Manual JAVA

Víctor Chico Rodríguez

November 8, 2019

Contents

1 Introducción

Java es un lenguage de programación orientado a objetos, se creó en 1996 y está basado en C++, es un lenguaje que se ejecuta en una máquina virtual que interpreta (la JVM, siglas de *Java Virtual Machine*) las instrucciones compiladas a bytecode (El lenguaje de la máquina virtual)

1.1 Estructura de un programa

```
//Este fichero pertecene al paquete (carpeta) curso.java.manual (curso/java/manual)
package curso.java.manual;

//Esto es una importación de la clase Scanner del paquete java.io
import java.io.Scanner;

//Esto es una clase pública que se llama HolaMundo
public class HolaMundo {

    /*Este es el método main (principal),
es un método especial que servirá comom punto de entrada a la aplicación.
Este método es:
- público (puede ser accedido desde cualquier clase)
- estático (puede ser accedido sin necesidad de crear un objeto de la clase
-  no devuelve nada (tipo void)
- recibe como argumentos un array (matriz) de objetos de tipo String (cadena de texto)
*/
    public static void main (String [] args) {
```

```
//Este es un objeto de la clase Scanner que se llama scan
//Este objeto se inicializa con la palabra reservada new y
//recibe como argumento System.in (Entrada del sistema)
Scanner scan = new Scanner(System.in);

/*
  Esto es una llamada a un método, concretamente al método print(String)
  del atributo out de la clase System, este método imprime en la pantalla
  (consola de texto) el texto que se le pase como parámetro y continúa
  en la misma línea.
  Como parámetro se le pasa la cadena de texto (String) "¿Cómo te llamas?"
  Los valores de tipo String van siempre entre comillas dobles "
 */
System.out.print("¿Cómo te llamas?");

//Este es un objeto de la clase String (cadena de texto) que se llama nombre.
//Este objeto se inicializa automáticamente con el valor que devuleve
//el método readLine() (Método sin argumentos) del objeto scan.
String nombre = scan.readLine();

/*
  Esto es otra llamada a un método, en este caso al println del atributo
  out de la clase System, nótese la diferencia con la llamada anterior
  (print -- println), ese ln añadido lo que hace es saltar de línea una
  vez haya impreso lo que le pasemos como parámetro.

  En este caso, como parámetro se le pasa una cadena de texto (igual que antes)
  con el valor "Hola, " a lo que le concatenamos (sumamos) el valor de
  la variable nombre
 */
System.out.println("Hola, "+nombre);
    }
}
```

2 Tipos de datos

Los tipos de datos primitivos en java son los siguientes:

tipo	descripción	clase asociada
byte	número entero de 8 bits (-128 a 127)	Byte
short	número entero de 16 bits (-32768 a 32767	Short
int	número entero de 32 bits (-232 a 232)	Integer
long	número entero de 64 bits (-264 a 264)	Long
float	número decimal de 32 bits	Float
double	número decimal de 64 bits	Double
boolean	valor booleano o lógico (verdadero o falso)	Boolean
char	caracter de texto (único)	Character

Los tipos de datos normalmente se usan en su forma primitiva (columna tipo) y se pueden asignar directamente, pero a veces es util usar métodos de su clase asociada.

3 Operadores

3.1 Asignación

El operador = se usa para asignar valores a variables:

```
int a = 0;
```

3.2 Aritméticos

En java se pueden realizar multitud de operaciones matemáticas con la misma precedencia que en la vida real, si se necesita modificar se pueden utilizar paréntesis, los operadores aritméticos son los siguientes:

Operador	Descripción
+	Operador de suma
-	Operador de resta
*	Operador de multiplicación
/	Operador de división
%	Operador de resto de la división

El siguiente código es una pequeña demostración de los operadores mencionados:

```
class Aritmeticos {

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

```
// Variable de tipo int que tendrá como valor el resultado de 1 + 2
int resultado = 1 + 2;
// El valor de resultado es 3
System.out.println("1 + 2 = " + resultado);
int resultado_original = resultado;

// Los operadores se pueden usar entre variables (numéricas) y números
// en este caso se resta 1 al valor de resultado primero y se asigna a
// la variable resultado después
resultado = resultado - 1;
// El valor de resultado es 2
System.out.println(resultado_original + " - 1 = " + resultado);
resultado_original = resultado;

// Multiplicamos el resultado por 2 y lo volvemos a asignar a la variable
//resultado
resultado = resultado * 2;
// El valor de resultado es 4
System.out.println(resultado_original + " * 2 = " + resultado);
resultado_original = resultado;

// Dividimos el resultado entre 2 y lo asignamos
resultado = resultado / 2;
// El valor de resultado es 2
System.out.println(resultado_original + " / 2 = " + resultado);
resultado_original = resultado;

resultado = resultado + 8;
// El valor de resultado es 10
System.out.println(resultado_original + " + 8 = " + resultado);
resultado_original = resultado;

// Dividimos el resultado entre 7 y nos quedamos con el resto, luego lo
// asignamos
resultado = resultado % 7;
// El valor de resultado es 3
System.out.println(resultado_original + " % 7 = " + resultado);
    }
}
```

Como vimos anteriormente, el operador suma + se puede utilizar también para concatenar texto:

```
class Concatenacion {
    public static void main(String[] args){
String firstString = "Esto es";
String secondString = " una cadena de texto concatenada.";
String thirdString = firstString+secondString;
System.out.println(thirdString);
    }
}
```

3.2.1 Operadores unarios

En java hay un tipo de operadores aritméticos que sólo se utilizan en un operando, son los operadores unarios:

Operador	Descripción
+	Indica un valor positivo
-	Indica un valor negativo
++	Incrementa en 1 el valor
–	Decrementa en 1 el valor
!	Invierte el valor de un booleano

```
class Unarios {

    public static void main(String[] args) {

int resultado = +1;
// El resultado es 1
System.out.println(resultado);

resultado--;
// El resultado es 0
System.out.println(resultado);

resultado++;
// El resultado es 1
System.out.println(resultado);

resultado = -resultado;
```

```
// El resultado es -1
System.out.println(resultado);

boolean exito = false;
// false
System.out.println(exito);
// true
System.out.println(!exito);
    }
}
```

Los operadores de incremento y decremento ($++$ y $--$) actuan de manera diferente dependiendo de si se ponen delante o detrás del valor a modificar, si se usan de manera prefija `++variable` el valor se incrementa primero y la variable se usa después (ya incrementada), si se usa de manera postfija `variable++` se utilizará el valor de la variable sin incrementar y luego se incrementará:

```
class PrePost {
    public static void main(String[] args){
int i = 3;
i++;
// imprime 4
System.out.println(i);
++i;
// imprime 5
System.out.println(i);
// imprime 6
System.out.println(++i);
// imprime 6
System.out.println(i++);
// imprime 7
System.out.println(i);
    }
}
```