1.11 Escritura en pantalla

1. Introducción

La salida por pantalla en Java se hace con el objeto System.out. Este objeto es una instancia de la clase **PrintStream** del paquete java.lang. Si miramos la API de PrintStream obtendremos la variedad de métodos para mostrar datos por pantalla, algunos de estos son:

- *print* y *println*: muestran los datos por pantalla. Están sobrecargados para varios tipos de datos. El *println* inserta un salto de línea después de mostrar el dato.
- *printf*: escribe una cadena de texto utilizando formato.

En *print* y *println*, cuando queramos escribir un mensaje y el valor de una variable debemos utilizar el operador de `concatenación` de cadenas (+), por ejemplo:

```
System.out.println(“Bienvenido, “ + nombre);
```

Escribe el mensaje de "Bienvenido, Carlos", si el valor de la variable *nombre* es Carlos. En *print* y *println*, todas las variables que se muestran se consideran como cadenas de texto sin formato, por ejemplo, no sería posible indicar que escriba un número decimal con dos cifras decimales. Para ello se utiliza *printf*.

2. El método printf

`printf(String format, Object… args)`

En `String format` se coloca la cadena a la cual se le quiere dar formato. Utiliza unos códigos de conversión para indicar del contenido a mostrar de qué tipo es. Estos códigos se caracterizan porque llevan delante el símbolo %, algunos de ellos son:

- `%c` : carácter.
- `%s` : cadena de texto.
- `%d` : entero.
- `%f` : número decimal.

También se pueden usar las secuencias de escape que van precedidas de `\`, como por ejemplo `\n` y `\t`, incluso el salto de línea `\n` también se puede especificar como `%n`.

En `Object... args` se colocan los valores correspondientes. El primero corresponde al primer %, el segundo al segundo % y así sucesivamente. Dichos valores tienen que ser expresiones que devuelvan resultados con tipos compatibles a los % que correspondan.

Ejemplos:

```
System.out.printf("El carácter es %c",'a');
```

Nos mostraría por pantalla: `El carácter es a`

```
int integer = 10;
System.out.printf("%d", integer);
```

Salida por pantalla: `10`

Si queremos mostrar el símbolo %, se utiliza otro % delante:

```
System.out.printf("El 20%% de %d es %d\n",200,200*20/100);
```

Salida por pantalla: `El 20% de 200 es 40`

2.1 Número de decimales

Para `%f` podemos especificar el número de decimales escribiendo `.n` entre % y f, siendo n el número de decimales:

```
System.out.printf("%.2f", 12.3698);
```

La salida por pantalla es `12,37` ya que se realiza un redondeo para mostrar los decimales indicados.

2.2 Números con signo

Para mostrar números con signo se utiliza el símbolo **+** entre % y d si es un número entero o entre % y f si es un número decimal.

```
int integer = 10;
System.out.printf("%+d", integer);
```

Salida por pantalla: `+10`.

```
int integer = -10;
System.out.printf("%+d", integer);
```

Salida por pantalla: `-10`.

```
double decimal = 3.968;
System.out.printf("%+.2f", decimal);
```

Salida por pantalla: `+3,97`

```
double decimal = -3.968;
System.out.printf("%+f", decimal);
```

Salida por pantalla: `-3,968000`

2.3 Mostrar más de un valor

Utilizaremos tantos % como valores vamos a formatear. Después de la primera coma, se van poniendo los valores separados por comas, el primero corresponde al primer %, el segundo al segundo % y así sucesivamente.

```
double decimal = 1.25036;
int integer = 10;
System.out.printf("decimal = %.2f integer = %d", decimal, integer);
```

Salida por pantalla: `decimal = 1,25 integer = 10`

También podemos cambiar el órden por defecto de correspondencia entre los valores y los % con el símbolo $. Ejemplo: `%2$d` significa que al número entero(*%d*) se le va a asignar el segundo valor(*2$*).

```
System.out.printf("decimal1 = %2$.2f integer = %1$d decimal2 = %2$+.1f",
integer, decimal);
```

Salida por pantalla: `decimal1 = 1,25 integer = 10 decimal2 = +1,3`

Ejemplo: mostrar el número 123.4567 y su cuadrado ambos con dos decimales:

```
double decimal = 123.4567;
System.out.printf("El cuadrado de %.2f es %.2f", decimal, decimal*decimal);
```

Salida por pantalla: `El cuadrado de 123,46 es 15241,56`

2.4 Especificar un ancho de campo

printf permite también mostrar valores con un ancho de campo determinado. Por ejemplo, si queremos mostrar un número entero en un ancho de campo de 10 caracteres, escribimos 10 entre *%* y *d*:

```
int integer = 1234;
System.out.printf("Ancho de 10 caracteres con un entero:%10d",integer);
```

Salida por pantalla:

```
Ancho de 10 caracteres con un entero:      1234
```

Otro ejemplo con números decimales: mostrar un número decimal con dos decimales, con signo y en un ancho de campo de 10 caracteres:

```
double decimal = 1.25036;
System.out.printf("Ancho de 10 caracteres con un decimal:%+10.2f",decimal);
```

Salida por pantalla:

```
Ancho de 10 caracteres con un decimal:     +1,25
```

En el ancho de 10 caracteres, se cuentan además de las cifras del número, la coma decimal y el signo si lo lleva. En este caso, el número ocupa un espacio de 5 caracteres (3 cifras, la coma y el signo), por lo tanto, se añaden 5 espacios en blanco al principio para completar el tamaño de 10.

Para completar el ancho de caracteres con ceros en lugar de con espacios, se coloca un 0 delante del ancho de caracteres:

```
double decimal = 1.25036;
System.out.printf("Ancho de caracteres rellenado con ceros:%+010.2f",
decimal);
```

Salida por pantalla:

```
Ancho de caracteres rellenado con ceros:+000001,25
```

Veamos un ejemplo con cadenas: mostrar la cadena "Manolo" con un ancho de 10 caracteres:

```
System.out.printf("Ancho de caracteres con cadenas:%10s", "Manolo");
```

Salida por pantalla:

```
Ancho de caracteres con cadenas:    Manolo
```

2.5 Alineación a la izquierda

Con el signo `-` se indica alineación a la izquierda.

Ejemplo: mostrar un decimal con un ancho de 9 caracteres, con tres decimales y alineado a la izquierda:

```
double decimal=58.965874f;
System.out.printf("decimal=%-9.3fQue ocupe 9 caracteres, con tres decimales y
alineado a la izquierda", decimal);
```

Salida por pantalla:

```
decimal=58,966   Que ocupe 9 caracteres, con tres decimales y alineado a la
izquierda
```

Ejemplo con cadenas: mostrar la cadena "Manolo" con un ancho de 10 caracteres y alineada a la izquierda:

```
System.out.printf("%-10s:Alineación a la izquierda con cadenas", "Manolo");
```

Salida por pantalla:

```
Manolo    :Alineación a la izquierda con cadenas
```

He aquí todos los ejemplos de *printf* vistos en este apartado:

```
package tema1_11_EscrituraEnPantalla;

public class Printf {
```

```
    public static void main(String[] args) {

        double decimal;
        int integer;

        System.out.printf("El carácter es %c%n", 'a');// El carácter es a
        integer = 10;
        System.out.printf("%d\n", integer);// 10
        System.out.printf("El 20%% de %d es %d\n", 200, 200 * 20 / 100);// El
20% de 200 es 40
        System.out.printf("%.2f\n", 12.3698);// 12,37
        decimal = 1.25036;
        System.out.printf("%.3f\n", decimal);// 1,250
        System.out.printf("%+d\n", integer);// +10
        integer = -10;
        System.out.printf("%+d\n", integer);// -10
        decimal = 3.968;
        System.out.printf("%+.2f\n", decimal);// +3,97
        decimal = -3.968;
        System.out.printf("%+f\n", decimal);// -3,968000
        decimal = 1.25036;
        integer = 10;
        System.out.printf("decimal = %.2f integer = %d\n", decimal, integer);
        // decimal = 1,25 integer = 10
        System.out.printf("decimal1 = %2$.2f integer = %1$d decimal2 =
%2$+.1f\n", integer, decimal);
        // decimal1 = 1,25, integer = 10, decimal2 = +1,3
        decimal = 123.4567;
        System.out.printf("El cuadrado de %.2f es %.2f\n", decimal, decimal *
decimal);
        integer = 1234;
        System.out.printf("Ancho de 10 caracteres con un entero:%10d\n",
integer);
        decimal = 1.25036;
        System.out.printf("Ancho de 10 caracteres con un decimal:%+10.2f\n",
decimal);
        decimal = 1.25036;
        System.out.printf("Ancho de caracteres rellenado con
ceros:%+010.2f\n", decimal);
        System.out.printf("Ancho de caracteres con cadenas:%10s\n", "Manolo");
        decimal = 58.965874f;
        System.out.printf("decimal=%-9.3fQue ocupe 9 caracteres, con tres
decimales y alineado a la izquierda\n",
                decimal);
        System.out.printf("%-10s:Alineación a la izquierda con cadenas\n",
"Manolo");

    }

}
```

3. Ejemplo de Scanner y printf

He aquí un ejemplo de entrada/salida de datos utilizando Scanner y printf:

```
package tema1_11_EscrituraEnPantalla;

import java.util.Scanner;

public class InputOutput {

    @SuppressWarnings("resource")
    public static void main(String[] args) {
```

```
        Scanner keyboard = new Scanner(System.in);
        String name;
        int age;
        float salary;

        // Entrada de datos
        System.out.print("Nombre: ");
        name = keyboard.nextLine();
        System.out.print("Edad: ");
        age = keyboard.nextInt();
        System.out.print("Salario: ");
        salary = keyboard.nextFloat();

        // Salida de datos
        System.out.printf("\nBienvenido: %s\n", name);
        System.out.printf("Tienes: %d años\n", age);
        System.out.printf("Tu salario es: %.2f euros\n", salary);

    }

}
```

4. Colores

Para poder utilizar colores en la escritura de datos por pantalla, hay que instalar un plugin de Eclipse:

Help → *Eclipse Marketplace* → Find *'ANSI'* → Install *'ANSI Escape in Console'*.

Las secuencias de escape ANSI permiten enviar información de control a la consola para cambiar los atributos del texto representado. Solo debemos anteponer a la cadena que queremos mostrar en color el código de escape ANSI referente al color.

Cuando aplicamos un color a la salida por consola, el resto de salida por consola seguirá saliendo de dicho color hasta que se especifique otro color de salida o finalicemos el texto con el código RESET, en cuyo caso se vuelve al color por defecto.

```
package tema1_11_EscrituraEnPantalla.colores;

public final class Colors {

    public static final String RESET = "\u001B[0m";
    public static final String BLACK = "\u001B[30m";
    public static final String RED = "\u001B[31m";
    public static final String GREEN = "\u001B[32m";
    public static final String YELLOW = "\u001B[33m";
    public static final String BLUE = "\u001B[34m";
    public static final String PURPLE = "\u001B[35m";
    public static final String CYAN = "\u001B[36m";
    public static final String WHITE = "\u001B[37m";

    public static final String BLACK_BACKGROUND = "\u001B[40m";
    public static final String RED_BACKGROUND = "\u001B[41m";
    public static final String GREEN_BACKGROUND = "\u001B[42m";
    public static final String YELLOW_BACKGROUND = "\u001B[43m";
    public static final String BLUE_BACKGROUND = "\u001B[44m";
    public static final String PURPLE_BACKGROUND = "\u001B[45m";
    public static final String CYAN_BACKGROUND = "\u001B[46m";
    public static final String WHITE_BACKGROUND = "\u001B[47m";

    public static final String BOLD = "\u001B[1m";// Negrita
```

```
    public static final String UNDERLINE = "\u001B[4m";// Subrayado
    public static final String REVERSED = "\u001B[7m";// Invierte los colores
del texto y del fondo

}
```

```
package tema1_11_EscrituraEnPantalla.colores;

import static tema1_12_EscrituraEnPantalla.colores.Colors.*;

public class ColorsUse {

    public static void main(String[] args) {

        System.out.println(RED + "Este texto es de color rojo" + RESET);
        System.out.println("Volvemos al color por defecto");
        System.out.println(GREEN + "...y ahora es verde");
        System.out.println(PURPLE_BACKGROUND + "Fondo morado");
        System.out.println(CYAN + WHITE_BACKGROUND + "Fondo blanco con texto
celeste");
        System.out.println(CYAN + WHITE_BACKGROUND + BOLD + "Fondo blanco con
texto celeste en negrita");
        System.out.println(CYAN + WHITE_BACKGROUND + UNDERLINE + "Fondo blanco
con texto celeste subrayado");
        System.out.printf("%s\n", YELLOW + RED_BACKGROUND + (char) 9733);//
Estrella
        System.out.println(YELLOW + GREEN_BACKGROUND + "Fondo verde con texto
amarillo");
        System.out.println(REVERSED + "Fondo amarillo con texto verde usando
REVERSED");

    }
}
```