CSS box model

CSS box model

Before we get too far into debugging, it's helpful to understand a couple of things about CSS more deeply.

The placement of elements on a Web page can be fairly complicated. One of the most basic features that influence where things go on a Web page is the CSS *Box Model*. The Box Model governs 3 important spacing features of CSS. We learned about *margins* previously as the space between elements. There are two other similar notions, *padding* and *borders*.

Perhaps the best way to understand is with a picture. All elements in an html document end up being treated as rectangles somewhere in the window. The *content* of each rectangle corresponds to the innermost rectangle in the image below. Just outside the content is the padding. This is kind of like an internal margin, meaning that it separates the contents from the *border*. The border essentially traces the sides of the padding rectangle.

It's important to note that the border goes around the *content* and the padding. There are sometimes visible things associated with an element that are not technically part of the content of the element. One such example is the list item:

- I'm in a blue box

The box does not include the bullets because it is outside of the content. Sometimes when you see that it might be a bit confusing, especially because it also affects padding (which is inside the border).

- padding-left: 1rem;

There is a list-style option , list-style-position, which can be used to include the bullet as part of the content:

```
- list-style-position: inside; padding-left: 1rem;
```

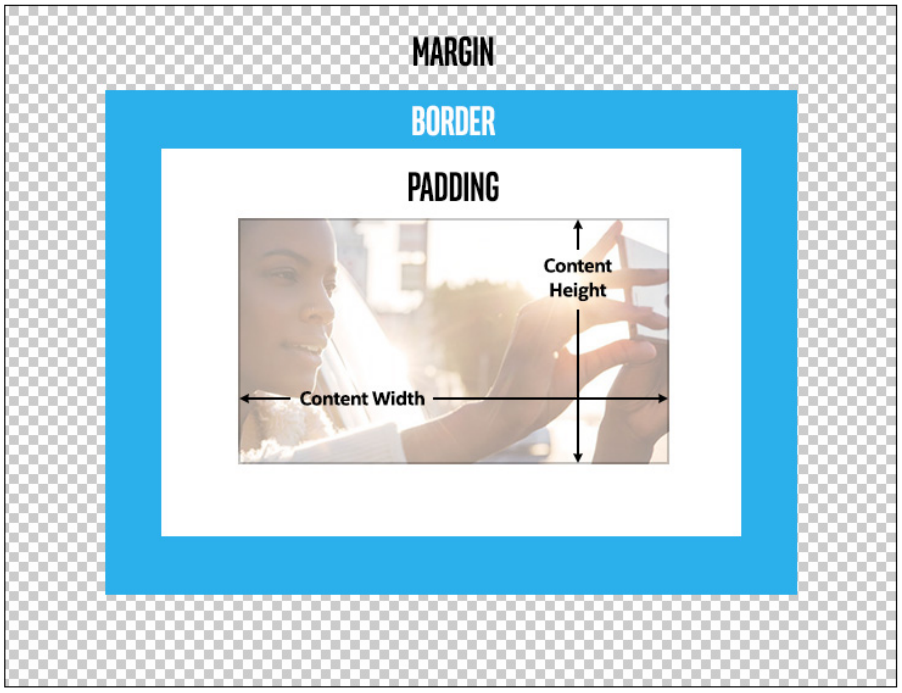
Now the bullet is inside the border, and padding affects the bullet as well as the text.

The border property has a lot more options than the padding or margin. Imagine using a pen to draw the edges of a rectangle. You can choose how thick the pen is, and whether you draw a solid or dotted line. You can even choose how you go around the corners, whether it's a sharp turn or a more gradual circular shape. All of these characteristics can be controlled by CSS properties, like `border-width`, `border-style` and `border-radius`.

While all of these border properties have default values, there are three that you'll see most often when specifying a border: `border-width` (the size of your imaginary pen), `border-style` (dashed, dotted, solid, etc.) and `border-color` (the color of your pen). In fact these are so commonly specified that there is a shorthand syntax to set all three in one line:

```
border: 5px solid blue;
```

There are many other shortcuts to learn, but this one is fairly common. To draw a border you need to know the width, style and color. There are defaults for these values, so you technically don't need to specify all of them, but it is the minimal info needed and is quite common.



The margin, as we learned earlier, specifies the position of the element relative to whatever is adjacent to it, either to the right or left, or top or bottom. The margin is always transparent, and each side can be set individually. The unique thing about the margin is that the values for any of the sides can be *negative,* even if that means that it overlaps with another element on the page. This can be useful when you want to control where an element is placed on a page. In the following pictures, the black rectangles encompass the content:

On the left, we see three blocks with no margins between them. On the right are the same 3 blocks, but now block 2 has a positive margin-left, creating space between blocks 1 and 2. Block 3 has a negative margin-left, causing its left side to overlap with block 2.

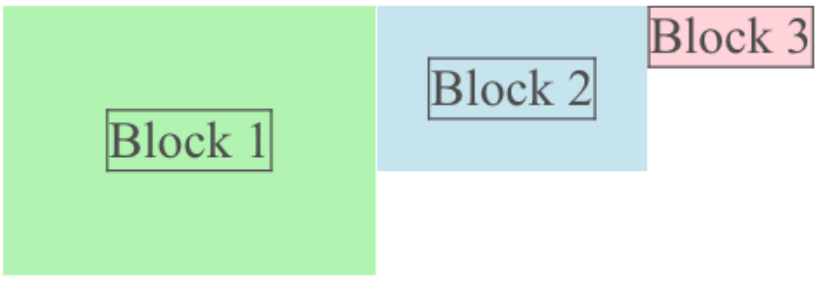
The border may be easier to comprehend because it is often visible, though it doesn't have to be. Unlike the margin (or the padding), there are many more options controlling the size, shape, color and style of the border. You can even create a completely or partially transparent border, or you can match the color of the border to the color of the background, essentially rendering it invisible. It's still there, though, taking up some space on the page and influencing the placement of elements

displayed in the browser. The width of the border controls it's size (thickness), so it only makes sense to accept numbers greater than or equal to 0. What the browser does if the border-width is less than 0 is undefined and shouldn't be relied on.

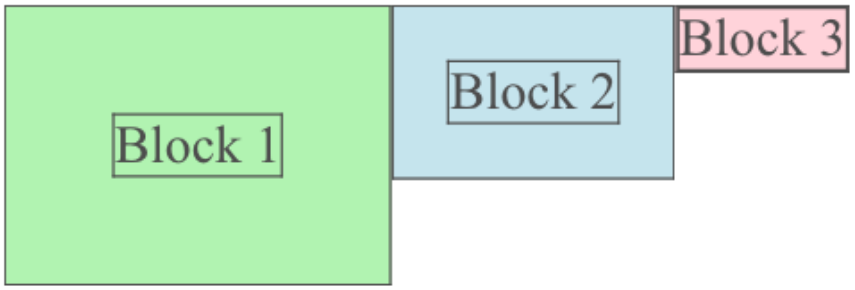
Here blocks 1, 2 and 3 all have borders with different widths, but the margins are zero. There is no overlap, the borders are up against each other regardless of their width. The contents have different positions, influenced by the width of the border. If we were to make the borders invisible (fully transparent), the positioning of the contents would be the same.

Inside the border is the padding. This controls the amount of space between the elements content and the border box (whether it's visible or not). If you have no padding, then the contents of the element (maybe text or image) would be right up against the border, which could be awkward if you have a visible border.

Like the margin and the border, all four sides can be independently set. The background of the padded area matches the background of element, so the effective visible size of the element includes the padding.



Here we've got a thin border directly around the content to delineate where the content ends and the padding begins. Again, the contents are affected by the width of the padding, but now the background of the padding is the same as the background of the content. This makes it look more like the contents have been expanded. If we add a thin border to these, we see that the padding is reflected by empty space between the contents and the border:



All of these can have a width of 0, which is equivalent to not having them, thus 'margin: 0;' is the same as 'margin: none;'. Each can be controlled individually with relative or absolute lengths. While the padding and borders require non-negative widths, margins can be either positive or negative.

Using these three settings in combination provides quite a bit of flexibility in terms of spacing and drawing of borders. If you have padding and a visible border, you can control how close the border comes to the contents. By setting the margin you can control how close the border comes to surrounding elements. You can even give your border rounded corners using the `border-radius` setting: