Java Formatting Output Tutorial

Contents

Time required: 30 minutes

Online Tutorial

Go through the following tutorial.

- Java String Format

Java String Format Specifiers

Format Specifier	Data Type	Output
%a	floating point (except BigDecimal)	Returns Hex output of floating point number.
%b	Any type	"true" if non-null, "false" if null
%c	character	Unicode character
%d	integer (incl. byte, short, int, long, bigint)	Decimal Integer
%e	floating point	decimal number in scientific notation
%f	floating point	decimal number
%g	floating point	decimal number, possibly in scientific notation depending on the precision and value.

%h	any type	Hex String of value from hashCode() method.
%o	integer (incl. byte, short, int, long, bigint)	Octal number
%s	any type	String value
%t	Date/Time (incl. long, Calendar, Date and TemporalAccessor)	%t is the prefix for Date/Time conversions. More formatting flags are needed after this. See Date/Time conversion below.
%x	integer (incl. byte, short, int, long, bigint)	Hex string.

Tutorial 1.1: Number Formatting with printf

The **printf** function in Java is a tool used for formatting and displaying text. It allows you to create structured output by combining text and variables in a highly customizable way. With its wide range of format specifiers, such as **%d** for integers, **%f** for floating-point numbers, and **%s** for strings, it enables precise control over how data is presented.

printf does not include a line return like **println** does. Insert the new line escape character **\n** at the end of a statement to get a new line.

Create a Java program named: **PrintfNumberFormatting.java**

```java
/**
 * Name: PrintfExample.java
 * Written by:
 * Written on:
 * Purpose: How to print formmated numbers in Java with printf
 * printf and format do the same thing
 */

public class PrintfExample {
    Run | Debug
    public static void main(String args[]) {
        System.out.println("Demonstrate printf formatting");

        // This program demonstrates printf formatting
        // It is better to declare all variables at the top of the program
        // This program is written like this for demonstration of formatting
        int x = 100;
        System.out.printf(
                "Printing simple integer: x = %d\n", x);

        x = 125841;
        System.out.printf(
                "Printing integer with , 1000 separators: x = %,d\n", x);

        // Print it upto 2 decimal places
        System.out.printf(
                "Formatted with precision: PI = %.2f\n",
                Math.PI);

        float n = 5.2f;
        // Automatically appends zero to the rightmost part of decimal
        System.out.printf(
                "Formatted to specific width: n = %.4f\n", n);

        n = 2324435.3f;
        // Number is formatted from right margin and
        // occupies a width of 20 characters
        System.out.printf(
                "Formatted to right margin: n = %,20.4f\n", n);
    }
}
```

Example run:

```
3.1
3.14
3.142
3.1416
3.14159
Printing simple integer: x = 100
Formatted with precision: PI = 3.14
Formatted to specific width: n = 5.2000
Formatted to currency: $2,324,435.30
Format 2 numbers: a = 35.56 b = 40.1245
```

Tutorial 1.2: DiscountPrice with System.out.Format

System.out.format allows you to define a format string with placeholders for variables, like the familiar **printf** function in Java and C. By using format specifiers like **%s** for strings, **%d** for integers, and **%f** for floating-point numbers, you can precisely control how data is displayed.

Create a Java program named **DiscountPrice.java**

```java
/*
 * Name: DiscountPrice.java
 * Written by:
 * Written on:
 * Purpose: Calculate a 15% discount on a price
 */

import java.util.Scanner;

public class DiscountPrice {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        double price;
        double discountPrice;

        System.out.print("Enter the price: ");

        price = input.nextDouble();
        discountPrice = price * 0.85;

        // %.2f is replaced by discountPrice display to two decimals
        // % is a format specifier, if you want to print a %, you have to put %%
        System.out.format("Your price is $%.2f after 15%% discount.\n", discountPrice);
        // Close scanner OS resources
        input.close();
    }
}
```

Example run:

```
Enter the price: 22.36
Your price is $19.01 after 15% discount.
```

Assignment: Format Numbers

Take one of our earlier programs, and add printf formatting to it.

Assignment Submission

1. Attach the program files.
2. Attach screenshots showing the successful operation of the program.
3. Submit in Blackboard.