Java printf( ) Method overview

*System.out.printf("String and format-string " *, arg1, arg2, … +);*

Format String:

Format String is composed of literals and format specifiers. Arguments are required only if there are format specifiers in the format string respectively. Format specifiers can include: flags, width, precision, and conversion characters in the following sequence:

% {flags} {width} {.precision} conversion-character (**Braces** denote optional parameters)

Flags:

- : left-justify (default is to right-justify)
+ : output a plus (+) or minus (-) sign for a numerical value
0 : forces numerical values to be zero-padded (default is blank padding)
, : comma grouping separator (for numbers > 1000)
: space will display a minus sign if the number is negative or space if it is positive

Width:

Specifies the field width for outputting the argument and represents the minimum number of characters to be written to the output. Include space for expected commas and a decimal point in the determination of the width for numerical values.

Precision:

Used to restrict the output depending on the conversion. It specifies the number of digits of precision when outputting floating-point values or the length of a substring to extract from a String. Numbers are rounded to the specified precision.

Conversion-Characters:

d :	decimal integer [byte, short, int, long]	
f :	floating-point number [float, double]	
c :	character	Capital C will uppercase the letter
s :	String	Capital S will uppercase all the letters in the string
h :	hashcode	A hashcode is like an address. This is useful for printing a reference
n :	newline	Platform-specific newline character- use %n instead of \n for greater compatibility

Examples:

```
System.out.printf("Total is: $%,.2f%n", dblTotal);
System.out.printf("Total: %-10.2f: ", dblTotal);
System.out.printf("% 4d", intValue);
System.out.printf("%20.10s\n", stringVal);

String s = "Hello World";
System.out.printf("The String object %s is at hash code %h%n", s, s);
```

String class format( ) method:

You can build a formatted String and assign it to a variable using the static format method in the String class. The use of a format string and argument list is identical to its use in the printf method. The format method returns a reference to a String. Example:

```
String grandTotal = String.format("Grand Total: %,.2f", dblTotal);
```

Converter	Flag	Explanation
d		A decimal integer.
f		A float.
n		A new line character appropriate to the platform running the application. You should always use `%n`, rather than `\n`.
tB		A date & time conversion—locale-specific full name of month.
td, te		A date & time conversion—2-digit day of month. td has leading zeroes as needed, te does not.
ty, tY		A date & time conversion—ty = 2-digit year, tY = 4-digit year.
tl		A date & time conversion—hour in 12-hour clock.
tM		A date & time conversion—minutes in 2 digits, with leading zeroes as necessary.
tp		A date & time conversion—locale-specific am/pm (lower case).
tm		A date & time conversion—months in 2 digits, with leading zeroes as necessary.
tD		A date & time conversion—date as %tm%td%ty
	08	Eight characters in width, with leading zeroes as necessary.
	+	Includes sign, whether positive or negative.
	,	Includes locale-specific grouping characters.
	-	Left-justified..
	.3	Three places after decimal point.
	10.3	Ten characters in width, right justified, with three places after decimal point.

Here is a sample program that illustrates the use of `Formatting`:

```
import java.util.Calendar;
import java.util.Locale;

public class TestFormat {

    public static void main(String[] args) {
      long n = 461012;
      System.out.format("%d%n", n);      //  -->  "461012"
      System.out.format("%08d%n", n);    //  -->  "00461012"
      System.out.format("%+8d%n", n);    //  -->  " +461012"
      System.out.format("%,8d%n", n);    // -->  " 461,012"
      System.out.format("%+,8d%n%n", n); //  -->  "+461,012"

      double pi = Math.PI;

      System.out.format("%f%n", pi);       // -->  "3.141593"
      System.out.format("%.3f%n", pi);     // -->  "3.142"
      System.out.format("%10.3f%n", pi);   // -->  "     3.142"
      System.out.format("%-10.3f%n", pi);  // -->  "3.142"
      System.out.format(Locale.FRANCE,
                        "%-10.4f%n%n", pi); // -->  "3,1416"

      Calendar c = Calendar.getInstance();
      System.out.format("%tB %te, %tY%n", c, c, c); // -->  "May 29, 2006"

      System.out.format("%tl:%tM %tp%n", c, c, c);  // -->  "2:34 am"

      System.out.format("%tD%n", c);    // -->  "05/29/06"
    }
}
```