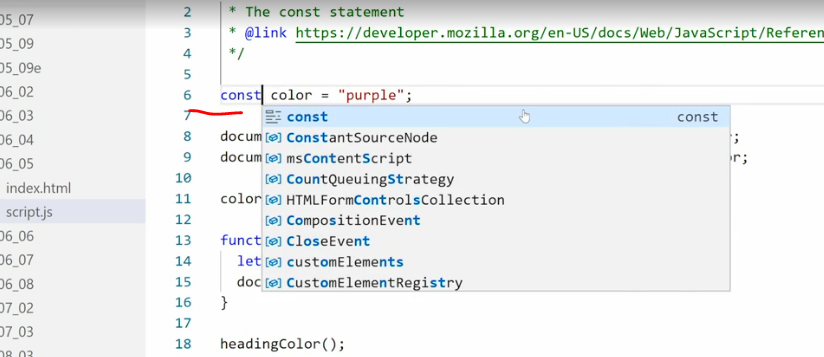
* - [Instructor] The const statement defines a block scoped constant.
* *From a scope perspective it works the same as the let statement,* but **a const is a constant meaning once it's defined, you cannot assign a new value to it.**
* Let's look at our example again to see how this works here.
* Here we have our example in the status it was when we left it in the previous movie.
* Jumping over into the code, you can see we're now using lets instead of are so that we ensure that there is block scope and everything is working properly.
* But we still have this redefinition of a value down here.
* We redefined the color value to sky blue.
* Now let's see what happens if I change the lets at the top here to a const.



* Save.
* Check in the browser.

Graphical user interface, diagram

Description automatically generated with medium confidence

* And once again we have that problem where the right-hand box turns green.
* Only this time, the title is also black.

Graphical user interface, application, website

Description automatically generated

* So what is going on? Looking at the console we get a new Uncaught TypeError assignment to constant variable at script.
* js line 11.
* What's happening is *we're trying to reassign a value inside a constant.*
* And because it's a constant, we can't do that.
* So the script stops rendering, not down here where it was stopping previously but all the way up at line 11.
* So that means the heading color function never runs.
* So ***by reassigning a value to a constant we are breaking our script.***
* Now you'll notice inside VS code I'm not getting any errors here.
* It's not telling me I'm doing anything wrong but in the browser the browser is telling me something is wrong.
* At first look, this might seem annoying, especially because you get no warning in the code editor when you accidentally tried to reassign a constant but it's actually a very helpful tool when programming.
* Assuming you follow my rule of thumb and **use a let for any changeable or mutable variable using a const for any unchangeable variable** makes a lot of sense.
* Using a const we know there is no risk of the assigned value suddenly changing or being changed by accident.
* And we know if we try to assign a new value to a constant the browser will flag the error and stop rendering immediately.
* The end result is cleaner code and less errors.
* One note, when I say this when I say we can't reassign values to a const that's exactly what I mean.
* We can't put new stuff in the const box.
* **That doesn't mean we can't change the status or properties or other features of what the constant holds.**
* As we've seen earlier, ***you can still change the properties of an object inside a const.***
* ***You can also change the entries in an array inside a const, you just can't reassign the whole constant to replace the object with another object or the array with another array or an object with an array.***
* So the key takeaway here is for **any data that should be protected from accidentally being overwritten, like an object, or an array or a function, use a const.**