# CSS Position: Relative vs Position Absolute

The CSS [position](https://www.w3schools.com/cssref/pr_class_position.asp) property defines, as the name says, how the element is positioned on the web page.

There are several types of positioning: **static, relative, absolute, fixed, sticky, initial, and inherit**. First of all, let's explain what all of these types mean.

* **Static** - this is the default value, all elements are in order as they appear in the document.
* **Relative** - the element is positioned relative to its normal position.
* **Absolute** - the element is positioned absolutely to its first positioned parent.
* **Fixed** - the element is positioned related to the browser window.
* **Sticky** - the element is positioned based on the user's scroll position.

The two most commonly used position values - **relative and absolute**.

**What Is Relative Positioning?**

When you set the position *relative to an element*, without adding any other positioning attributes (top, bottom, right, left) **nothing will happen**. When you add an additional position, such as left: 20px the element will move 20px to the right from its normal position. Here, you can see that **this element is relative to itself**. When the element moves, no other element on the layout will be affected.

*There is a thing you should keep in mind while setting position - relative to an element limits the scope of absolutely positioned child elements. This means that any element that is the child of this element can be absolutely positioned within this block.*

After this brief explanation, we need to back it up, by showing an example.

In this example, you will see how the relatively positioned element moves when its attributes are changed. The first element moves to the **left** and **top** from its normal position, while **the second element stays in the same place** because none of the additional positioning attributes were changed.

**HTML**

<div id="first\_element">First element</div>

<div id="second\_element">Second element</div>

**CSS**

#first\_element {

position: relative;

left: 30px;

top: 70px;

width: 500px;

background-color: #fafafa;

border: solid 3px #ff7347;

font-size: 24px;

text-align: center;

}

#second\_element {

position: relative;

width: 500px;

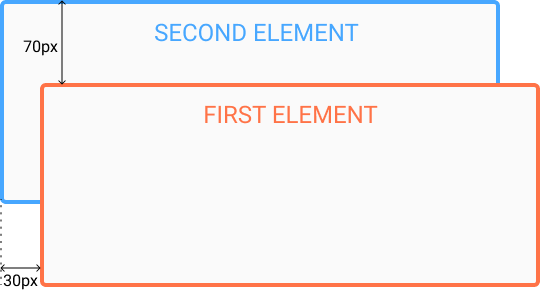
background-color: #fafafa;

border: solid 3px #ff7347;

font-size: 24px;

text-align: center;

}



**What Is Absolute Positioning?**

This type of positioning allows you to **place your element precisely where you want it**.

The positioning is done **relative to the first relatively (or absolutely) positioned parent element**. In the case when there is no positioned parent element, it will be positioned related **directly to the HTML element (the page itself)**.

An important thing to keep in mind while using absolute positioning is to make sure it is **not overused**, otherwise, it can lead to a maintenance nightmare.

In the example, the parent element has the position set to relative. Now, when you set the position of the child element to absolute, **any additional positioning will be done relative to the parent element**. The child element moves relative to the top of the parent element by 100px and right of the parent element by 40px.

### HTML

<div id=”parent”>

<div id=”child”></div>

</div>

### CSS

#parent {

position: relative;

width: 500px;

height: 400px;

background-color: #fafafa;

border: solid 3px #9e70ba;

font-size: 24px;

text-align: center;

}

#child {

position: absolute;

right: 40px;

top: 100px;

width: 200px;

height: 200px;

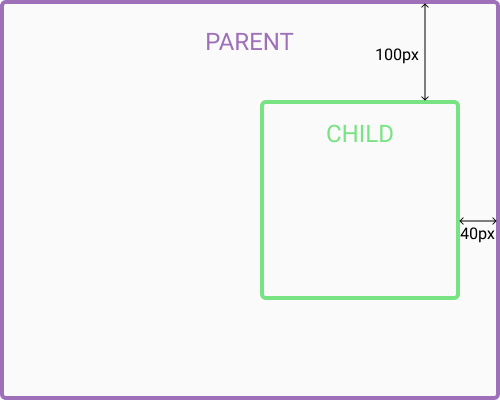
background-color: #fafafa;

border: solid 3px #78e382;

font-size: 24px;

text-align: center;

}

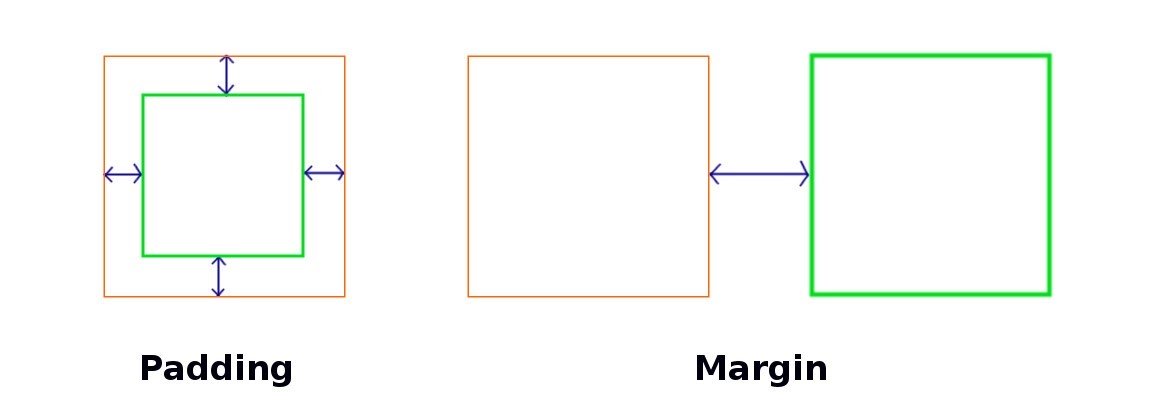


**What’s the Difference Between Margin and Padding in CSS?**

Basically, a margin is the space *around* an element and padding refers to the space between an element and the content *inside* it.

The margin falls outside two adjacent elements. Each side of the element has a margin size you can change individually. In creating the gap, the margin pushes adjacent elements away.

On the other hand, padding is placed inside the border of an element. To create the gap, the padding either grows the element’s size or shrinks the content inside. By default, the size of the element increases.

If you want to create the gap by shrinking the content, set the **box-sizing** property value to **border-box** (i.e. **box-sizing: border-box**).

**When Should You Use Margins vs. Padding?**

When you’re adjusting the layout of your design, you’ll need to determine whether to adjust the margins or the padding to achieve the desired visual effect.

**CSS margins** determine the space surrounding an element, therefore margins would be used to move an element up or down on the page, as well as left or right. If the width of your page is fixed, centering an element horizontally is very simple: just assign the value **margin: auto**.

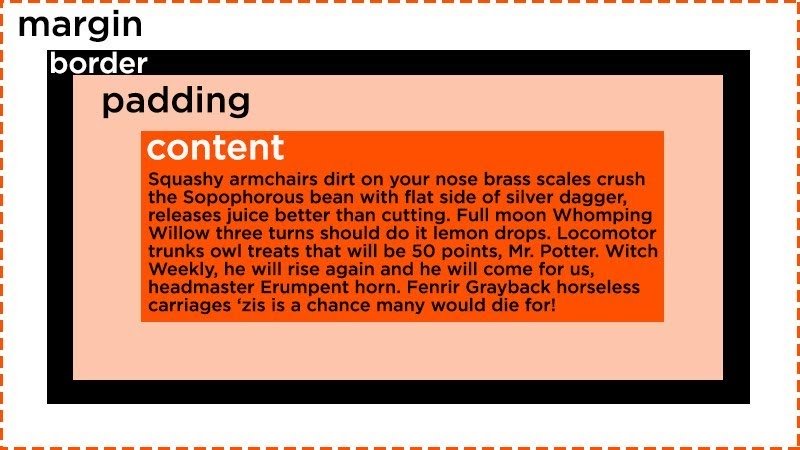
You would also use margins to set the distance between nearby elements. For instance, use margins to add space between images or between an image and the text description below it.

On the flip side, a [**negative margin**](https://www.quirksmode.org/blog/archives/2020/02/negative_margin.html) value would let you overlap elements. You could also use a negative margin to create a header element within a container that has padding values set to keep other elements aligned.

**CSS padding** determines how elements look and sit within a container. You would change the padding if you want to create space between an element and the edge of the container, or the border. This would also show the container’s background around the element inside it.

Padding is also used to change the size of an element. When you increase the padding value, the text will stay the same size, but there will be more space around it. The element will also fill more space inside the container.

**The CSS Box Model**

In CSS, the box model is used for page design and layout. Essentially, every HTML element in a document is wrapped inside a box. A CSS box consists of the margin, border, padding, and content — as shown below:

Applying height and width to your elements will be easier once you understand the CSS box model. To ensure proper alignment, you’ll just need to do some simple math. However, if you’re confused about how the box model works, you could end up with a sloppy layout.

**CSS Margin vs. Padding vs. Border**

Margin and padding are always parts of an element, even if there’s no visible border. This can get confusing for beginners.

This image illustrates such a scenario:



The two blocks of content don’t have a border, but the margin and padding still apply.

Now, let’s look at the difference between the margin and padding CSS codes.

**How to Add Margins in CSS**

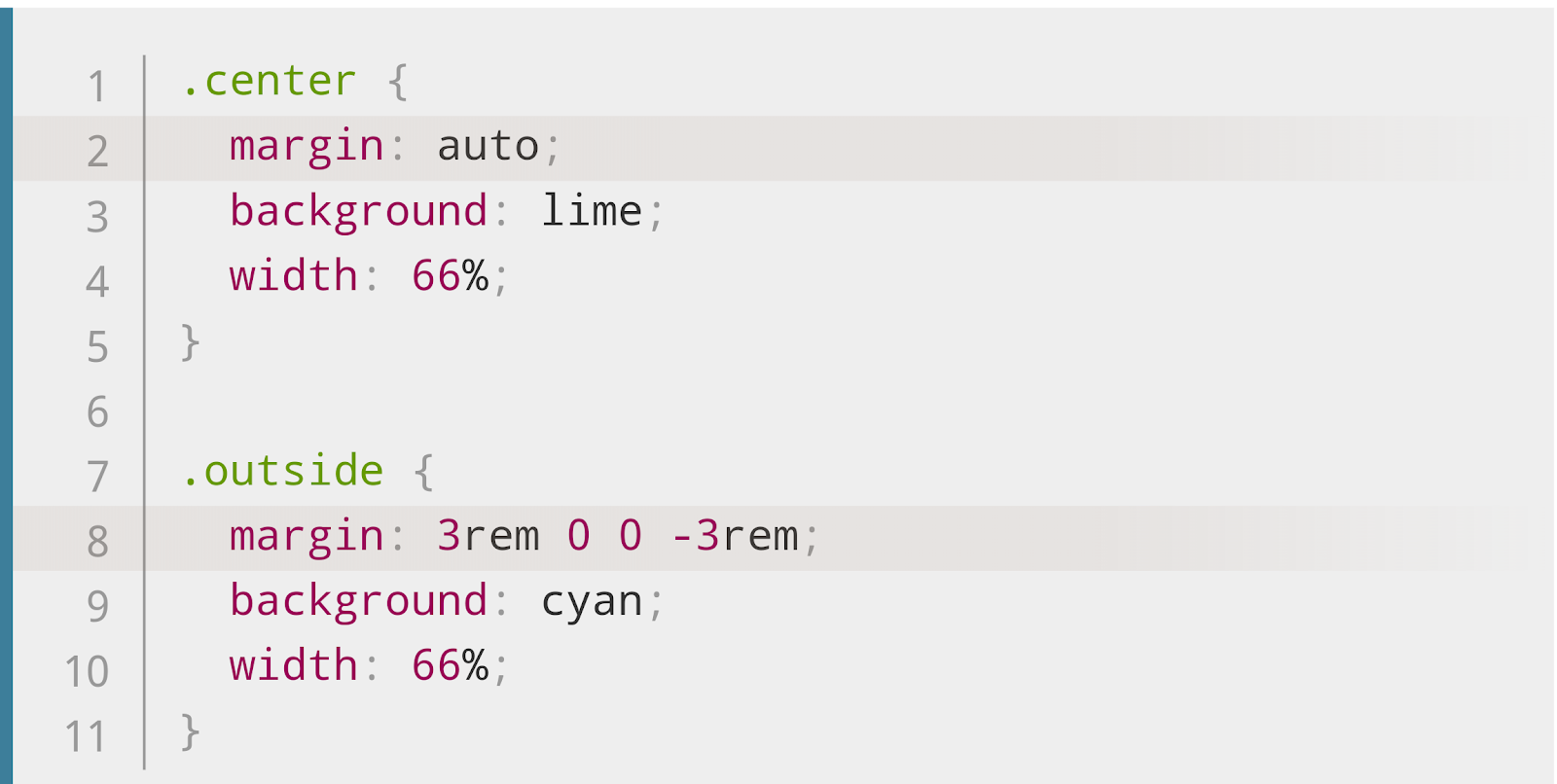
Each element has four margins to be declared: top, right, bottom, and left.

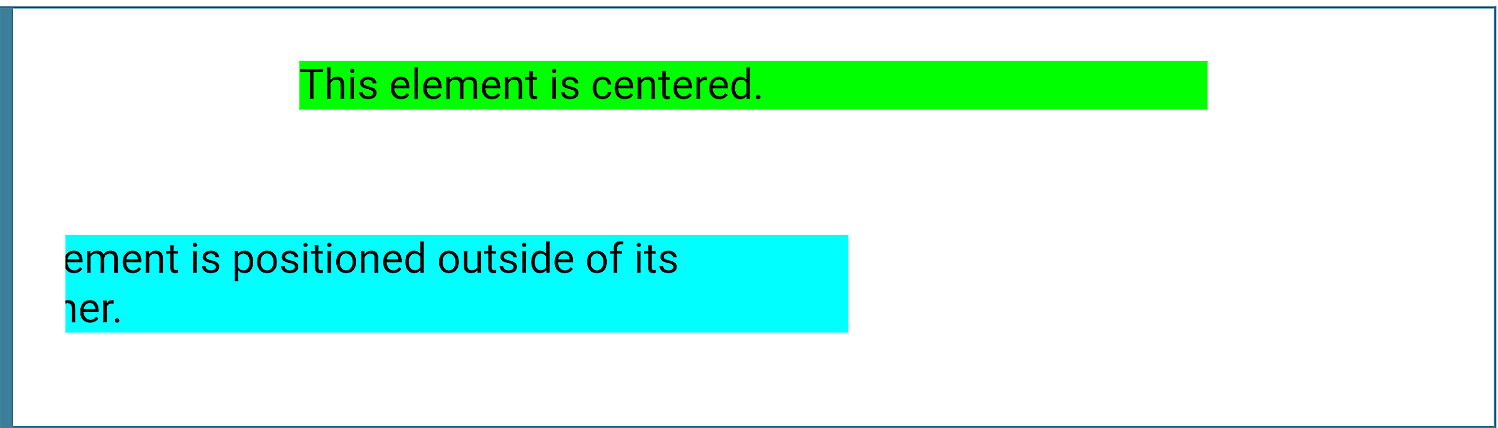
To set the margin area for a side of an element, use **margin-top, margin-right, margin-bottom,** and **margin-left** properties respectively. You can also set the margin on all four sides using the shorthand property **margin**.

You may specify the margin property with one, two, three or four values depending on the side you want to apply it to. If you want even margins on every side, you’ll only need to apply one value. Otherwise, the order of the values is crucial:

* **Four values** apply to the top, right, bottom, then left.
* **Three values** apply to the top, right and left, then bottom.
* **Two values**apply to the top and bottom, then right and left.

Each value is represented as **<length>**(which defines a fixed value for the margin), **<percentage>**(which defines it as a percentage of the width of the container), or **auto**(which lets the browser set the margin)**.**





**How to Add Padding in CSS**

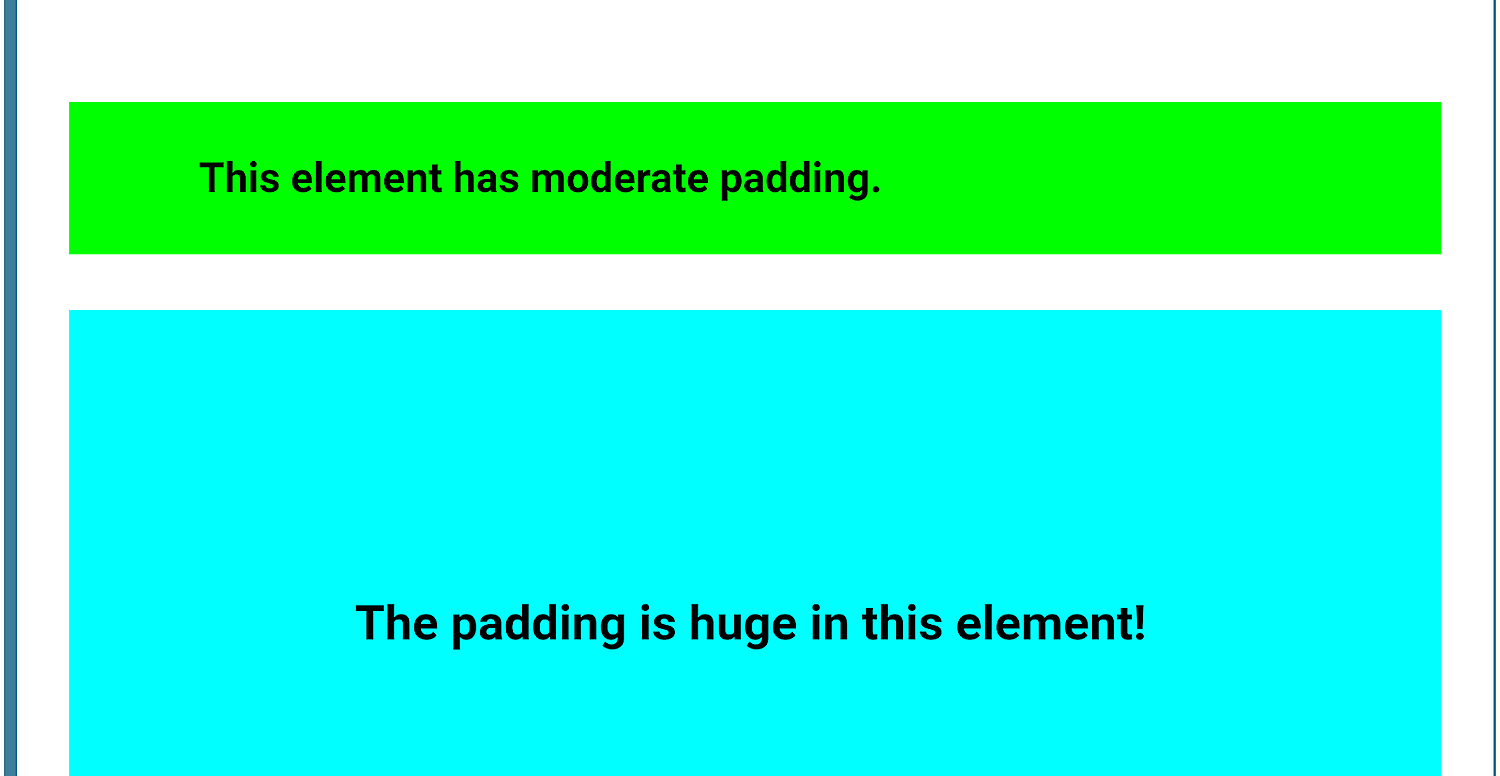
Like in margins, the padding area has four sides to be declared: top, right, bottom, and left.

To set the padding, **use padding-top, padding-right, padding-bottom,**and**padding-left** properties. Alternatively, you can use the shorthand property **padding**.

When using the shorthand property, you may define the padding using one, two, three or four values as follows. As with margins, one value will apply on all four sides. Otherwise, the order the values are written will determine which sides each applies to:

* Four values apply to the top, right, bottom, then left.
* Three values apply to the top, right and left, then bottom.
* Two values apply to the top and bottom, then right and left.

Each value may be represented as **<length>** (which defines it by a fixed value) or **<percentage>** (which defines it as a percentage of the width of the container).



**Resources:**

* <https://dzone.com/articles/css-position-relative-vs-position-absolute#:~:text=Relative%20%2D%20the%20element%20is%20positioned,related%20to%20the%20browser%20window.>
* <https://blog.hubspot.com/website/css-margin-vs-padding#:~:text=What's%20the%20Difference%20Between%20Margin,and%20the%20content%20inside%20it.&text=The%20margin%20falls%20outside%20two,size%20you%20can%20change%20individually.>