We discovered the DRY abbreviation in the first lesson. Now we're going to discover what DRY CSS code looks like.

Properly abstracted code is often times DRY. The opposite of DRY is WET which stands for we enjoy typing. Our goal, as software engineers, should be to rely on WET code less and less in our careers, and make our code more DRY.

However, if you make your code too DRY, it's basically nothing else but premature optimization. You're spending a lot of time, effort, and resources in perfecting every single line of your CSS. And as a side effect of this perfectionism, you're going to get less maintainable code because your CSS structure, your CSS hierarchy's, going to become so complex that it's not going to be productive anymore.

Therefore, whenever you DRY up your code, try to apply some common sense in what you would like to abstract and what you would like to keep just to be WET.

Let's see the advantages of writing DRY CSS.

The first one is better maintainability. If I wanted to summarized what better maintainability means in terms of writing CSS is that DRY code attracts more DRY code. Whenever you surround yourself with DRY code you tend to think in terms of abstractions all the time. It is a lot easier to start grouping elements than to repeating yourself and copy pasting your code over and over again. This is how you start defining a CSS hierarchy for yourself as well because you're encouraged to form abstractions.

Your code is going to be more consistent, there are gonna be less rooms for errors. Because in general, whenever you make a modification, you modify one thing in one place. DRY code also encourages better design because whenever you have an override, for instance, you will start thinking about, where does this override belong to?

Which class has the responsibility of encapsulating this override? And then you're going to see that you're going to define less and less overrides. And you're going to start including these overrides in their proper places, and in only one place.

Let see the process of DRYing out WET CSS. This process is called **abstraction**. What we do while abstracting code is that we scan our code for reoccurring pieces that have the same purpose. Whenever we see the same code written over and over again, we have to check what the purpose of the code is, and if they served the same purpose, then we create an abstraction that's meaningful.

The abstraction is created by replacing all code segments with one code segment one level higher. Alternatively if you don't want to create a hierarchy, we can simply create a utility function or a utility mixin that serves the purpose of describing an abstracted piece of functionality.