A matter of perspective

One of the things I tend to forget when working with CSS is that you can work on three dimensions.

We'll look at perspective and perspective-origin, and how they work on the CSS.

At its simplest, the code would look like this:

```
#container {
  -webkit-perspective: 600px;
  perspective: 600px;
}

img {
  transform: rotateY(60deg);
}
```

We set the perspective on the parent element (`#container` in the example) and then use one or more transform functions in the transform property of the object we want to work with (`img` in the example).

There may be browsers that still use the `-webkit` prefix for perspective so we account for that duplicating the value of perspective on the prefixed version.

These transformations may have undesirable effects. The transform functions will change the way the image looks on the page so you need to test them and make sure the results match your needs and expectations.

The following Pen illustrates the results of the code. You can play with the values to see the different effects it has.

We can also play in the z-axis, representing the depth of a three-dimensional object. The z-axis is perpendicular to both the x-axis and the y-axis. When using the z-axis in CSS we can say the z-axis defines how close or far the object is from the person viewing it.

The example below tries to illustrate how the z-axis and y-axis rotation work together. You can experiment by editing the pen and changing the value of `rotateY`, or eliminating it altogether to see what happens.

So now that we have a basic understanding of how they work we can look at some things we can do to make our work with a 3D perspective easier.

Use the same container for images with the same perspective. If you don’t then the images may become disorienting.

If you have nested elements that you want to work with. Make sure you use `transform-style:preserve-3d;` on the intermediate containers, the parents of the elements you want to work with

Some browsers will produce jagged results when rotating images (particularly in older versions of Firefox). To fix this we need to add a transparent outline.

Working with perspective is fun if your project warrants it. But, as always, we need

to be careful on how we use it and how accessible our content is when we do.

Additional resources

- The Secret To Life, Happiness & 3D: Perspective
    - Associated Codepen
- Tips & Tricks in CSS 3D
- More Tricks and Tips For CSS 3D: Smoothing Transforms & Fixing Floated Elements