CSS – margin:auto; – How it Works

By Preethi Ranjit in Coding. Updated on November 27, 2018.

Using `margin:auto` to center a block element horizontally is a well known technique. But have you ever wondered why or how it works? To answer this, we first need to take a look at how margin:auto works. Also in the mix is what `auto` can possibly do in margins, if it works for vertical centering, and a few other issues.

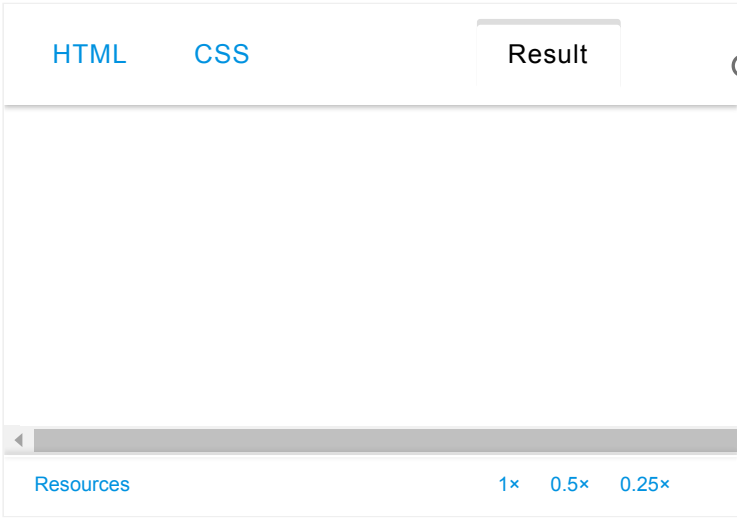
But first, **what does `auto` actually do?**

The definition of `auto` varies with **elements**, **element types** and **context**. In margins, `auto` can mean one of two things: take up the available space or 0 px. These two will **define different layouts for an element**.

> **Read more:** 6 CSS Tricks to Align Content Vertically

"auto" Taking Up Available Space

This is the most common use of margin `auto` we come across often. By assigning `auto` to the left and right margins of an element, they take up the available horizontal space in the element's container equally – and thus the element gets centered.



However, this will work for horizontal margins only (more on the *why* later), and it also **won't work with floated** and **inline elements** and by itself, it also **cannot work** in **absolute** and **fixed positioned elements** (we will however see how to make those work).

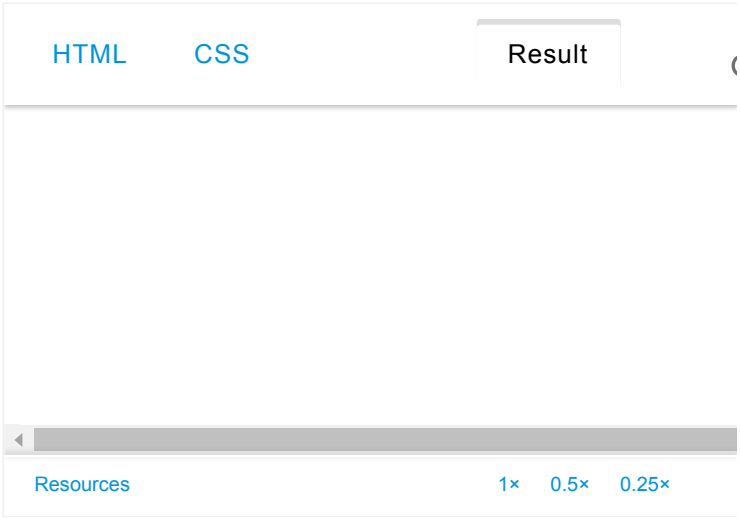
Faux Float By Taking Up Available Space

Since `auto` in both right and left margins take up the "available" space equally, what do you think will happen when the value `auto` is given to only one of those?

CSS Floats Explained in Five Questions
Preethi Ranjit

20 Cool Stuff Built Using CSS
Agus

A left or right margin with `auto` will take up all of the "available" space making the element look like it has been flushed right or left.



"auto" Computed to 0px

As mentioned before, `auto` will not work in floated, inline and absolute elements. All these elements already have **decided on their layouts**, so there is no use in using `auto` for the margins and expecting it to get centered just like that.

That will defeat the initial purpose of using something like `float`. Hence `auto` will have a value of 0px in those elements.

`auto` will also not work on a typical block element if it doesn't have a width. All the examples I showed you so far have widths.

A width of value `auto` will have `0px` margins. A block element's width typically covers its container's when it is `auto` or `100%` and hence a margin `auto` will be computed to `0px` in such a case.

What Happens to Vertical Margins With The Value `auto`?

`auto` in both top and bottom margins is always computed to 0px (except for absolute elements). W3C spec says it like this:

"If "margin-top" or "margin-bottom" is "auto", their used value is 0"

The why, well that is so far, a mystery. It could be because of the typical vertical page flow, where **page size increases height-wise**. So, centering an element vertically in its container is not going to make it appear centered, relative to the page itself, unlike when it's done horizontally (in most cases).

And maybe it's because of this same reason, they decided to add an exception for absolute elements which can be centered vertically along the entire page's height.

It could also be because of the margin collapse effect (a collapse of adjacent

elements" margins) which is another exception for the vertical margins.

However, the latter seems to be an unlikely case – since elements which don't collapse their margins – like Floats, and elements with `overflow` other than `visible`, still assign 0px vertical margins for `auto`.

Centering Absolutely Positioned Elements

Since there happens to be an exception for absolutely positioned elements, we"ll use `auto` value to center one vertically and horizontally. But before that, we need to find out when will `margin:auto` actually work like we want it to in an absolutely positioned element.

This is where another W3C spec comes in:

"If all three of "left", "width", and "right" are "auto": First set any "auto" values for "margin-left" and "margin-right" to 0…"

"If none of the three is "auto": If both "margin-left" and "margin-right" are "auto", solve the

equation under the extra constraint that the two margins get equal values"

That pretty much says that for **horizontal** `auto` **margins** to seize equal spaces, the **values for** `left` **,** `width` **and** `right` **shouldn't be** `auto` , their default value. So all we have to do is to give them some value in an absolutely positioned element. `left` **and** `right` should have **equal values for perfect centering**.

The spec also mentions something similar for vertical margins.

"If all three of "top", "height", and "bottom" are auto, set "top" to the static position…"

"If none of the three are "auto": If both "margin-top" and "margin-bottom" are "auto", solve the equation under the extra constraint that the two margins get equal values…"

Hence, for an absolute element to be **centered vertically**, its `top` **,** `height` **, and** `bottom` **values shouldn't be** `auto` .

Now by combining all these, this is what we"ll get:

Conclusion

If you ever want to flush an element on your page to right or left without the following elements wrapping it (like whats happens with float), remember there's the option to use `auto` for margins.

Converting an element to absolute just so it can be centered vertically may not be a great idea. There are other options like flexbox and CSS transform which are more suitable for those.

Show Comments