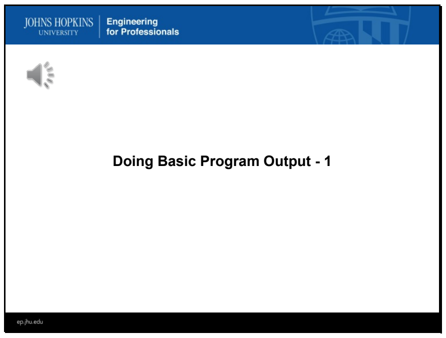
1



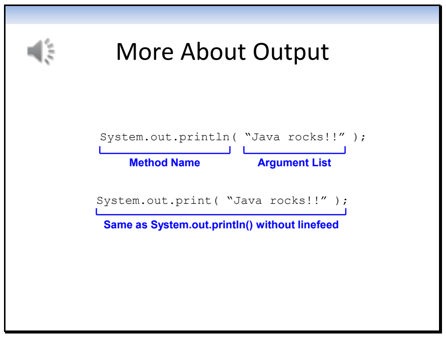
JOHNS HOPKINS UNIVERSITY | Engineering for Professionals

Doing Basic Program Output - 1

ep.jhu.edu

In this lecture you will learn how to have your programs perform some basic output functions.

2



More About Output

```
System.out.println( "Java rocks!!" );
```

$\underbrace{\texttt{System.out.println(}}_{\text{Method Name}}\ \underbrace{\texttt{"Java rocks!!"}}_{\text{Argument List}}$

```
System.out.print( "Java rocks!!" );
```

Same as System.out.println() without linefeed

I have been using the built-in System.out.println() method to have my programs perform some simple output functions, and I want to discuss this function a bit more so that you can do more things with your program output.

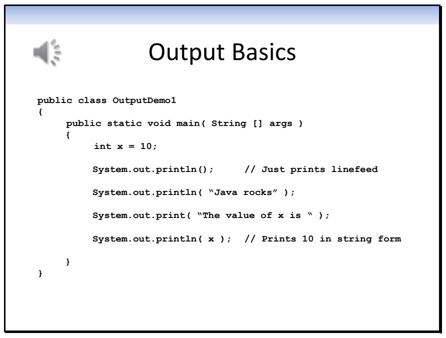
As you know by know, the name of this method is System.out.println(), and you've already seen that if you put a text string inside the parenthesis of this method, the string will be output to your display device…minus the quotes…followed by a linefeed.

Technically, what goes inside the parentheses is called the method's argument list. The argument list can be almost anything you want it to be…as long as the System.out.println() method can make sense of it.

Basically, this method takes what's in its argument list, converts it to a string of text, and displays it on the standard output device (which is your video display) followed by a linefeed.

There is another built-in method that works the same way as System.out.println(). Its name is System.out.print() and it does everything that System.out.println() does except it does not generate a linefeed.

3



Output Basics

```
public class OutputDemo1
{
     public static void main( String [] args )
     {
          int x = 10;

          System.out.println();     // Just prints linefeed

          System.out.println( "Java rocks" );

          System.out.print( "The value of x is " );

          System.out.println( x );  // Prints 10 in string form

     }
}
```

This program demonstrates a few things you can do with System.out.println() and System.out.print().

If you don't put anything in the System.out.println() argument list, it simply generates a linefeed…also called a newline.

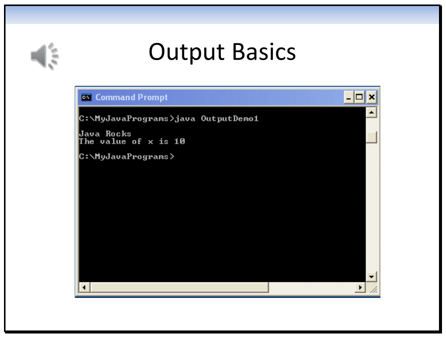
You already know what the second call to System.out.println() will do…it will display the string "Java rocks" and then generate a linefeed.

The call to System.out.print() will display the string "The value of x is ", but won't generate a linefeed, so whatever is printed next will be displayed to the right of that string.

When you call System.out.println() and put the name of a variable in its argument list, it will convert the value of the variable to a string and display it. In this case it will print a 10 in string form.

Okay, let's see this program in action.

4



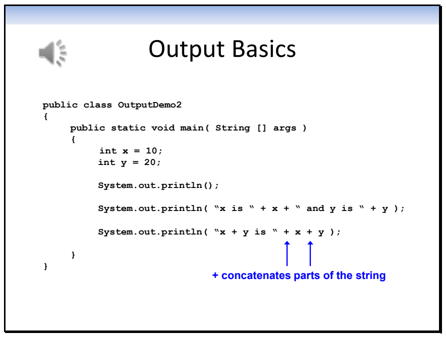
Output Basics

```
C:\MyJavaPrograms>java OutputDemo1

Java Rocks
The value of x is 10

C:\MyJavaPrograms>
```

I've already compiled the program, so I'll just run it.

Note that the value of x, which was 10, was printed on the same output line as the string "the value of x is " … this is because I used the System.out.print() method to print the string, and this method does not print a linefeed.

5



Output Basics

```
public class OutputDemo2
{
     public static void main( String [] args )
     {
          int x = 10;
          int y = 20;

          System.out.println();

          System.out.println( "x is " + x + " and y is " + y );

          System.out.println( "x + y is " + x + y );

     }
}
```

+ concatenates parts of the string

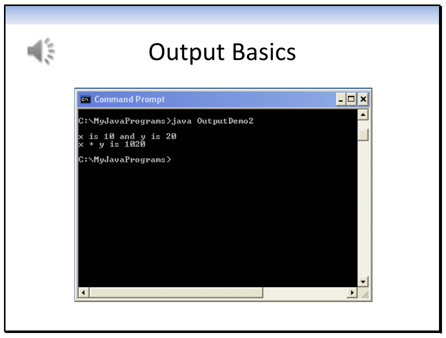
This program demonstrates a few more things you can do with the System.out.println() method.

It is important to understand that whatever you specify in this method's argument list will be converted to a character string before the output is displayed. So, you can insert literal character strings like "x is". The quotes around the string specify that it's a string. You can also specify a variable name…and the method will convert it to a string, as we have seen already.

You can also specify an expression that has several parts and that can be converted to a character string. To do this, a plus symbol (+) is used. The plus symbol causes the parts of the argument list to be concatenated…or strung together…into a single string.

I'd like you to pause this lecture and write down what you think the output of this program will be. Then, continue the lecture to see the actual output.

6



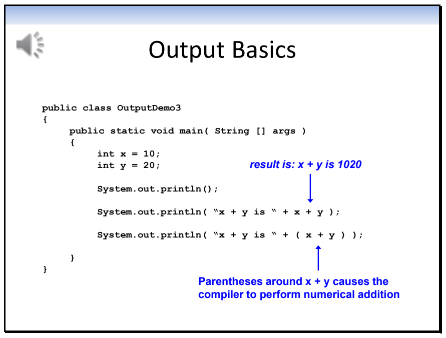
Output Basics

```
C:\MyJavaPrograms>java OutputDemo2

x is 10 and y is 20
x + y is 1020

C:\MyJavaPrograms>
```

If you thought the last System.out.println() was going to display "x + y is 30" then you're probably a bit surprised at what you've just seen.

Let's talk about what really happened in this program.

7



Output Basics

```
public class OutputDemo3
{
     public static void main( String [] args )
     {
          int x = 10;
          int y = 20;

          System.out.println();

          System.out.println( "x + y is " + x + y );

          System.out.println( "x + y is " + ( x + y ) );

     }
}
```

result is: x + y is 1020

Parentheses around x + y causes the compiler to perform numerical addition

In the last example, if you thought the last System.out.println() was going to display “x + y is 30” then you were probably a bit surprised.

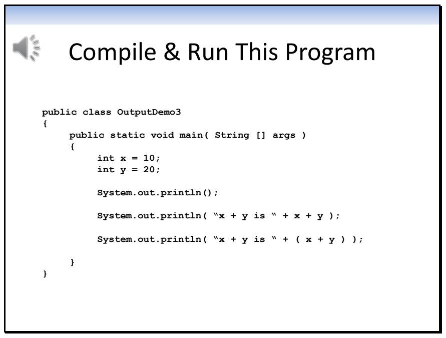
That last program did exactly what it was told: it took the string “x + y is” and concatenated, or added, the string form of x’s value…which is simply one and zero…and then added the string value of y’s value…which is two and zero. The result was therefore the string “x + y is 1020”.

What if we wanted to display the sum of x and y? How would we do that?

A simple way to to that is to put a set of parentheses around the “x + y” in the method’s argument list…like this.

The parentheses cause the Java compiler to treat the “x + y” expression as a numerical addition expression. In this case, the numeric values of x and y will be added first and the result will be concatenated to the argument list string.

8

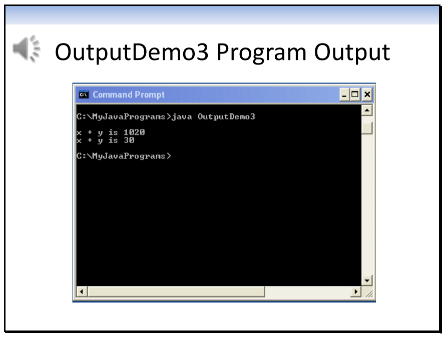


Compile & Run This Program

```
public class OutputDemo3
{
     public static void main( String [] args )
     {
          int x = 10;
          int y = 20;

          System.out.println();

          System.out.println( "x + y is " + x + y );

          System.out.println( "x + y is " + ( x + y ) );

     }
}
```

I’d like you to try this for yourself, so please pause this lecture now, then compile and run this program. When you’re done, come back and continue the lecture.

9



OutputDemo3 Program Output

```
C:\MyJavaPrograms>java OutputDemo3

x + y is 1020
x + y is 30

C:\MyJavaPrograms>
```

Your program output should look like this. If it doesn’t, please check your source code against the code on the prior and make any required changes so that you get the correct output.