Transcript: String Class

>> Let's look at a few more details of the string class. Recall that when we create a string, we have two ways to do that. One is using the new operator. Here we're saying title equals a new string using classes. We assume that title has already been declared to be of type string.

And then the second way is just using a string literal directly, just title equals using classes. Now each string literal that you use in your program, actually represents the string object and one gets created under the hood so to speak. So we've got string objects, so that we don't know about necessarily.

In other words, we haven't actually set them to a variable, but they exist each time we use a string literal. Now, for most reference types, the new operators is the one you're gonna use to call a constructor to create an object. String is this exception here. Now, one important concept about the string is, that string objects are immutable.

That means once you create a string, it can't be changed in memory. Here's an example, we have the replace method. And when we call the replace method on a string, it actually returns an entire new string object. The target string, the one that the method was called on is actually unchanged.

So here, we've got an example, String title2 = title.replace, lowercase s with uppercase S. Now nothing actually happens to title here, but when we call a replace method, we get back a brand new string in which the original string has the lowercase ss replaced with uppercase Ss. And we have assigned that new string to title2, may not always want to create new string reference.

In fact, you may just want to have one string reference and, and make it appear that you've replaced the lowercase ss with uppercase S. So here's an example here, here we've got title.replace lowercase s with uppercase S semicolon. Now this doesn't actually do anything, it, it calls the replace method on title.

But as we know, it actually doesn't change title, it gives back a new string. And we didn't do anything with that new string, so we probably meant to do this. We meant to say title = title.replace, lowercase s with uppercase S. In that case, we get back the new string and then we assign it to title.

Well, the old string that title used to point to now isn't pointed to by title anymore, it's pointing at the new string. So effectively, it looks like we've replaced those characters and title. Now let's revisit the idea of string indexes. Characters in the string, a string is made up of of an array of characters.

They begin at 0, the index starts at 0 there. And so, and Hello we've got the character H as we see in the figure here starting at 0. So we've got H-E-L-L-O and there is five characters there and they're indexed from 0 to 4 there. And we see that H is it index 0 and the little o is at index 4.

Here's another string Hi There. And notice it has a space in it and the space actually takes up a character as well, just like you would think. If we want to get at a particular character in the string, we could use the charAt method and we'd give it the index.

If we wanted to get, get at a sort of a piece of the string, as in more than one character we could use the substring method. And there we would give it the indexes, the start and stop indexes, or in some cases just the start index and retrieve that part of the string.

So, here's charAT, here's a couple of examples. This is, here we've got a string, myString Hello, and if we say myString.charAT (0), that's gonna return that H. And to use a substring method, again, there is two ways to call that, one is with two indices, and the other is with a single index.

Here we want to take this string Hi There, and we wanna find the first half of the string and the second half of the string. This looks slightly complicated, but it's not. So we've got two indices, 0 followed by, so we wanna start at 0 and we wanna go up to string.lenght () / 2.

So in other words, we wanna get the first half the string. This is integer arithmetic, so we 'll lose the fraction there and just use the integer that's close to the half way point. To get the second half of the string, we can call substring. And here we can just give it a single indice.

In this case, myString.length() / 2, we want to start there and go all the way to the end of the string. When you want to go all the way to the end of the string, you don't actually have to give it the second indice. OK, so let's, let's look at a couple of examples.

In example one, these are all in that examples folder. Here we're going to set title to using classes. We'll print that out and then we're gonna get the count of title.length, so we're just calling some of the methods we looked at. And here we're creating some new strings and we're gonna say, title.toUpperCase, to.LowerCase, replace('s', "S").

And then here's one a little bit different, we're gonna say title two lowercase, and then replace lowercase s with the uppercase S and so on, and then we'll just sorta print those out. So let, let's take a look at that. I'm gonna run this in debug mode, so that we can see this going on.

So I am going to step, so here's title over here, we have using classes and there is the count, we see count over here 13 and now here is to upper case. There's a little more room there. And next we've got toLowerCase, And this is after replacing lowercase s with uppercase S here.

And this last one, we're gonna drive string to all LowerCase. Notice a title has an uppercase U and an uppercase C, we're going to get the lower case version of that, and then replace the lower case s with upper case S in that. And we see we've done that, and so this is like the one earlier except now it started off all lowercase before we replaced those lowercase ss, and so on.

And then this other these other few lines just sort of print things out at the bottom and we see that. Now if you want to run this in the canvas there is a canvas that comes with the example, of course you can always make your own. Here we've got Let me make that a little bit larger there, so I'm just gonns step through, here's step, there's title, and here we can see all the indices and count.

And next, we're gonna get title to all uppercase. There it is, this is essentially what we saw over in the debug tab. And there is lowercase here we replaced some characters, replaced the lowercase version of it, and so on. Now, this would be useful if you had a really long string, since over in the debug tab, you only see the first few characters of it.

In this case we had a short string and we could see it all. So I'm gonna end this and then let's look at string example two. In this one, we're going to actually let the user enter a string, and then we will say what was

entered. Were gonna trim it by calling the trim method and then we're gonna get charAt 0 and say the first character is.

And we'll get the last character, notice how we're getting the last character. It's the charAt, we want it's input.charAt, and the index we want is input.legnth()- 1. So, so this right here is, is the entire the single index where we're going to get the character. And then next we're going to ask if it contains a space and notice this is going to return Boolean, input.contains(" ").

If it does contains a space, it will come back true, and we will say that it contains at least one space. And, and so on, and then we print all this stuff out. So let's, let's run this. I'm gonna run it in debug mode first, so let me I'm gonna set a breakpoint right here.

And we'll just click the little debug button. And it runs down here and it's waiting for my input. I'm gonna key in automobiles, automobiles, automobiles there we go. All right and we can see it over here on the side here as, as input. And now next we're going to say what the string, string was and next we're gonna trim it and I didn't key in any spaces here.

So we're not gonna get to see anything trim, but had I had a space in front of automobiles it would get trimmed off there. And now we are going to get the character, that character at 0 and print it out at the bottom there, it's an A as you see down here in the run IO tab.

And the last character, it's at input .length() -1 we see that should be an S, and we print that out, there it is, an S. And it doesn't contain any spaces, so this is gonna be false. And as you know, down here we can actually, in interactions you'll do this at some point, I've switched from Run IO to Interactions, we can actually copy this down below here.

And execute it in interactions and it should come back false, because automobiles doesn't contain any space. So back to Run IO,when I step, go back to Run IO this should be false. And so we're not going to print that it has spaces, we're gonna say it was entered with no spaces, so we print that out and so on.

And then we're gonna replace some vowels, you can see those getting replaced over here. So the As are gonna become asterisks, and we see the A went away, and Es Is, Os, and Us. Automobiles doesn't quite look the same anymore, does it? And then we're gonna print that out.

Now, and then finally here we're going to do that first half and last half of the string there that we talked about, and so let's just go ahead and print that. First half looks like all Tom, and the second half is, is the rest of it there, and that's what we've got.

We also have a canvas for this, if it was a long string you would see it in the canvas a little better perhaps. And here it is, in fact I'll just, I'm gonna play it in the canvas as enter a string. I'm gonna say, automobiles are fun to drive.

So that's a pretty long string there. And so there is our long string and you can see as we go through here, we're about to start replacing things. It's stepping along here in the program. And here is the first half and the second half. We may have to make that a little larger to see, let's see what we got here.

There is the first half and the second half. We can click this of course and look at the values replaced and so on as we drive through or scroll through I should say. All right, I hope this has been a little more in-depth look at strings. There's, there's tons of tons of methods on strings if you look at the Java API for string, you'll see all the methods of return types and, and how to use them.