Transcript: Autoboxing and Methods

>> Now let's take a look at the example, AutoBoxingAndMethodsExamples.java. You can find this in the examples folder for this set of notes. And this will demonstrate the differences between a primitive event and the object integer, the wrapper class. And we'll also look at some of the wrapper class methods.

So we're going to set a break point at the first executable statement and then launch the debugger. And we'll take a look at all the variables that show up over here to see the differences between primitives and the wrapper class. So the first three lines there, lines 15 through 17, they're gonna create our regular primitive int.

And we're initializing them all to 1. Now the next three lines there, 19 line 19 to 21. Here we're creating integer objects. And notice, first off we're setting n1, to 1. Well, 1 is actually a primitive literal, and so it's gonna get boxed up into an object. And we see that over in the debug tab, it's object 140.

And it's a java.line.integer, that's its type. And we'll go ahead and assign create these next two, n2 and n3. And notice they're all equal to 1 over there. Now, I'm gonna set i1, the primitive i1 to 3. And we see it over in the debug tab, it's red.

And now I'm gonna set the object n1 to 3 as well. And, again, this 3 gets boxed up into a new object. Notice currently n1 is object 140. When I step, it's now object 150 but it has a value of 3. So the 3 got boxed up into this object.

And and we see it's got the right value. Now next I'm gonna set i2 to a new integer 2. So new Integer(2) creates an integer object with a value of 2. But it gets unboxed automatically and i2 gets set to that. We see over the debug tab i2 is now 2.

I'll do the same thing for n2. This is new Integer(2) but in this case, we're actually assigning it to something of type integer. So no surprise there. Now in, in the next line here, line 29, we're going to do some arithmetic that involves a primitive and an object integer.

And, that works out fine. Notice that, i1 is equal to, 3. And n2 is equal to 2. So this addition here is gonna get assigned n3 and that will be 5, and we see that there. And now we're doing the same arithmetic. I, I've sort of

reversed the variables, now we're taking n1 and i2 they have the values 3 and 2 as well and, and we get that assignment.

So, now let's take a look at some of some of the methods. In this example here, we're going to find the digits to the left and right of the decimal point and pi. So I'm gonna call a math.pi to set that up. So now, in the debug tab, I've got x = 3.14 and change.

And now I'm going to assign that to y which a double object. And notice it's the same value over here but it is an object, but it's has that same value of pi. And now if I wanted to get a string out of these, since y is an object, I can call y.tostring.

It's got a tostring, method there. And notice this gets the string for pi. And, we're showing part of it here. And, and another way to get that would be to call the, the double wrapper class of the static, method tostring. And, and send it the actual value and this could be a double object or a double primitive.

And that also gets you the a string of it as we see over in the debug tab. Now we want to find the index of the decimal point, or period. And so we're gonna s.indexOf the decimal point and set that to p, p is just an int. And we see that comes back 1.

And this was on s and we can see here that 3 is in the 0 position and the decimal point is in the 1 position just like we would expect. And now we wanna print the digits to the left of the decimal point. And we're gonna do that with substring.

We're gonna begin at 0 and go up to p, p happens to be 1 so we're just gonna get the first one. Then to get the digits to the right of the decimal point, we're gonna start at p + 1 and go to the end of the string.

So I'm printing these down in the, runio tab. You can see digits to the left of, the decimal point in pi, just three. And to the right we get a bunch more. In fact the rest of it there. So, here's, here's some other methods we can call. This is, not methods, these are actually constants we're going to get at.

So we're going to get at Integer.MAX_VALUE, Integer.MIN_VALUE, Double.MAX_VALUE, and Double.MIN_VALUE. These are available to us. So I'm just going to step, step, step. And you see them coming out at the bottom there. And you see their values. Notice that the, maximum double is 1.79 etc., and it's an e to the 308.

So this, that Mantissa times 10 to the 308, big number. And the, smallest magnitude doubled is 4.9 times 10 to the -24. If you wanted an actual negative value, you could multiply this by by -1 and get that. All right, one other example and this is an important one.

You'll use this a lot. Here we're going to take strings that are digits and convert them to numbers. So S3 we're we're setting equal to 1 0. The string 1 0. And S4 we're setting to a 4.1, you see them over here in the debug tab with quotes around them so these are strings.

If we want to do arithmetic with these, then we could convert them to their integer values. And so, integer.parseint, and you send it a string as long as it's just digits, this will work. And so I'll step here and we'll see over the debug tab that j is equal to 10 and int.

S3 was that string int, but now we've got the regular int we can do arithmetic with. Same thing for the double. So we're gonna say double.parsedouble. Double is the name of the class. Parse double is the, method. And we're, calling that on s4, which is 4.1. So this can be any str-, a string that contains any, sequence of digits.

And it can have a, begin with a minus sign, of course. It can have a decimal point in it. It just has to be a valid v double looking thing that we can get a double out of. So I'm going to step here and we see our 4.1 over here is now actually a double instead of a string like it was up above there.

And so now we can actually do do arithmetic. So we can say double product = j times z and and then print that out. Okay now just one more thing before we finish here. We've had some these integer objects, and we haven't unfolded them. We've unfolded our own objects.

Let's unfold this one. And we'll see it's got a lot of stuff. In fact, it's carrying MIN_VALUE, MAX_VALUE, these are the digits in can have, and so on. Down here is its value. That's one of its fields there. But by and large we don't need to worry about all this, but just be aware that since this is an object, it does have feels.

I'm gonna fold it right back up. So, this has been a sort of a a presentation or a demonstration if you will of of primitive ints versus object integers. The the integer wrapper class, if you will. And also, we call some methods. These last two down here, Integer.parseInt and Double.parseDouble, you'll use, pretty often.

Because recall, keyboard input comes in as a string. And also if you're reading in a file, usually that's text. And so, this need eventually get converted to ways. And this is the a good way to do it.