Position Types

Introduction

Welcome to the lesson about the CSS position types. The CSS position property defines, as the name says, how the element is positioned on the web page. The top, right, bottom, and left properties determine the final location of the positioned elements.

The z-index property specifies the stack order of the positioned elements.

CSS position property values

There are five main values a position property can have, and we will discuss them briefly.

`element {position: static;}`

Static is the default value. All HTML elements are positioned static by default. It simply means that elements will be positioned in the order in which they appear in the document. Elements with the static position won't be affected by the top, bottom, left, and right properties.

When we use the term positioned element, we refer to an element whose position value is either relative, absolute, fixed, or sticky. In other words, it's anything except static.

`element {position: relative;}`

An element with position: relative is positioned relative to its normal position. Setting the top, right, bottom, and left properties of a relatively positioned element will cause it to be adjusted away from its normal position. Other content will not be adjusted to fit into any gaps left by the element.

`element {position: fixed;}`

An element with position: fixed is positioned relative to the viewport, which means it always stays in the same place. Even if the page is scrolled, the top, right, bottom, and left properties are used to position the element. A fixed element does not leave a gap in the page where it would normally have been located.

`element {position: absolute;}`

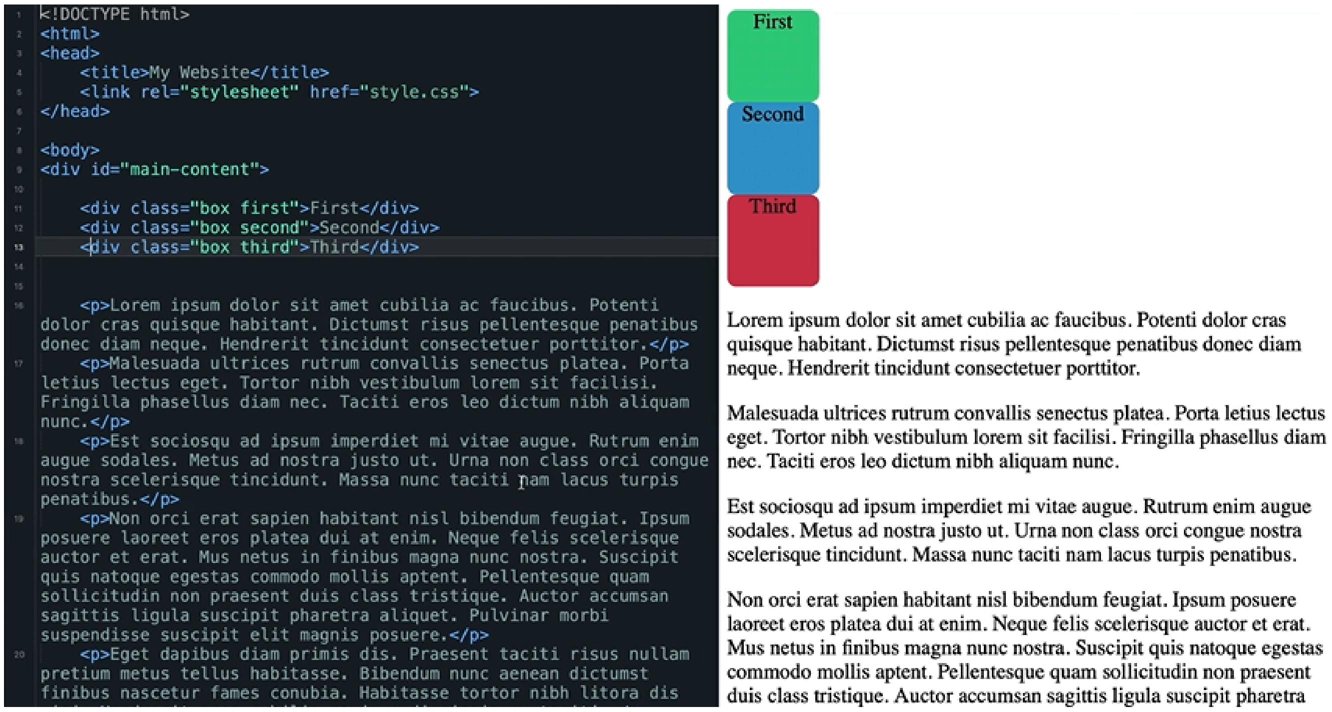
Now, an element with position: absolute is positioned relative to the nearest position ancestor instead of being positioned relative to the viewport like fixed. However, if an absolute positioned element has no position ancestors, it uses the document body and moves along with page scrolling.

`element {position: sticky;}`

An element with position: sticky is positioned based on the user's scroll position. A sticky element toggles between relative and fixed depending on the scroll position. It is positioned relative until a given offset position is met in the viewport, then it sticks in place like position: fixed.

CSS positioning examples

Now, I know this may all sound confusing, so let's look at some examples. This is my HTML structure.



```
<!DOCTYPE html>
<html>
<head>
    <title>My Website</title>
    <link rel="stylesheet" href="style.css">
</head>

<body>
<div id="main-content">

    <div class="box first">First</div>
    <div class="box second">Second</div>
    <div class="box third">Third</div>


    <p>Lorem ipsum dolor sit amet cubilia ac faucibus. Potenti
dolor cras quisque habitant. Dictumst risus pellentesque penatibus
donec diam neque. Hendrerit tincidunt consectetuer porttitor.</p>
    <p>Malesuada ultrices rutrum convallis senectus platea. Porta
letius lectus eget. Tortor nibh vestibulum lorem sit facilisi.
Fringilla phasellus diam nec. Taciti eros leo dictum nibh aliquam
nunc.</p>
    <p>Est sociosqu ad ipsum imperdiet mi vitae augue. Rutrum enim
augue sodales. Metus ad nostra justo ut. Urna non class orci congue
nostra scelerisque tincidunt. Massa nunc taciti nam lacus turpis
penatibus.</p>
    <p>Non orci erat sapien habitant nisl bibendum feugiat. Ipsum
posuere laoreet eros platea dui at enim. Neque felis scelerisque
auctor et erat. Mus netus in finibus magna nunc nostra. Suscipit
quis natoque egestas commodo mollis aptent. Pellentesque quam
sollicitudin non praesent duis class tristique. Auctor accumsan
sagittis ligula suscipit pharetra aliquet. Pulvinar morbi
suspendisse suscipit elit magnis posuere.</p>
    <p>Eget dapibus diam primis dis. Praesent taciti risus nullam
pretium metus tellus habitasse. Bibendum nunc aenean dictumst
finibus nascetur fames conubia. Habitasse tortor nibh litora dis
```

Lorem ipsum dolor sit amet cubilia ac faucibus. Potenti dolor cras quisque habitant. Dictumst risus pellentesque penatibus donec diam neque. Hendrerit tincidunt consectetuer porttitor.

Malesuada ultrices rutrum convallis senectus platea. Porta letius lectus eget. Tortor nibh vestibulum lorem sit facilisi. Fringilla phasellus diam nec. Taciti eros leo dictum nibh aliquam nunc.

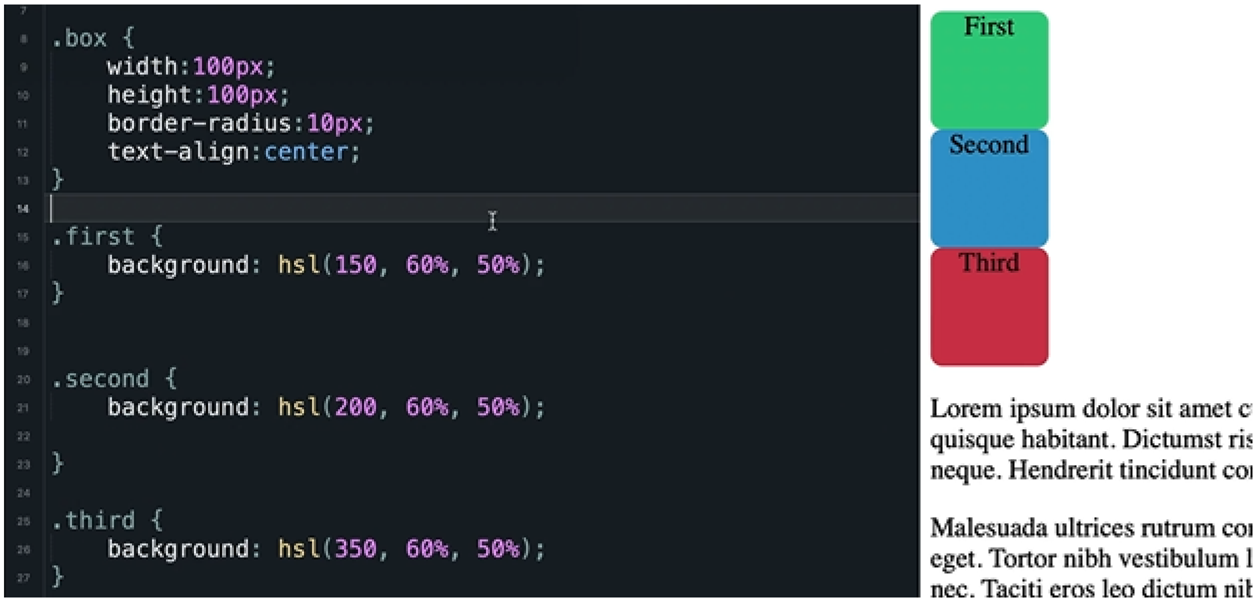
Est sociosqu ad ipsum imperdiet mi vitae augue. Rutrum enim augue sodales. Metus ad nostra justo ut. Urna non class orci congue nostra scelerisque tincidunt. Massa nunc taciti nam lacus turpis penatibus.

Non orci erat sapien habitant nisl bibendum feugiat. Ipsum posuere laoreet eros platea dui at enim. Neque felis scelerisque auctor et erat. Mus netus in finibus magna nunc nostra. Suscipit quis natoque egestas commodo mollis aptent. Pellentesque quam sollicitudin non praesent duis class tristique. Auctor accumsan sagittis ligula suscipit pharetra

```
<body>

<div id="main-content">

<div class="box first">First</div>

<div class="box second">Second</div>

<div class="box third">Third</div>
```

I have three divs here and a few paragraphs. I've included the paragraphs so that I can scroll my preview and show you what happens when scrolling. If we look at the HTML, we can see each div has a class "box", and I have additional classes: first, second, and third.

And in the style sheet, all the boxes have a set width and height and some styling for the border and text alignment. And I added different background colors for each of the classes.



```
.box {
    width:100px;
    height:100px;
    border-radius:10px;
    text-align:center;
}

.first {
    background: hsl(150, 60%, 50%);
}


.second {
    background: hsl(200, 60%, 50%);

}

.third {
    background: hsl(350, 60%, 50%);
}
```

Lorem ipsum dolor sit amet c
quisque habitant. Dictumst ris
neque. Hendrerit tincidunt co

Malesuada ultrices rutrum co
eget. Tortor nibh vestibulum l
nec. Taciti eros leo dictum nil

`.box {width: 100px; height: 100px; border-radius: 10px; text-align: center;}`

`.first {background: hsl(150, 60%, 50%);}`

`.second  {background: hsl(200, 60%, 50%);}`

`.third  {background: hsl(350, 60%, 50%);}`

Static

Each of these elements has a position: static by default. So if I want the second to be moved, if I add left: 20 pixels, nothing happens.

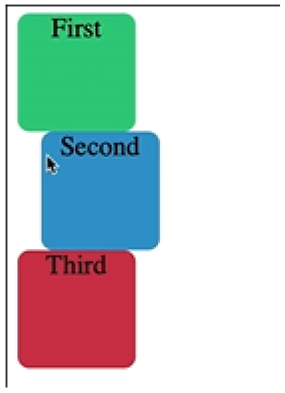
`.second  {background: hsl(200, 60%, 50%); left: 20px;}`

That's because the properties top, bottom, left, and right will not take any effect on elements that have a static position.

Relative

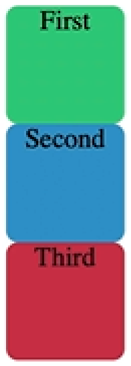
But as soon as I add a position, for example relative, it will take effect.

`.second  {background: hsl(200, 60%, 50%); left: 20px; position: relative;}`



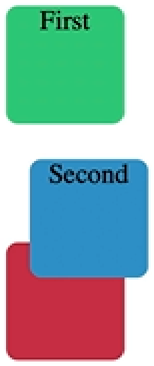
We can see that the property left has moved the element relative to its original position. Removing any top, left, bottom, or right values and just adding position: relative does nothing to an element, it doesn't change its appearance.

```
.second  {background: hsl(200, 60%, 50%); position: relative;}
```



But if I add 20 pixels on the left and let's say top: 30 pixels.

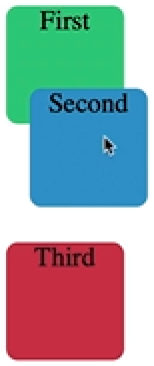
```
.second  {background: hsl(200, 60%, 50%); left: 20px; top: 30px; position: relative;}
```



This will move the element relative to its original position, and as you can see, the gap where the element was originally located is still there. The other elements' positions haven't been changed.

We can also do negative values. For example, top -30px would move it to the top.

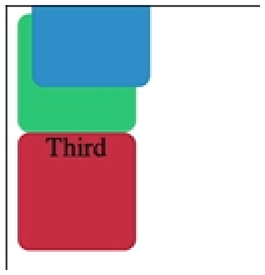
`.second  {background: hsl(200, 60%, 50%); left: 20px; top: -30px; position: relative;}`



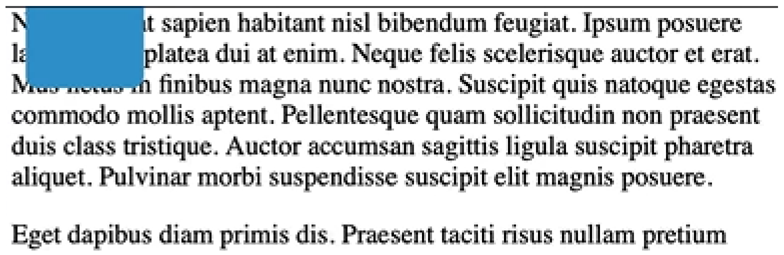
Fixed

Now, what if I set the position to fixed, let's see how that behaves.

`.second  {background: hsl(200, 60%, 50%); left: 20px; top: -30px; position: fixed;}`

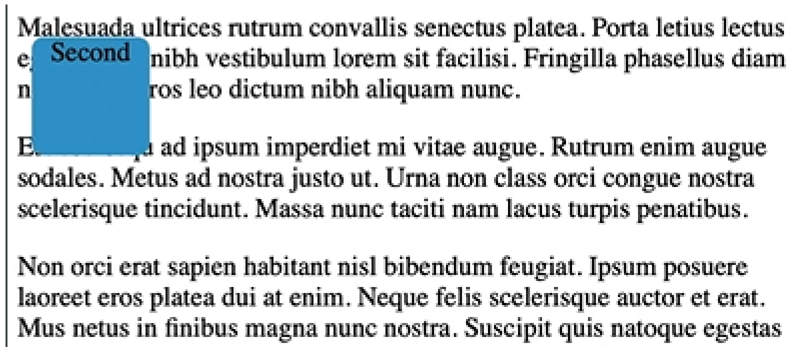


So you can see that it moved the other content. The gap where the element was originally located is no longer here, and if I scroll the page, the element stays fixed, always visible.


t sapien habitant nisl bibendum feugiat. Ipsum posuere
platea dui at enim. Neque felis scelerisque auctor et erat.
n finibus magna nunc nostra. Suscipit quis natoque egestas
commodo mollis aptent. Pellentesque quam sollicitudin non praesent
duis class tristique. Auctor accumsan sagittis ligula suscipit pharetra
aliquet. Pulvinar morbi suspendisse suscipit elit magnis posuere.
Eget dapibus diam primis dis. Praesent taciti risus nullam pretium

But it's positioned relative to the viewport. So a top negative value moves it outside of the viewport. If I define top as 30 pixels, it will be fixed 30 pixels from the top and 20 pixels from the left.

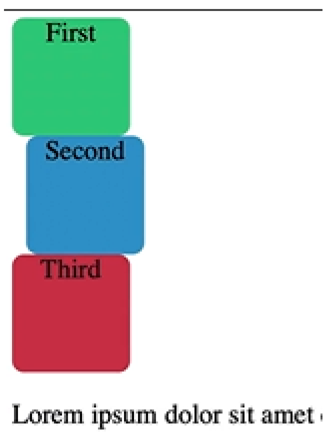
`.second  {background: hsl(200, 60%, 50%); left: 20px; top: 30px; position: fixed;}`


Malesuada ultrices rutrum convallis senectus platea. Porta letius lectus
Second
nibh vestibulum lorem sit facilisi. Fringilla phasellus diam
ros leo dictum nibh aliquam nunc.
ad ipsum imperdiet mi vitae augue. Rutrum enim augue
sodales. Metus ad nostra justo ut. Urna non class orci congue nostra
scelerisque tincidunt. Massa nunc taciti nam lacus turpis penatibus.
Non orci erat sapien habitant nisl bibendum feugiat. Ipsum posuere
laoreet eros platea dui at enim. Neque felis scelerisque auctor et erat.
Mus netus in finibus magna nunc nostra. Suscipit quis natoque egestas

Sticky

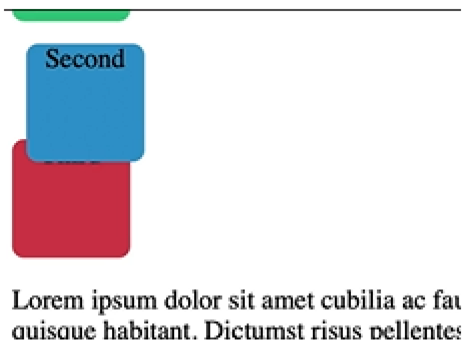
Now position: sticky. And I need you to know that it does not work in Internet Explorer if for whatever reason you need to support it. So position: sticky behaves the same as relative at the beginning, but as soon as users scroll to the position it would have if it were fixed, as soon as it reaches that place, it becomes as if it were fixed. So at first it behaves like position: relative.

```
.second  {background: hsl(200, 60%, 50%); left: 20px; top: 30px; position: sticky;}
```

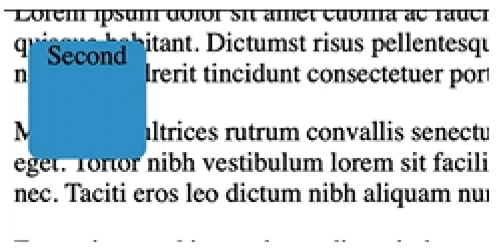


Lorem ipsum dolor sit amet

And as you scroll, it behaves like position: fixed.



Lorem ipsum dolor sit amet cubilia ac fau
quisque habitant. Dictumst risus pellentes



Lorem ipsum dolor sit amet cubilia ac fauci
q[illegible]itant. Dictumst risus pellentesqu
n[illegible]lrerit tincidunt consectetuer port

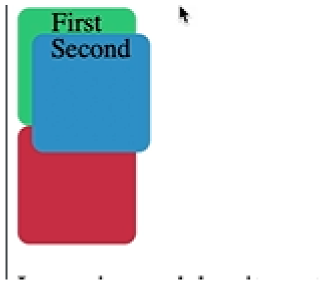
M[illegible]ltrices rutrum convallis senectu
eget. Tortor nibh vestibulum lorem sit facili
nec. Taciti eros leo dictum nibh aliquam nu

That's how position: sticky works.

Absolute

And the final position value is absolute.

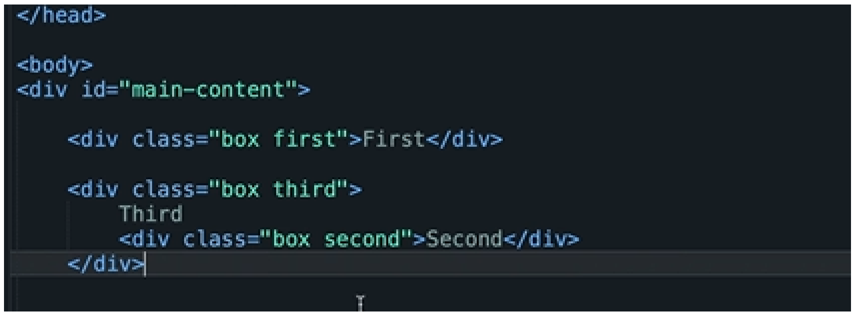
`.second  {background: hsl(200, 60%, 50%); left: 20px; top: 30px; position: absolute;}`



And as you can see, setting the position to absolute moves the element outside of the location it would have been and doesn't leave a gap. So the other elements are placed as if the element is not even here.

And right now, we've set the left and top values, and you can see that it's positioned relative to the viewport, but that is only because it doesn't have any positioned parents.

So let's try moving the second box inside the third. So let's go back to the HTML and I will move that box inside this third div.



```
</head>

<body>
<div id="main-content">

    <div class="box first">First</div>

    <div class="box third">
        Third
        <div class="box second">Second</div>
    </div>
```

`<div class=”box third”>`

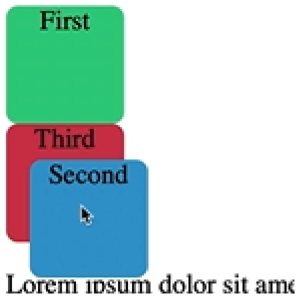
`Third`

`<div class=”box second”>Second</div>`

`</div>`

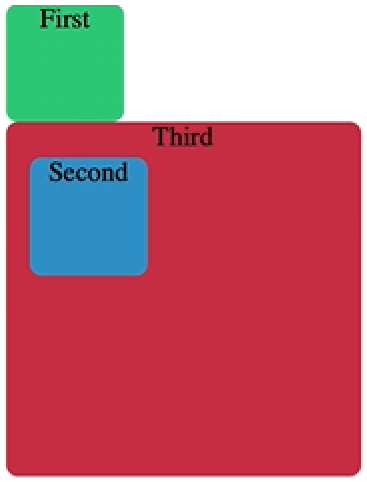
And now, if my third div has position:relative, that second div with position:absolute will be positioned relatively to its parent.

```
.third  {background: hsl(350, 60%, 50%); position: relative;}
```



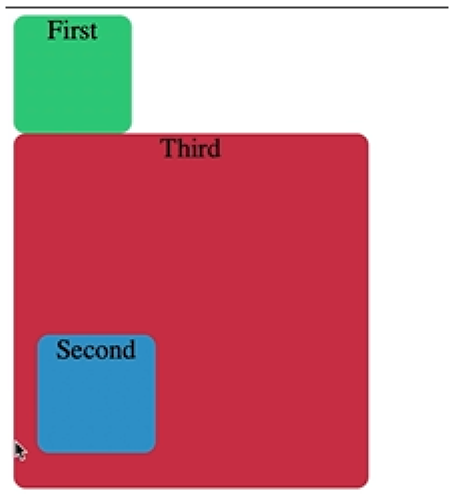
So let's maybe make the Third box a bit bigger, so you can see it better.

```
.third  {background: hsl(350, 60%, 50%); position: relative; width: 300px; height: 300px;}
```



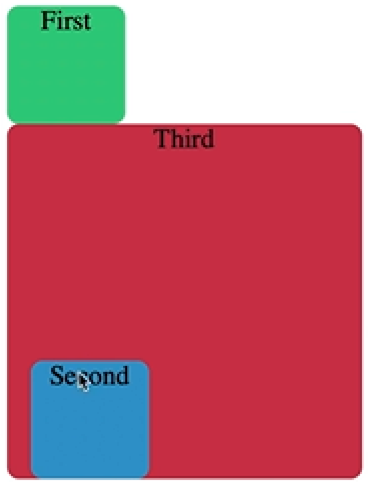
And let’s change the placement of the Second box. Instead of top, that would be bottom.

```
.second  {background: hsl(200, 60%, 50%); left: 20px; bottom: 30px; position: absolute;}
```



Let's try bottom: zero.

```
.second  {background: hsl(200, 60%, 50%); left: 20px; bottom: 0; position: absolute;}
```



So an element with position:absolute will be relative to the parent container. As long as the parent has a position:relative. If it doesn't, the element's position will be relative to the viewport.

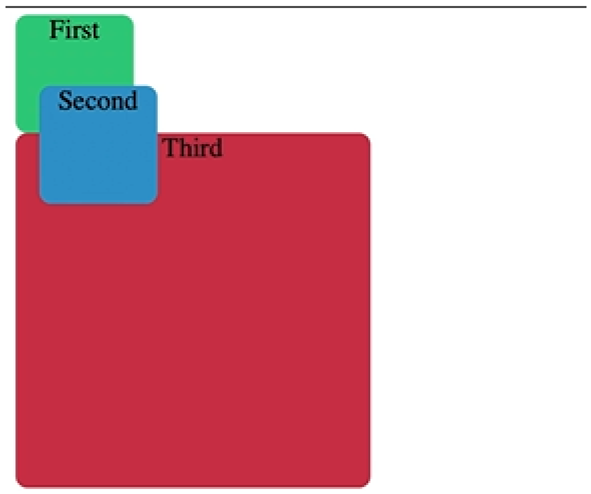
So if you use position:absolute for an element, the browser will look for the first parent container with position:relative, and the element will be positioned absolutely relative to that parent element. All the top, right, left, and bottom values will be relative to the parent container. So it doesn't stay fixed or visible once you scroll.

So that would be the difference between different position types.

The z-index property

Now, another property that is important to mention here is the z-index property. So let's try actually overlapping that Second box element with the other boxes.

```
.second  {background: hsl(200, 60%, 50%); left: 20px; top: -40px; position: absolute;}
```



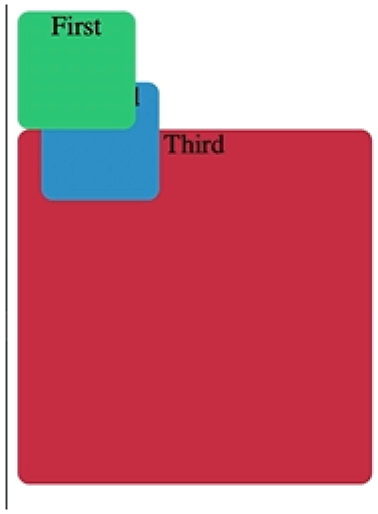
And now you see that this second box is on top of the first one. And as you probably already know, the Z-index specifies the Z axis as opposed to the X and Y axis. It defines how close the element is towards us. So by default, the order in the HTML document would define the order of which element is on top of another.

Let’s see what happens if we specify the z-index property for this first box.

```
.first {background: hsl(150, 60%, 50%); z-index: 5;}
```

It doesn't do anything. The z-index only works on positioned elements. So I do need to add position:relative.

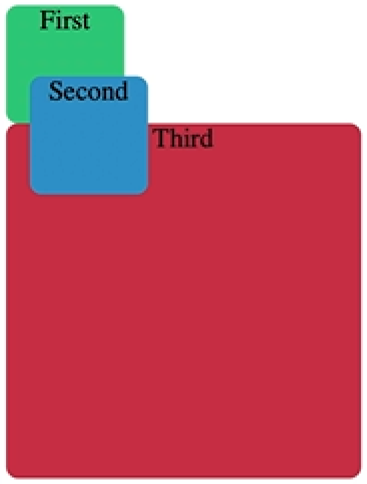
```
.first {background: hsl(150, 60%, 50%); z-index: 5; position: relative;}
```



And right now, my first div has a higher Z-index because the default is 1 - kind of. So it has a higher index value than the second div, and the div with the higher value will be on top.

But if I change the z-index value of the Second box to six, it will be back on top.

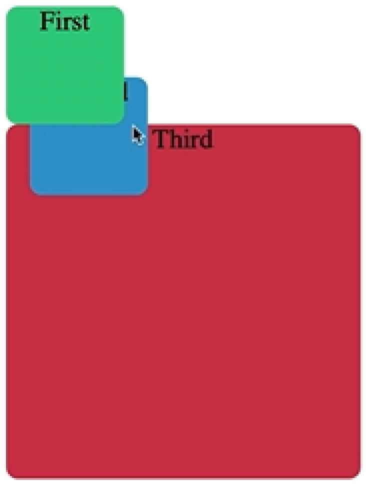
```
.second  {background: hsl(200, 60%, 50%); left: 20px; top: -40px; position: absolute;
z-index: 6;}
```



But there is a very important aspect here that we need to remember: the position of an element will be relative to the current position stack. What I mean by this is if the third div has a z-index of three, I can set 99999 for the second div and it won't move it on top of the first div with z-index 5.

```
.third  {background: hsl(350, 60%, 50%); position: relative; width: 300px; height: 300px; z-index: 3;}
```

```
.second  {background: hsl(200, 60%, 50%); left: 20px; top: -40px; position: absolute; z-index: 99999;}
```



That is because the z-index value can be only as high as the z-index value that is defined for a parent container. That is very, very important.

And I know that it is confusing, and in Divi, a column, for example, can have a different z-index property and then moving your elements outside and overlapping them can cause issues. But as long as you remember that the z-index can only be as high as the parent Z-index value and that the z-index can only be applied along with positions other than static, then you should be able to troubleshoot and position your website elements easily. So hopefully that is helpful.