**CSS Basics #3: Position, Overflow and Alignment -** [Eric Hu](https://huericnan.medium.com/?source=post_page-----3029ac45c1f--------------------------------) Feb 7, 2022

In this article, we are going to focus on how to define the position of an HTML element using CSS.

*You can download the source code for this tutorial here:*

[*Download the Source Code*](https://www.ericsdevblog.com/index.php/beginners-roadmap-to-web-development/)

**Boxes**

Everything in CSS can be seen as a “box”, and understanding these boxes enables us to create a layout for our web page. In general, the box has three components, border, margin and padding. The border can be seen as the box itself, the margin is the space around the box, and the padding is the space between the box and its content

Diagram, PowerPoint

Description automatically generated

Let’s take a look at some examples.

**Border**

<body>  
 <div class="box1">  
 <p>...</p>  
 </div> <div class="box2">  
 <p>...</p>  
 </div>  
</body>.box1 {  
 border-style: double;  
}.box2 {  
 border-style: dashed;  
}

Text

Description automatically generated with medium confidence

There are several other border properties that we can use here. For example, border-width can be used to define the thickness of the border and border-color can define the colour of the border. Please note that these border properties will not have any effect unless the border-style property is set.

**Margin**

.box1 {  
 border-style: double;  
 margin: 10px;  
}.box2 {  
 border-style: dashed;  
 margin: 20px;  
}

Table

Description automatically generated

**Padding**

.box1 {  
 border-style: double;  
 margin: 10px;  
 padding: 20px;  
}.box2 {  
 border-style: dashed;  
 margin: 20px;  
 padding: 40px;  
}

Text

Description automatically generated

As you probably already noticed, after we added padding to the box, the vertical space between the box and the content is bigger than the horizontal space. That is because all the inline blocks/boxes, in this case, the <p> element, has a default vertical margin even if you did not define it.

**Positions**

The position property specifies the type of positioning method used for an element.

There are five different position methods:

* static
* relative
* fixed
* absolute
* sticky

Elements are then positioned using the top, bottom, left, and right properties. However, these properties will not work unless the position method is set first. They also work differently depending on the position value.

The static method is the default option and the element with the static method is not positioned in any special way.

The relative method positions an element relative to its normal (static) position, meaning you can add top, bottom, left, and right properties to adjust its position.

<body>  
 <div class="div-static">This div element has position: static;</div>  
 <div class="div-relative">This div element has position: relative;</div>  
</body>.div-static {  
 border: solid;  
 position: static;  
}.div-relative {  
 border: solid;  
 position: relative;  
 left: 5px;  
 top: 10px;  
}

A picture containing text

Description automatically generated

The fixed method place the element at a fixed location relative to the viewport. For example, those ads that pop up at the bottom right corner of your screen/browser, all have a fixed position, because even if you scroll the page, it always stays at the bottom right corner.

The sticky method is the combination of the relative and the fixed method. A sticky element is, by default, positioned relatively, until a given offset is met in the viewport, then it will "stick" to the place.

It is difficult to demonstrate the fixed and the sticky method here, but if you are interested, there are some examples from w3schools: <https://www.w3schools.com/css/css_positioning.asp>

Finally, we have the absolute method. An absolute element is positioned relative to the nearest positioned element (anchor). A positioned element is one whose position is anything but "static". For instance, in the following example, the absolute element is positioned relative to the relative element:

<body>  
 <div class="relative">  
 This div element has position: relative;  
 <div class="absolute">This div element has position: absolute;</div>  
 </div>  
 </body>.relative {  
 border: solid;  
 position: relative;  
 width: 400px;  
 height: 200px;  
}.absolute {  
 border: solid;  
 position: absolute;  
 top: 80px;  
 right: 0;  
}

Text

Description automatically generated

**Overflow**

The overflow property controls what happens when the content is too bid to fit into the box.

The default value is visible. The overflow will be rendered outside of the box.

<body>  
 <div>  
 ...  
 </div>  
</body>div {  
 border-style: solid;  
 width: 200px;  
 height: 50px;  
 overflow: visible;  
}

A picture containing text, person, screenshot, document

Description automatically generated

As for the hidden option, the content will be clipped, and the overflow will be hidden.

div {  
 border-style: solid;  
 width: 200px;  
 height: 50px;  
 overflow: hidden;  
}

Text

Description automatically generated

Setting the value to scroll, the overflow is clipped and a scrollbar is added to scroll inside the box. Note that this will add a scrollbar both horizontally and vertically (even if you do not need it):

div {  
 border-style: solid;  
 width: 200px;  
 height: 50px;  
 overflow: scroll;  
}

Graphical user interface, text

Description automatically generated

Finally, the auto option is very similar to scroll, but it adds scroll bars only when necessary.

div {  
 border-style: solid;  
 width: 200px;  
 height: 50px;  
 overflow: auto;  
}

Text

Description automatically generated

**Align**

**Center Align Boxes**

We can use margin: auto; to horizontally center a block element like this:

<body>  
 <div>  
 ...  
 </div>  
</body>div {  
 border-style: solid;  
 width: 200px;  
 margin: auto;  
}

Text

Description automatically generated

We first set a width for the <div> element, so that it does not stretch out to the edges of the container. The element will then take up the specified width, and the remaining space will be split equally between the two margins.

**Center Align Text**

What if we want the box to stretch out to the edges, and make its content/text center inside the box? To horizontally center the text, we can use text-align: center;. Let's see an example:

<body>  
 <div>  
 <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit.</p>  
 </div>  
</body>div {  
 border-style: solid;  
 text-align: center;  
}

A picture containing text

Description automatically generated

To vertically center the text, the easiest way is to use the top and bottom padding/margin:

div {  
 border-style: solid;  
 text-align: center;  
 padding-top: 50px;  
 padding-bottom: 50px;  
}

Text

Description automatically generated with low confidence

We can also use position and transform or the flexbox to position elements, but they are slightly more advanced topics in CSS. In this article, we’ll only focus on the basics.

**Left and Right Align**

Sometimes, instead of aligning a box at the centre, we want it to stick to the left or the right side of the page. To do that, we can use the absolute position:

<body>  
 <div class="left">  
 This div block should stick to the left.  
 </div> <div class="right">  
 This div block should stick to the right.  
 </div>  
</body>div {  
 border-style: solid;  
 width: 300px;  
}.left {  
 position: absolute;  
 left: 0px;  
}.right {  
 position: absolute;  
 right: 0px;  
}

