Units: px, em, rem, %, vh, vw

`font-size`, `line-height`, `margins` and many other CSS properties expect some sort of dimension value. Dimension values support a wide variety of units. But the most common and useful ones are: `px, em, rem, %, vh` and `vw`.

PX

'px' is short for 'pixel', which is a single dot on the screen. So text with `font-size:20px` is 20 pixels tall on-screen. In actuality, due to browser zooming, retina displays, or other factors, this may or may not match to 20 physical on-screen pixels.

px are useful for both horizontal and vertical dimensions.

EM

'em' is a typographic term that has come to the Web. On the Web, em units are usually used for vertical dimensions. One 'em' maps to the height of one capital letter in the *parent context*.

```
p  { font-size: 0.9em; }  /* text in a paragraph tag is smaller than its parents */
h1 { font-size: 1.2em; }  /* but an h1 will be bigger than the parent */
i  { font-size: 0.5em; }  /* and any italicized text will be half as big. */
```

All the text sizes above are relative to the pages base size. You'll see radically different results on the rest of the page from either of these rules applied to the body, but relative to one another they'll remain sized correctly.

```
html, body { font-size: 50px; } /* 50 px base text size */
```

```
html, body { font-size: 20px; } /* 20 px base text size */
```

REM

'rem' is much like 'em'. But while 'em' sizes an element relative to its *parent*, 'rem' always derives its size relative to the ***root***. In an HTML document with lots of nested elements, 'rem' will generally prove to be more reliable than 'em'. rem is supported in all modern day browsers, including mobile. But not older ones.

Using the CSS rules from the em section immediately above, nested paragraphs (`<p>one<p>two</p></p>`)would get increasingly smaller. But if rem units were used, they would be the same size.

Note to ensure you are setting the root size, use ***both*** the `html` and`body` selectors.

```
   html, body { font-size: 20px; }
```

%

Whereas em is a measure relative to the parents text size, the percentage unit (%) is relative to the parent dimension. This is a useful unit for both horizontal and vertical dimensions, though often more useful in the horizontal.

```
p {
  margin-left:  10%;
  margin-right: 10%;  /* 10% of parent width will be spent on the two
side margins */
 }
```

Initially, the percentage unit may seem very handy (and it is), and many developers fall in love with it. But the love affair is usually short lived. One of the limitations of this rule is that for it to work correctly, the parent must have an explicit width or height set. This limitation is particularly noticeable in the vertical dimension. If the parent element doesn't have an explicit height set then child percentages may be percentages of 0.

VH / VW

'vh' stands for viewport height, and 'vw' for viewport width. The vh and vw units work much

like the percentage (%) unit. But instead of percentage of the parent, it is percentage of the screen (aka viewport). Obviously, vh is for vertical dimensions, and vw for horizontal dimensions.

vh and vw do not suffer the parent limitation that the % unit does. Most modern browsers support these units, but there are some exceptions on older mobile browsers.

```
p {
  margin-left:  10vw;
  margin-right: 10vw;  /* 10% of screen width will be spent on the two side margins */
 }
```

EXTERNAL RESOURCES

The list of CSS units above is not exhaustive. There are various tutorials and explanations about CSS units on the internet. Here are a few that you might find helpful.

- This CSS Tricks article from March 2016: "Use for Global Sizing; Use for Local Sizing"
- New CSS3 Units: Root EM and Viewport Units
- From the W3C specification: Viewport-percentage lengths