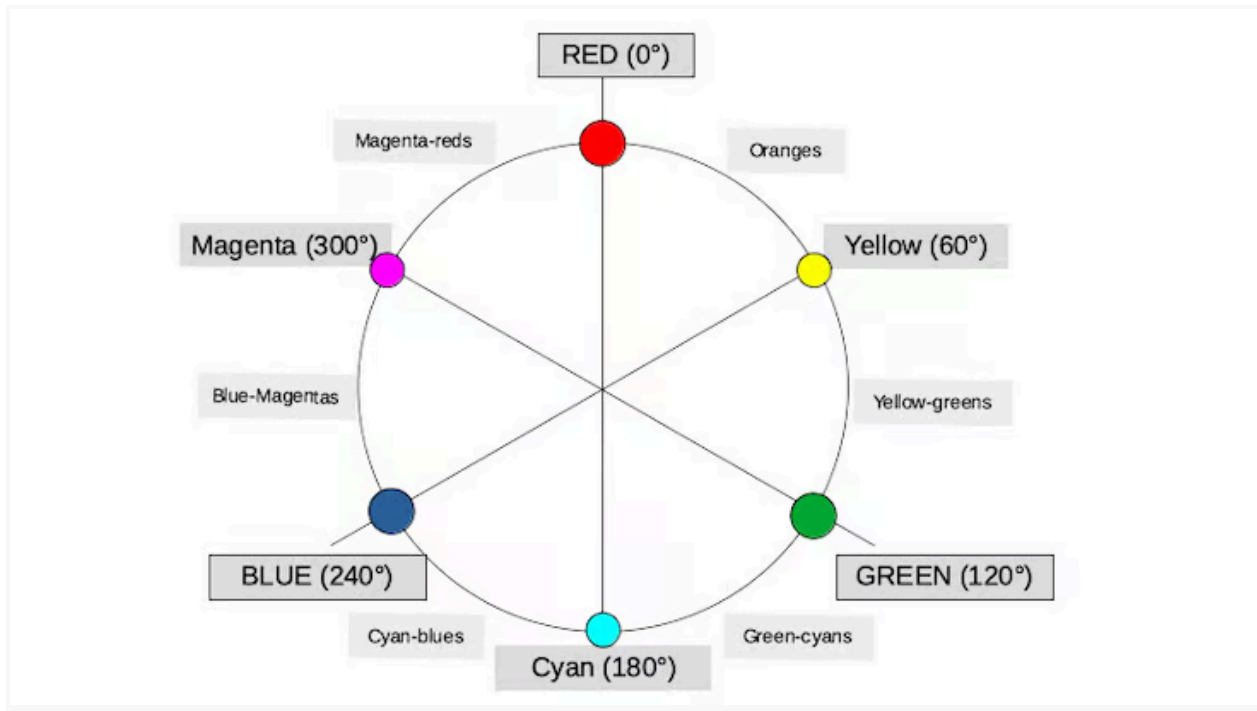
By default, hue is measured in degrees:

- ○ Red: 0 degrees and 360 degrees
- ○ Yellow: 60 degrees
- ○ Green: 120 degrees
- ○ Cyan: 120 degrees
- ○ Blue: 240 degrees
- ○ Magenta: 300 degrees

All other colors fall in between these values in an intuitive way. For example, orange — a secondary color that falls between red and yellow on the visible light spectrum — falls somewhere between 0 and 60 degrees, depending on the particular shade of orange you want.

Saturation

The intensity of a color can be described as saturation, which is measured by percentage value. Take a look at the image below, which demonstrates how different percentages of saturation for the color red may look:

saturation

| 0% | 20% | 40% | 60% | 80% | 100% |
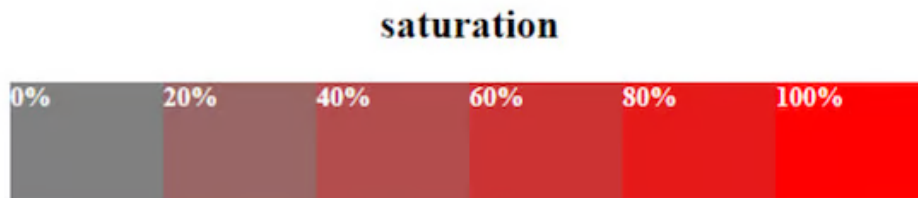
As you can see, 0 percent saturation indicates a shade of gray, while a value of 100 percent indicates a fully saturated color in its most vibrant hue. Selecting a value between 10% and 100% is recommended so that users can see the color.

Lightness

We can describe the lightness of a color as how much light you give to the color. Like saturation, the lightness in HSL is also measured by percentage. See how red can look with different levels of lightness:

**lightness**

| 0% | 20% | 40% | 60% | 80% | 100% |
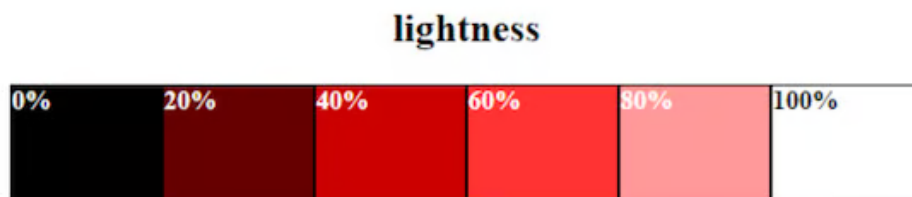
0 percent lightness will appear black. 50 percent is neither light nor black; we can say this is neutral. 100 percent lightness will appear white therefore, It is preferable to select a value between 10 and 90 percent as this will enable us to see the base color.

How to use HSL in CSS

To use HSL in CSS, we must specify the value each component will have.

Keep the syntax for HSL in mind; the first value supplied to HSL is the hue, followed by saturation, and finally, lightness.
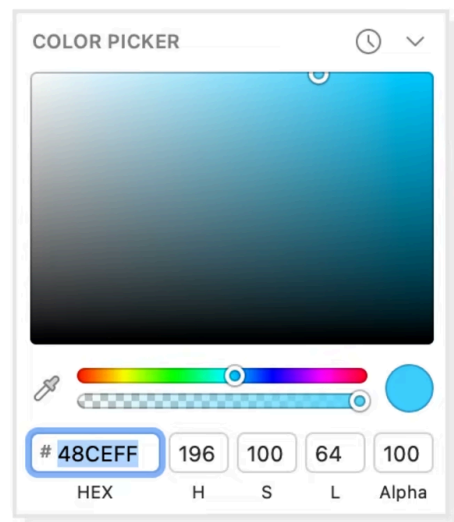
What makes HSL unique?

Readability

One of the most significant advantages of HSL is its readability. You don't have to spend much time learning how to interpret HSL code, unlike HEX code.