Guide Collection

Centering in CSS: A Complete Guide

Published
Sep 2, 2014

Updated
May 29, 2019

Centering things in CSS is the poster child of CSS complaining. Why does it have to be so hard? They jeer. I think the issue isn't that it's difficult to do, but in that there so many different ways of doing it, depending on the situation, it's hard to know which to reach for. So [...]

Centering things in CSS is the poster child of CSS complaining. *Why does it have to be so hard?* They jeer. I think the issue isn't that it's difficult to do, but in that there so many different ways of doing it, depending on the situation, it's hard to know which to reach for.

So let's make it a decision tree and hopefully make it easier.

I need to center...

▼ Horizontally (#center-horizontally)

▼ Is it inline or inline-* elements (like text or links)? (#horizontal-inline)

You can center inline elements horizontally, within a block-level parent element, with just:

css

```css
.center-children {
  text-align: center;
}
```

Embedded Pen Here

This will work for inline, inline-block, inline-table, inline-flex, etc.

▼ Is it a block level element? (#horizontal-block)

You can center a block-level element by giving it `margin-left` and `margin-right` of `auto` (and it has a set `width`, otherwise it would be full width and wouldn't need centering). That's often done with shorthand like this:

css

```css
.center-me {
  margin: 0 auto;
}
```

Embedded Pen Here

This will work no matter what the width of the block level element you're centering, or the parent.

Note that you can't `float` an element to the center. There is a trick though. (https://css-tricks.com/float-center/)

▼ Is there more than one block level element? (#horizontal-multiple-block)

If you have two or more block-level elements that need to be centered horizontally *in a row*, chances are you'd be better served making them a different `display` type. Here's an example of making them `inline-block` and an example of flexbox:

Embedded Pen Here

Unless you mean you have multiple block level elements stacked on top of each other, in which case the auto margin technique is still fine:

Embedded Pen Here

▼ Vertically (#center-vertically)

Vertical centering is a bit trickier in CSS.

▼ Is it inline or inline-* elements (like text or links)? (#vertical-inline)

▼ Is it a single line? (#vertical-inline-single)

Sometimes inline / text elements can appear vertically centered, just because there is equal padding above and below them.

css

```css
.link {
  padding-top: 30px;
  padding-bottom: 30px;
}
```

Embedded Pen Here

If padding isn't an option for some reason, and you're trying to center some text that you know will not wrap, there is a trick were making the `line-height` equal to the height will `center` the text.

css

```
.center-text-trick {
  height: 100px;
  line-height: 100px;
  white-space: nowrap;
}
```

Embedded Pen Here

▼ Is it multiple lines? (#vertical-inline-multiple)

Equal padding on top and bottom can give the centered effect for multiple lines of text too, but if that isn't going to work, perhaps the element the text is in can be a table cell, either literally or made to behave like one with CSS. The `vertical-align` (https://css-tricks.com/almanac/properties/v/vertical-align/) property handles this, in this case, unlike what it normally does which is handle the alignment of elements aligned on a row. (More on that. (https://css-tricks.com/what-is-vertical-align/))

Embedded Pen Here

If something table-like is out, perhaps you could use flexbox? A single flex-child can be made to center in a flex-parent pretty easily.

css

```css
.flex-center-vertically {
  display: flex;
  justify-content: center;
  flex-direction: column;
  height: 400px;
}
```

Embedded Pen Here

Remember that it's only really relevant if the parent container has a fixed height (px, %, etc), which is why the container here has a height.

If both of these techniques are out, you could employ the "ghost element" technique, in which a full-height pseudo-element is placed inside the container and the text is vertically aligned with that.

css

```css
.ghost-center {
  position: relative;
}
.ghost-center::before {
  content: " ";
  display: inline-block;
  height: 100%;
  width: 1%;
  vertical-align: middle;
}
.ghost-center p {
  display: inline-block;
  vertical-align: middle;
}
```

Embedded Pen Here

▶ Is it a block-level element? (#vertical-block)

▼ Both Horizontally and Vertically (#center-horizontally-and-vertically)

You can combine the techniques above in any fashion to get perfectly centered elements. But I find this generally falls into three camps:

▼ Is the element of fixed width and height? (#both-fixed)

Using negative margins equal to half of that width and height, after you've absolutely positioned it at 50% / 50% will center it with great cross browser support:

css

```css
.parent {
  position: relative;
}

.child {
  width: 300px;
  height: 100px;
  padding: 20px;

  position: absolute;
  top: 50%;
  left: 50%;

  margin: -70px 0 0 -170px;
}
```

Embedded Pen Here

▼ Is the element of unknown width and height? (#both-unknown)

If you don't know the width or height, you can use the transform property and a negative translate of 50% in both directions (it is based on the current width/height of the element) to center:

css

```css
.parent {
  position: relative;
}
.child {
  position: absolute;
  top: 50%;
  left: 50%;
  transform: translate(-50%, -50%);
}
```

Embedded Pen Here

▼ Can you use flexbox? (#both-flexbox)

To center in both directions with flexbox, you need to use two centering properties:

css

```css
.parent {
  display: flex;
  justify-content: center;
  align-items: center;
}
```

Embedded Pen Here

▼ Can you use grid? (#both-grid)

This is just a little trick (sent in by Lance Janssen) that will pretty much work for one element:

css

```css
body, html {
  height: 100%;
  display: grid;
}
span { /* thing to center */
  margin: auto;
}
```

Embedded Pen Here

▼ Conclusion (#conclusion)

You can totally center things in CSS.