Universidad Europea

Fundamentos de programación

Tipos de datos, variables, constantes y asignación

Índice

Objetivos de aprendizaje

Los objetivos que se pretenden alcanzar con este recurso son los siguientes:

- Conocer cómo comentar código, las expresiones y las sentencias en Programación.
- Reconocer los tipos de datos en Java y sus principales características.
- Comprender las diferencias entre variables y constantes, así como la forma de asignarles valores en Java.

1. Literales y comentarios en código Java

Java es "*Case Sensitive*" respecto a los literales, es decir:

- Distingue entre minúsculas y Mayúsculas.
- ¡No es lo mismo **resultado** que **Resultado**!.

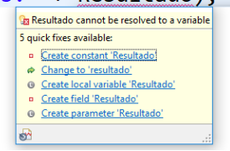
Comentarios de código

- Son fragmentos de texto embebidos en el código que son ignorados a la hora de compilar o interpretar el programa.
- Se utilizan para inhabilitar fragmentos de código o para anotar información útil para los desarrolladores.
- Existen dos tipos básicos de comentarios:
  - De línea: `// texto comentado`
  - De párrafo: `/* texto comentado`
  - `en varias líneas */`

Ejemplo

No es lo mismo **resultado** que **Resultado**

```
1 public class Hola {
2
3     public static void main (String[] args) {
4         int resultado = 2 + 3;
5         System.out.print("Resultado es:" + Resultado);
6     }
7 }
```



2. Expresiones

- Es la combinación de operadores y de operandos (los valores). Ejemplos de operadores:
  - Aritméticos: +, -, *, /, %, ...
  - Relacionales: <, <=, >, >=, ==
  - Lógicos: !, &, &&, |, ||
- Una expresión siempre tiene un resultado, determinado por la evaluación de su contenido.

Ejemplos de expresiones	evaluación
45	**45**
45 + 5	**50**
((4-6)*4)/10)-(20*3)	**¿-60.8 o -60?**

	evaluación
45 > 50	**false**
((2*4)==(4+4))&& (3<56)	**true**

3. Sentencias

- Sentencias:
  - Una sentencia es una orden dentro de un programa.
  - Puede estar formada por una o varias expresiones.
  - Termina siempre con un punto y coma: `;`
- Bloques de sentencias:
  - Son varias sentencias entre llaves: `{...}`
  - Constituyen un ámbito.

Ejemplo

Ejemplo de 3 sentencias:

```
x = x * 4 - 7;

background (44, 44, 56);

ellipse (100,x+20, 35, 35);
```

Ejemplo

Ejemplo de 2 bloques de sentencias (uno dentro de otro):

```
public class HolaMundo {
   public static void main(String[] args) {
      System.out.println("¡Hola Mundo!");
   }
}
```

4. Tipos de datos

Los tipos de datos permiten definir y conocer:

- El **rango de valores** que puede tomar un dato de ese tipo.
- La **cantidad de memoria** que necesitan para almacenarse.
- Las **operaciones** que se pueden aplicar a ese dato.

Según su complejidad se distinguen:

- **Simples**
    - Numéricos.
    - Carácter.
    - Lógicos.
    - Punteros.
- **Compuestos**
    - Arrays.
    - Cadenas de caracteres (Strings).
    - Registros.

4. Tipos de datos

4.1. Literales

Los literales son símbolos constantes que representan un valor determinado. En Java los tipos literales son los siguientes:

- Enteros (byte, short, int, long).
- Reales (double, float).
- Booleanos (boolean).
- Caracteres (char).
- Cadenas de caracteres (String).

1. **Enteros.** Tienen tres formatos posibles:

- Decimal (0-9). No existe notación especial.
- Hexadecimal (0-F). Precedido por 0x o 0X.
    - Ejemplo: **`0xA3, 0XA3`**
- Octal (0-7). Precedido por un 0.
    - Ejemplo: **`072`**

En Java: **`byte, short, int, long`**

4. Tipos de datos

4.1. Literales

2. **Reales** (en punto flotante). Pueden ser expresados en notación estándar o científica:

- Estándar: **`583.45`**
- Científica: **`5.8345e2`**

En Java: **`float, double`**

3. **Booleanos.** Tienen dos valores posibles:

- **`true o false.`**

En Java: **`boolean.`**

4. **Caracteres.** Representan caracteres en formato Unicode:
    - Conjunto de caracteres de 16 bits.
    - Permite la inclusión de símbolos especiales.
- **Formato:** un carácter encerrado entre comillas simples.
    - Ejemplo: **`'a'`**
- Algunos caracteres especiales incluyen la barra invertida: **`'\r', '\n', '\t'`** son retorno de carro, salto de línea, tabulador. Hay otros muchos.
- En Java: **`char`**

4. Tipos de datos

4.1. Literales

5. **Cadenas de caracteres.** Se representan mediante una secuencia de caracteres encerrados entre dobles comillas.

- Ejemplo: **`"¡Hola, mundo!"`**
- ¡OJO A las comillas!
    - No es lo mismo **`"foo"`** que **“foo”**

En Java: **`string`**

4. Tipos de datos

4.2. Resumen de posibles valores de los literales

Java define 8 tipos de datos simples o primitivos. En la tabla se muestran sus valores por defecto así como los rangos.

Tipo de Dato	Valor por Defecto	Valor Mínimo	Valor Máximo
byte	0	-128	127
short	0	-32,768	32,767
int	0	-2,147,483,648	2,147,483,647
long	0L	-9,223,372,036,854,775,808	9,223,372,036,854,775,807
float	0.0f	**	**
double	0.0d	**	**
char	'\u0000'	'\u0000‘ (0)	'\uffff‘ (65,535)
String*	null	""	Total de Memoria
boolean	false	false	true

Tabla 1. Valores por defecto y rango de valores de los tipos básicos de Java.

***** Técnicamente hablando el tipo String no es un primitivo, pero Oracle nos incentiva a tomarlo de esa manera debido a su soporte especial en el lenguaje.

****** Muy dependiente de la arquitectura hardware del sistema <https://docs.oracle.com/javase/specs/jls/se7/html/jls-4.html#jls-4.2>

*** En la tabla se utilizan las comas (',‘**)** como separadores de miles (no son decimales)

5. Variables

Variable: Representación simbólica que hace referencia a un valor que puede cambiar, almacenando su estado.

Características de las variables. Una variable tiene las siguientes características:

- **Nombre.** Debe seguir las reglas de los identificadores.
- **Tipo de dato**. Tipo de valor que puede almacenar.
- **Valor**. Información literal almacenada en la variable.

Sintaxis para la declaración de variables:

```
<tipo> <nombre> [ = <valor inicial> ] { , <nombre> [ = <valor inicial> ] };
```

Ejemplos de variables:

```
int numero1;          int numero2=1, numero3=7;          boolean encontrado;    float altura;
```

6. Constantes

Constante: Representación simbólica que hace referencia a un valor que **NO cambiará**, almacenando su estado.

Características de las constantes. Una constante tiene las siguientes características:

- **Nombre.** Debe seguir las reglas de los identificadores. Muchas veces se ponen en mayúsculas.
- **Tipo de dato.** Tipo de valor que puede almacenar.
- **Valor.** Información literal almacenada en la variable.

Sintaxis para la declaración de variables:

```
final <tipo> <nombre> [ = <valor inicial> ] { , <nombre> [ = <valor inicial> ] };
```

Ejemplos de variables:

```
final int MAX_PLAZAS = 5;        final boolean encontrado = true;  final float PI = 3.1416f;
```

7. Asignación

Asignación: Operación que permite **almacenar** el resultado de una expresión en una variable (o una constante en el caso de inicializaciones).

Sintaxis para la asignación:

```
<identificador> = <expresión>;
```

Ejemplo

Ejemplos de asignación:

Comentarios de línea

```
int numero1;       // declaración de variable
numero1 = 5;       // asignación de valor a la variable
int numero2 = 5;       // declaración de variable con inicialización (asignación de valor)
final float PI = 3.1416f;   // declaración de constante con inicialización
```

Referencias bibliográficas

Chapter 4. Types, Values, and Variables: <https://docs.oracle.com/javase/specs/jls/se7/html/jls-4.html>

Eckel, Bruce (2008). "Thinking in Java" 4th ed. Prentice Hall.

Horstmann, C.S. (2018). "Core Java I – Fundamentals" 11th ed. Prentice Hall.

Horstmann, C.S. (2016). "Core Java II – Advanced Features" 10th ed. Prentice Hall.

Schildt, H. (2018). "Java. A Beginner's Guide" 8th ed. Oracle Press.

Schildt, H. (2018). "Java. The Complete Reference" 11 th ed. Oracle Press.

Universidad Europea

Ve más allá