CSS LAYOUT

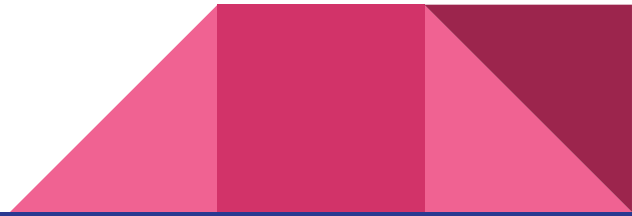
The position Property

Manesh Jhawar

According to W3Schools,

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

In other words, the position property can help you change the location of an element in the document.



The different type of position properties are:

1. Static
2. Relative
3. Absolute
4. Fixed
5. Sticky

position: static

1. **`position: static` is the default positioning for every element.**
2. If you have three statically positioned elements in your code, they will stack one on top of the next.
3. You can use the `static` value for simple, single-column layouts where each element must sit on top of the next one.

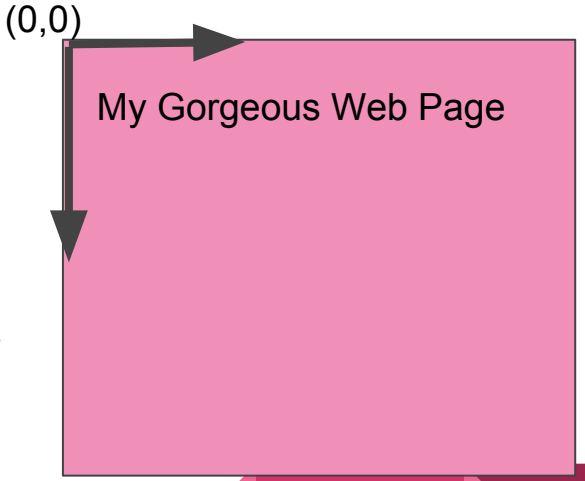
Check out an example here:

https://codepen.io/manjhawar96/pen/aYXojV

Coordinate Systems

Before we go on, we need to describe the coordinate system(s) used by a web page.

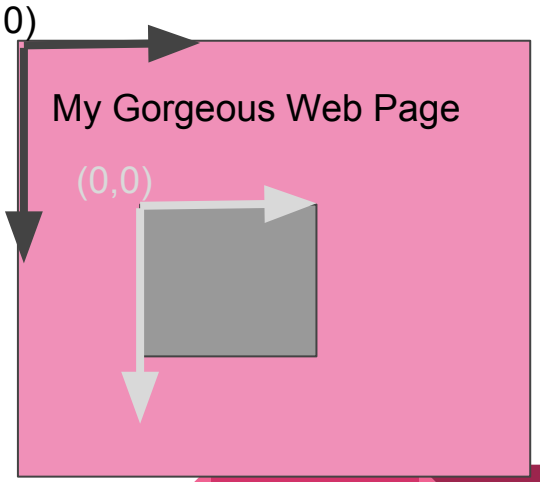
1. The (0,0) coordinate for the entire web page is in the upper left hand corner of the page. The values of the X-coordinates increase to the right, the values of the Y-coordinate increase downward.



Coordinate Systems

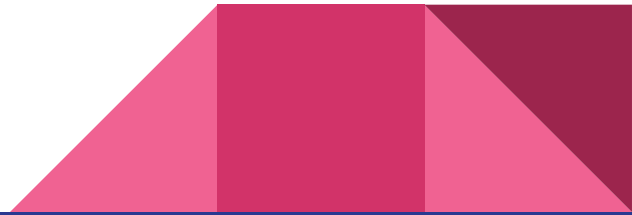
Before we go on, we need to describe the coordinate system(s) used by a web page.

2. Some positioned elements (described on the next few slides) will have coordinate systems of their own. If they do, their coordinate system starts in the upper left hand corner of that element, again with the X-coordinates increasing to the right and the Y-coordinates increasing down.



What's wrong with `static`?

1. Want to shift a box that is a static element (styled as `position: static`) in some direction? You can't, because offset properties such as `top`, `right`, `bottom`, and `left` are not available to a `static` element.
2. Static elements cannot create a coordinate system for their children either.
3. Children of a static element (Element inside the static element) will use the document's (the whole browser window) coordinate system. Recall that means (0, 0) is in the upper left hand corner of the page.



Looks like there are a lot of limitations to static elements? We have relative elements to fix all the problems. Let's look at relative positioning!

position: relative

1. Elements that are have a style of `position: relative` are considered relatively positioned. Relatively positioned elements behave just like statically positioned elements and stack the same way static elements do.
2. If an element is styled as `position: relative`, we can adjust its location based on an offset from where it **should** be positioned These offset properties are `top`, `right`, `bottom`, and `left`.

Checkout an example here: https://codepen.io/manjhawar96/pen/zWeYqj

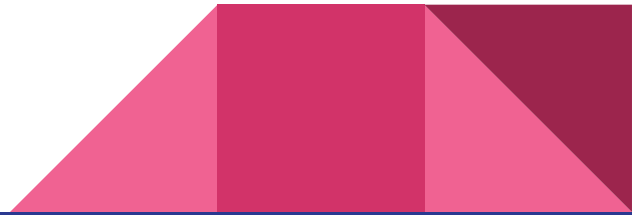
Uncomment properties in CSS to see how we can use offset properties to move elements around.

3. The `position: relative` property also create its own coordinate system for its child element. Now the child elements will measure it's offset properties from the box rather than the document (see slide 6).

Check this example where we see how relative positioning align their own child elements:

https://codepen.io/manjhawar96/pen/pLGoBV?editors=1100

We can see that the red box calculates 40px from left side of the green-box rather than the left end of the document.



position: absolute

1. An element with the property `position: absolute` is called an absolutely positioned element, and it is removed from the normal ordering of the elements on the document page. This means you can put it anywhere, and it won't affect or be affected by any other element in the flow.
2. Absolute elements are positioned relative to the nearest positioned* ancestor element. If there are no positioned ancestors, the element positions itself according to the outermost container of the document or, in many cases, the document itself.

Meaning the ancestor can **not have the property `position: static`.*

Check how position absolute works:

https://codepen.io/manjhawar96/pen/bvzNgv

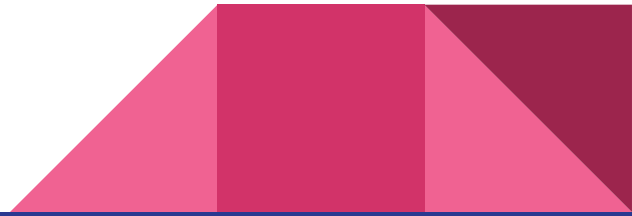
3. You can use offsets for absolute elements as well.

position: fixed

1. A fixed element shares all the properties of an absolutely positioned element but the element's parent container becomes the viewport. That means, that the fixed element does not scroll with the document. It is just there, fixed on the screen.

Check this example of a `position: fixed` property on an element:

https://codepen.io/manjhawar96/pen/wmNBLQ



position: sticky

1. Sticky positioning can be thought of as a hybrid of relative and fixed positioning. A sticky positioned element is treated as relatively positioned until it crosses a specified threshold, at which point it is treated as fixed until it reaches the boundary of its parent.

Check this example of a sticky element:

https://codepen.io/manjhawar96/pen/xWMGxg

Putting it all together

Here’s an example of a using the position properties to center an image between two containers:

https://codepen.io/manjhawar96/pen/ZxwGLW