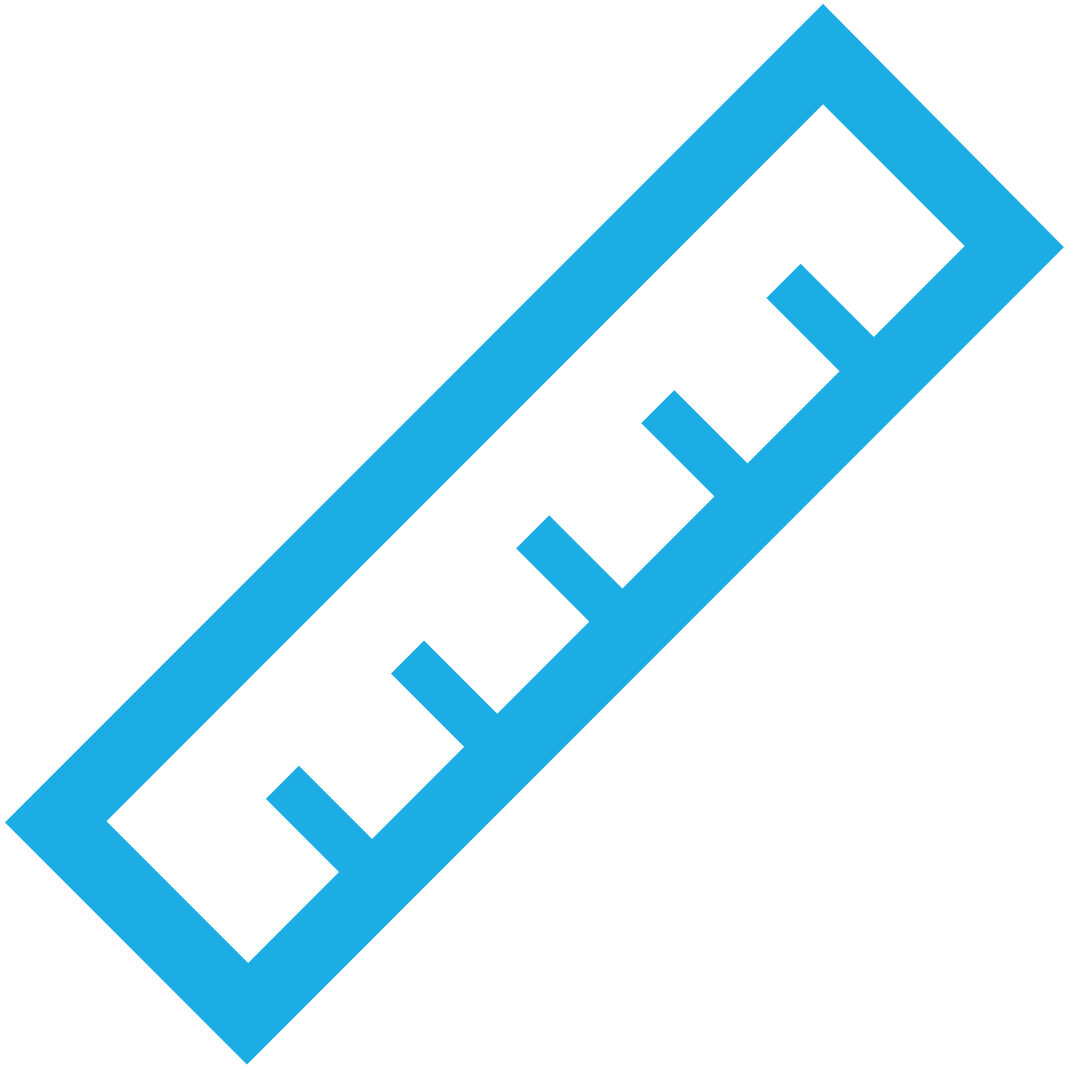
CSS MEASUREMENT



UNIT TYPES

In CSS we have two different types of measurement. They are:

- Relative, where your size is going to be in reference to something else on the screen / device.
- Absolute, where your size is going to be determined by some measurement in actual pixels on the device.

We will see that pixels aren't actually counting the amount of pixels in your screen. This would cause layouts and fonts to become tiny as screen resolution gets higher.

What we will actually see is that "pixels" are more of a standard unit of measurement (like inches, centimeters etc.)

ABSOLUTE UNITS

Back before mobile development was a concern, it was easy to assume everyone viewing your page had a pretty standard screen size.

This allowed developers to use absolute units of measurement and just assume that things would work out on the user's screen.

Today we have too many different screen sizes to really use absolute units for anything but tiny operations.

ABSOLUTE UNITS CONT.

CSS Name	English Name
cm	Centimeters
mm	Millimeters
in	Inches
px	Pixels
pt	Points
pc	Picas

KNOWLEDGE CHECK

Let's get our hands dirty with some measurement units:

1. Create a folder called MeasurementUnits in your Scratch directory
2. Connect this folder to Git/GitHub
3. Create a simple index.html and style.css
4. Add in 6 empty div tags to your index.html
5. Give each div a unique id
6. Write a CSS rule for each unique id
    1. Give each div a different **width** and **height** using the 6 different units of measurement in the slide before. Make sure you give each div a **background color** so you can see it on the page!
7. Add, commit and push your code

PIXELS

It should be noted that the px unit is **not** the actual pixels on your screen! The actual pixels on your screen are called “device pixels” and their density can change based on the resolution of the screen.

This means if you were to use device pixels as a unit of measurement, the look of your page could be totally different on two identical screen sizes if they have a different screen resolution.

To fix this, CSS defines pixels as device independent. The goal is to have 96px ~=~ 1 in. Now no matter what the resolution, 96px will take up about 1 inch of the screen.

RELATIVE UNITS

The size of relative units always use another unit's size as a reference starting point.

Try to keep the distinction clear in your head as you size elements on your page.

Relative units do a better job of adapting to screen size changes in the browser.

RELATIVE UNITS CONTS.

CSS Name	English Name
em	Relative to parent font size
rem	Relative to root font size
vw	Viewport width
vh	Viewport height
%	Percentage relative to parent

EM V REM

The units em and rem are very similar, but have one important distinction in what they are relatively measured against.

The rem unit is relative to the **font-size** of the root (html tag) element. This means that if you give your html tag a new font-size and all other font-size properties are measured in rem, all other font sizes will now change.

The em unit is relative to the **font-size** of its direct parent element. 1em = the font-size of your parent element. So if you are using em and change the parent value for font-size, the child will now also change.

VH
VW
%

Both the vh and the vw units are relative to the viewport of the users screen (the browser window). This means that you can define your sizes and have them adapt to different screen sizes or the user making their browser bigger / smaller.

Both vh and vw are defined from 0 -100 where 100 is the whole screen and 0 is none of the screen.

The % unit is relative to the direct parent element and **NOT** to the actual screen size. This means that if your parent element is defined with a width of 50vw and the child as 100%, the child will still only take up 50vw.

KNOWLEDGE CHECK

Let's add even more measurement units:

1. Add 2 different text base content tags to your HTML
2. Add 3 more empty divs to your HTML and give each one a unique id
3. Write a CSS rule for the new ids
    1. Give each div a different **width** and **height** using **vw, vh and** %. Make sure you give each div a **background color** so you can see it on the page!
4. Write a CSS rule for each text based tag
    1. Change the font-size for each using **rem** for one and **em** for the other
5. Add, commit and push your code

WHAT TO WORRY ABOUT

When learning about measurement units, we need to talk about their most common use case which is basic layout styling.

When you need to specify how much width or height an element should take up, or how much space should be between two different elements- that's when we need to worry about measurement units.

The first thing we need to worry about is breaking screen width. Most of the time you don't want to break out of the screen width causing horizontal scrolling. The only time you might want to do this is with a section on a site that is intentionally scrolling sideways (think Netflix shows inside a category).

The next thing you want to worry about is a child tag breaking out of the parent tag restrictions. For example, an article inside a section having a width greater than the section.

WHEN TO USE WHAT?

With all of these different units of measurement you might be a little confused when deciding which one to use. While there is no 100% correct rule that you can follow all the time, there are some general guidelines you can follow with questions:

Is the thing you are trying to space or size very small?

- You can probably use things like **px** for this. A logo being 16px wide and 16px tall will probably work on most screen sizes and if you use a relative measurement for this it will probably look bad on different screen sizes

Are you trying to make something take up a certain amount of space in the whole window?

- This is where things like **vw** and **vh** really come and shine. Good for giving large containing tags like sections an appropriate size

Are you trying to make a child tag take up a certain amount of a parent tag?

- This is where % is your friend. It is designed for this purpose.

Are you trying to give different font sizes that scale well?

- The units **rem** and **em** are your best bet here. They are designed to be relative to the font-size property.

COMMON ISSUES AND ERRORS

When trying to do some layout basics with the different units of measurement it is common to run into issues where something somewhere breaks the width of the page or their parent tag. Here are some common issues:

1. **Images** are breaking out of my layout – This is almost always caused by either not setting a width or height, or just setting one and letting the other be a default value.
    1. If you need a very specific image size, it is best to set the width and height and then set the **object-fit** property to **cover** in css.
2. Children are breaking out of their parents or containers are breaking out of the page – There is a very good chance here you have simply exceeded the amount of screen real estate you have in the viewport.
    1. This is **very** common when you start mixing and matching different measurement units. Using px for margin, % for width and then em for padding. This is a recipe for disaster and should be avoided.

Knowing the difference between units of measurement will allow you to take more control of your style across different screen sizes and resolution.

Not knowing them will lead to layout bugs that are very tricky to debug!