# Transcript: Indentation Revisited

>> Let's take another look at indentation in programs. Remember that indentation in Java is, is for the human reader, it's ignored by the computer. When you compile your Java program, all this extra space is just ignored. However, look at the following if statement. Here we've got, if total greater than MAX, then we want to print out Error!, and we want to increment the errorCount++.

We'll recall that an if statement is defined for a single statement or a block of statements. So the body of the if can be one statement or a block of statements. A block meaning open brace statements, close brace, and in this case, we don't have braces. So the indentation sort of suggests the errorCount++ is inside the body of the if, but, in fact, it's not.

In fact, errorCount++ is going to get executed, errorCount++ will get incremented, whether or not total is, greater than MAX. If we had, generated this with the control structure diagram, we would see the proper indentation. And the error becomes a little more obvious, so lesson here is, braces are a good thing.

This is why we've used them from the beginning. And this braces is a block statement, and essentially you just have a group of statements delimited by braces. And it can be used anywhere in the Java syntax where a single statement is used. So to correct the, the if statement above, we would simply just add braces there.

And then we would get the, correct implementation. So our coding standard, which is supported by Checkstyle requires blocks, or braces in if statements, and, and this is the reason why. Now let's consider nested if statements. If you've got an if or else if, and it contains other if or else if statements, then we have what's called nested if statements.

Now one little caveat, if you're not using braces, and we are using braces, so this shouldn't be an issue. But, an else clause is matched with the last unmatched if, no matter what the indentation implies, unless you do use these braces. So using braces allows you to specify which else the if goes with, and, and keeps you out of trouble, so always use braces here.

Let's look at a couple of examples here, Taxes.java and Taxes, TaxesIfWithoutBraces.java there. So, here, here's a, now we're just looking at a snippet of the of the if, if else, so this is a nested if statement. And notice we have braces, we've got if and a brace here, and it ends down here, in fact, you can see it's lined up.

And it's got an else, inside here, with, with braces, in fact, we've got braces everywhere we should have them here. So, in the event that I remove an else, I'm gonna remove this else block right here, cuz then I'm gonna comment it out. So it's not there, and it looks like the indentation here, if I just remove the, the CSD.

It looks like that this else at line 43 goes with this if up here, but in reality, because of no, actually, it will. Because we've used braces here, it was a good thing we used braces. So when when we generate the CSD, which, takes into account this commented, commented out code.

Notice that, that it stays as it is, so we've used braces here, that's a good thing. While I'm at it, I'm going to uncomment this. Let's just put this back, Uncomment, OK, so this is where we want to be. Now, let's look at Taxes, the IfWithoutBraces, and so, here, it's the same code exactly, but I've just removed the braces.

And so, now, if I comment out the if, here's the if right here. Just as we did before, it's commented out, and I'm gonna remove the the CSD, and it looks like this else is lined up with this if here. But because there's no braces, the way the computer is gonna interpret this, the compiler will interpret it.

Is if this else goes with this enter if statement here, if I generate the CSD, it actually shows that. Notice it moved over, and it goes now with the enter if, which changes the meaning of the code. So, bottom line is here, you always want to use braces with your ifs, and whiles, it just makes, good sense to do that.