

# Using CSS Variables

If you're coming from the world of preprocessors, you must be used to using a variable by just referencing its name. With native css variables, things are a little different.

Once a variable has been defined and assigned a value, you can go ahead and use it within a property value. There's a bit of a gotcha though.

If you're coming from the world of preprocessors, you must be used to using a variable by just referencing its name.

For example:

```
$font-size: 20px

.test {
  font-size: $font-size
}
```

With native CSS variables, things are a little different. You reference a variable by using the `var()` function.

With the example above, using CSS Variables will yield this:


```
:root {
  --font-size: 20px
}

.test {
  font-size: var(--font-size)
}
```

Quite different.

```
:root {  
  --font-size: 20px  
}  
  
.test {  
  font-size: var(--font-size)  
}
```

You need to use  
the variable within a `var()` function




Once you get that out of the way, you'll begin to love CSS variables - a lot!

Another important to note is that unlike SASS (or other preprocessor) variables where you can use the variables in a lot of places, and do math like you want, you need to be careful with CSS variables. You'll mostly have them set as property values.

```
/*this is wrong*/  
.margin {  
  --side: margin-top;  
  var(--side): 20px;  
}
```

```
/* this is wrong */
```

```
.margin {  
  --side: margin-top;  
  var(--side): 20px;  
}
```



Aargh, this is so wrong.  
This isn't the same as  
`margin-top: 20px`

You also can't do math. You need the CSS `calc()` function for that. I'll discuss

examples as we proceed.

```
/*this is wrong */  
.margin {  
  --space: 20px * 2;  
  font-size: var(--space); /*not 40px*/  
}
```

If you must do math, then use the `calc()` function like so:

```
.margin {  
  --space: calc(20px * 2);  
  font-size: var(--space); /*equals 40px*/  
}
```