0:00 Okay, let's go ahead and practice with variables constants and- scope. We're gonna- I'm gonna go ahead and use this same HTML that we used, um, earlier.

0:13 The JavaScript really won't have anything to do with this HTML, but it's just a place for us to go ahead and look.

0:21 At some of the variables just used in the console. So I'm gonna go ahead and add a JavaScript file. I'm gonna call it variables in constants.js.

0:36 And we're going to- to externally link that up with our script tag. And because I'm doing it in the head, I do want to use defer.

0:45 We'll use the SRC attribute to reference where that is and what the name of this file is. Okay. And then we will get the ending script tag and no content because we are just referencing it externally.

1:04 Okay, now we can go ahead and use, some variables and constants in here. Let's start with a const keyword or a constant, and sometimes you will see very, erm, constants all capitalized.

1:22 So we'll capitalize it at this time, and, hi. A is always 3.14, so that's a great use of a constant value will never change.

1:33 Let's set up a variable of radius, so we're going to be getting in the area of a circle, and radius definitely can change.

1:41 So we'll from circle to circle, depending on the size. So we'll start with a radius of 3. Now the area of a circle is r squared times pi, so we're going to set up another variable called.

1:57 And we're going to use, now, our two variables in our calculations. Radius times radius times pi.

2:11 Notice we've got semi-colons at the end of- each of our statements, which is what you should be doing in JavaScript.

2:18 And we can now, we're, we assign these first two constants, and this variable, 3.14 and 3. And now we're assigning a new variable and a calculation, so the result.

2:29 Of this calculation will then be assigned to area variable. If we want to go ahead and take a look at what that looks like, we'll just put it in the console with console.log, which is a object with a method reference.

2:47 And we'll learn a little more about that later, but basically it's just saying we're going to put this on our console, the result of area, or what the value of area holds.

2:58 So if we go back to our HTML and the look at that in life server, and we look at our depth tools, and we go to the console tab, we will see that there is an error.

3:15 and and so I'm going to check my spelling, rables, constants, there we go. there should be an S right there.

3:31 there we go. alright. Right. So, it's console logging what area, so after they're 3 times 3 times 3.4, we get the 28.6, line 6 is where that console logged, and so we can see that.

3:46 Now, later in code, we might want to change the value of, radius. You don't have to set it up again with the keyword let, because it's already been set up, but we might want to reassign it at bigger radius.

3:58 Maybe 20. And then, when we rerun the calculation and the console, later, we'll see that we get a bigger number, because of our bigger circle, or our bigger radius.

4:13 So, that's a good example of kind of using that constants and variables and some calculations.

0:00

Another thing to be aware of is variables and constants are case sensitive, so if I were to put a lower case PI, it would not recognize that, so be careful with case sensitivity with variables and constants, as well as if I went ahead and tried to reassign my constant and eight different.

0:24

Value, I would also get an error because you cannot reassign a new value to a constant, that's the purpose of a constant, that the value would not change.

0:34

Another thing to be aware of as well I I later down in code I refer to it as pi, for some reason I misspell it, and reassign something to it.

0:48

It's going to assume that that's a brand new variable, even though I didn't use const or let, it's going to think I've created a new variable.

0:57

So if you misspell a constant or variable later, it might think it's a brand new variable. Well.

Alright, let's take a look.

4:22 Look, at some interesting things that might happen with, um, variable constants. So, I'm in the ponder, and I should have grabbed that code, but let's grab this code here.

4:34 And this is setting up two more constants. This constant is, It's called 1, and it's being assigned to number 1.

4:43 This constant is 2, and it's being assigned a string or a text or character 2. And we know that because it's got the quotes around it.

4:53 So, kind of different, of data types going on there. But we didn't have to tell it what data type it was when we set those two up.

5:00 But if we go ahead and do some math and assign it to a new variable called result, and we say 1 times 2, Then we can see that, console log this one.

5:15 Okay. And it will go ahead and do that multiplication one times two. Now this was a string. So I'm gonna just put a little comment here.

5:33 This is called type coercion. And that means it's, JavaScript is saying, well, the, they're trying to do multiplication on this string.

5:44 So I'm gonna assume that they want it to there. And go ahead and let that happen. So. It can be flexible to a point that sometimes is not, it's not as strict as a lot of the other programming languages.

5:58 It will kind of try to assume what's going on there and do what it can't. However, let's take a look at this.

6:05 If we go ahead and. Let's say let's have the result be one plus two. So here's an exception to this whole rule.

6:17 And we go ahead and console log our new result out at this point. Then we see it comes up with twelve, and that is not one plus two.

6:28 You might have expected it to be three, but it's showing a one concatenated with a two. So it's just kind of putting those two together.

6:37 So anytime you have a string. With the plus, it's going to assume that is concatenation, meaning just shove those two together, put the one right next to the two.

6:48 And that's why it looks like there's a twelve there. So in this case, with the plus sign, it's not using it as an- additional or some operator, it is using it as a concatenation.

6:59 So you would have to go in and say, I really want to see this one as a number. You could use something like this, oops.

7:07 Uhm, or Oh, goodness. And now, it will see it as a number, and you can go ahead and put the three in there.

7:20 This will happen a lot with, um, any numbers that the user puts into an input box. We'll see that later when we talk.

7:27 About forms. They always come in the strings. So a lot of the time you will have to convert those over to numbers if you want to do math on them.

7:34 Okay, so that's just with the plus because they'll see it as concatenation and set. So a little few little different things that might happen as you're looking at that.

7:43 Let's go ahead. And look at the scope. So let's copy this code right here. And paste that in. Okay, so let's look at scope now.

7:55 Just putting little comments in there so we know what we're looking at here. Alright, so we have a global variable called course.

8:03 We're assigning it this string, CSE-131. It is global because it is at the very outside level. It's not inside any conditional.

8:13 It's not inside of, uh. Function, um, so it's kind of just at the outside of everything. So, no matter where a course is referenced, you should be able to see it.

8:24 Now we have a block of code here. And we can see that it's got curly braces, whatever. What's happening on the inside of these curly braces, if we declare a variable inside of here, it's only going to be recognized inside.

8:40 If we try to reference it outside of those curly braces, it's not going to work. So. We can, when we set up the, um, inside this if clause, we're going to set up student in here, therefore course, which is global, can be seen, and student can be seen.

8:59 Again, course is definitely going to be able to be seen here, inside or outside. Inside of the block, but you cannot reference student, which was declared inside of the block.

9:10 It's not going to work. So let's go ahead and say that and just take a look at what that looks like in the console.

9:16 And we see the first three work just fine. But this last one is not working because we cannot see student when it's inside of a block like that.

9:26 Okay, so a little bit about block, um, and global scope and variables and constants.