Transcript: Formatting Output

>> Now let's consider formatting output. You may want to format values in certain ways so they can be presented properly for a particular application. Consider the raw value shown here is a double. We may wanna print that with just three significant digits to the right of the decimal place.

Or we should, we could also say three decimal places. Or we might have the value 1.08, and we want to print that as, as a currency. As in $1.08. Well, Java has two classes, NumberFormat and DecimalFormat. The NumberFormat, provides values in currency and percentage. And the DecimalFormat class, does all that, does both, percentages and currency but does a lot more.

It's a lot more flexible. And it's based on a pattern. Both of those are in java.text and we need to import that to get at them. So let's say a few words about NumberFormat. It has static methods, it returns a formatter object for currency. So getCurrencyInstance and also getPercentInstance.

And then you call the format, method on those objects that you got back, to get the appropriate format. There's an example of PriceChange.java in the examples folder. And I invite you to take a look at those, and see, and see how that works. We're gonna concentrate on the decimal format, since it does everything that number format will do, plus more.

So, let's look at the DecimalFormat. It can be used for formatting both floating point and integer values in, in some useful ways. It's got a constructor in, in the sort of traditional sense this is one reason I like it better than the NumberFormat class. And it just takes a string that's a pattern for formatting the number.

And once you get the, the DecimalFormat object, you can call its format method and then it returns a string representing that value that you're formatting. Let's take a look at an example. Suppose that we wanna round a double value to three, three significant decimal places using the DecimalFormat object.

Well, we could do that with the pattern shown there, hash period hash hash hash. That says we essentially want three decimal places of precision, no matter how how many there are in the number in memory. So the way we would do that, we would import java.text.DecimalFormat. And then down in the code, we would create that DecimalFormat object.

So DecimalFormat df, let's say, you could call it anything. Equals a new DecimalFormat, and we give it that pattern that we want. And then suppose we had a value, a double, 4.123456789. And we want to print that with just these three decimal places, if you will. So we could, we could do that with the print line, we could say print, df.format, and the value.

So notice the the object there is df, that was the decimal format object, and we're calling the, its format method and we're sending it the value. And for output, we would get 4.123. Notice that digit following 3 was 4 and it, it got dropped, rounded down, had it been above 5, it would have actually rounded up.

So let's look at some patterns. Here's a pattern, for a large amount and, and notice we, we're gonna want commas separating the, digits on the left. So we've got, hash comma and then three digits period. And then this is gonna be six significant decimal places here. If we printed out the number 12.0, it would print out as just 12, we wouldn't get the point 0 because it's not significant.

And if we had the number123.456, it would actually print the way it is it wouldn't print anything beyond the 6, because again, they're not significant. We would just see the numbers required there. Now, if we wanted to print 12.0 with a .0, we could add in a 0 after the decimal place.

And that says, always print this digit, no matter what it is, and then the these others are optional. If they're there, we wanna print those, other, otherwise we don't. So this would print 0 as 0.0 and would print 12.0 as 12.0. Again, without the 0, you'd just get 12.

Here's a pattern to use for money. Here we've got a leading dollar sign and then we wanna group our digits and, and in groups of three there, separated by a comma. We always want one digit, let's suppose, and then we've got a decimal place and, and two zero, two zeros here.

The two zeros indicate that we always wanna see two digits there. So, if we have the number 10.1, we would actually wanna see 10.10, as in $10 and 10 cents. So, example here 123456 a decimal place, 6789, is gonna be printed as $12,345.68. If we wanted to print negative numbers and parentheses, we could give it two patterns.

And all this is one long string. It's the string above that we had for, for dollars. And then this one, we've repeated it surrounded by parentheses. And so this would print a large amount with commas, as you see. But negative values would be returned in parentheses. And we'll see an example of that in just a moment.

So, this last example is for percentages. Here we've got hash decimal point hash hash percent. So this says we want two decimal places, up to two decimal places and a percent sign. So here's our number. If our number was 0.123456, this would be returned as 12.35%. Notice the 5 there comes from rounding.

We had 123456, so the 56 rounds up to 5 and we get 12.35. So next, we'll take a look at some program examples that use these.