

Elementos básicos del lenguaje Java

Ricardo Pérez López

IES Doñana, curso 2020/2021

Generado el 3 de enero de 2021 a las 20:07:00

Índice general

1. Tipos y valores en Java	1
1.1. Introducción	2
1.2. Tipos primitivos	2
1.2.1. Integrales	3
1.2.2. De coma flotante	4
1.2.3. Booleanos	6
1.2.4. Subtipado	6
1.2.5. Conversiones entre datos primitivos	6
1.2.6. Promociones numéricas	6
1.3. Tipos por referencia	6
2. Variables en Java	6
2.1. Variables de tipos primitivos	6
2.2. Variables de tipos por referencia	6
2.3. Declaraciones de variables	6
3. Estructuras de control	7
3.1. Bloques	7
3.2. <code>if</code>	7
3.3. <code>switch</code>	7
3.4. <code>while</code>	7
3.5. <code>for</code>	7
3.6. <code>do ... while</code>	7
4. Entrada/salida	7
4.1. Flujos <code>System.in</code> , <code>System.out</code> y <code>System.err</code>	7
4.2. <code>java.util.Scanner</code>	7

1. Tipos y valores en Java

1.1. Introducción

El lenguaje de programación Java es un **lenguaje de tipado estático**, lo que significa que cada variable y cada expresión tiene un tipo que se conoce en el momento de la compilación.

El lenguaje de programación Java también es un **lenguaje fuertemente tipado**, porque los tipos limitan los valores que una variable puede contener o que una expresión puede producir, limitan las operaciones admitidas en esos valores y determinan el significado de las operaciones.

El tipado estático fuerte ayuda a **detectar errores en tiempo de compilación**.

Los **tipos** del lenguaje de programación Java se dividen en **dos categorías**:

- Tipos **primitivos**.
- Tipos **referencia**.

Consecuentemente, en Java hay dos categorías de **valores**:

- Valores primitivos.
- Valores referencia.

Los **tipos primitivos** son:

- El tipo **booleano** (`boolean`).
- Los tipos **numéricos**, los cuales a su vez son:
 - * Los tipos **integrales**: `byte`, `short`, `int`, `long` y `char`.
 - * Los tipos **de coma flotante**: `float` y `double`.

Los **tipos referencia** son:

- Tipos de **clase**.
- Tipos de **interfaz**.
- Tipos de **array**.

Además, hay un tipo especial que representa el **nulo** (`null`).

En Java, un objeto es una de estas dos cosas:

- O bien es una instancia creada dinámicamente de un tipo de clase.
- O bien es un **array** creado dinámicamente.

Los valores de un tipo referencia son referencias a objetos.

Todos los objetos, incluidos los **arrays**, admiten los métodos de la clase `Object`.

Las cadenas literales están representadas por objetos de la clase `String`.

1.2. Tipos primitivos

Los **tipos primitivos** están predefinidos en Java y se identifican mediante su nombre, el cual es una palabra clave reservada en el lenguaje.

Un valor primitivo es un valor de un tipo primitivo.

Los valores primitivos **no son objetos** y no comparten estado con otros valores primitivos.

En consecuencia, los valores primitivos no se almacenan en el montículo y, por tanto, las variables que contienen valores primitivos no guardan una referencia al valor, sino que almacenan el valor mismo.

Los tipos primitivos son los **integrales**, los tipos de **coma flotante** y los **booleanos**.

1.2.1. Integrales

Los **tipos integrales** son:

- **Enteros** (`byte`, `short`, `int` y `long`): sus valores son **números enteros con signo** en complemento a dos.
- **Caracteres** (`char`): sus valores son **enteros sin signo** que representan caracteres Unicode almacenados en forma de *code units* de UTF-16.

Sus tamaños y rangos de valores son:

Tipo	Tamaño	Rango
<code>byte</code>	8 bits	-128 a 127 inclusive
<code>short</code>	16 bits	32768 a 32767 inclusive
<code>int</code>	32 bits	2147483648 a 2147483647 inclusive
<code>long</code>	64 bits	9223372036854775808 a 9223372036854775807 inclusive
<code>char</code>	16 bits	'\u0000' a '\uffff' inclusive, es decir, de 0 a 65535

Los literales que representan números enteros pueden ser de tipo `int` o de tipo `long`.

Un literal entero será de tipo `long` si lleva un sufijo `l` o `L`; en caso contrario, será de tipo `int`.

Se pueden usar caracteres de subrayado (`_`) como separadores entre los dígitos del número entero.

Los literales de tipos enteros se pueden expresar en:

- **Decimal**: no puede empezar por `0`, salvo que sea el propio número `0`.
- **Hexadecimal**: debe empezar por `0x` o `0X`.
- **Octal**: debe empezar por `0`.
- **Binario**: debe empezar por `0b` o `0B`.

Ejemplos de literales de tipo `int`:

`0`

`2`

`0372`

`0xDada_Cafe`

1996

0x00_FF__00_FF

Ejemplos de literales de tipo `long`:

0l

0777L

0x100000000L

2_147_483_648L

0xC0B0L

Un literal de tipo `char` representa un carácter o secuencia de escape.

Se escriben encerrados entre comillas simples (también llamadas *apóstrofes*).

Los literales de tipo `char` sólo pueden representar *code units* de Unicode y, por tanto, sus valores deben estar comprendidos entre `'\u0000'` y `'\uffff'`.

Ejemplos de literales de tipo `char`:

'a'

'%'

'\t'

'\ '

'\''

'\u03a9'

'\uFFFF'

'\177'

'™'

En Java, los **caracteres** y las **cadenas** son **tipos distintos**.

1.2.2. De coma flotante

Los **tipos de coma flotante** son valores que representan **números reales** almacenados en el formato de coma flotante **IEEE-754**.

Existen dos tipos de coma flotante:

- **float**: sus valores son números de coma flotante de 32 bits (simple precisión).
- **double**: sus valores son números de coma flotante de 64 bits (doble precisión).

Un literal de coma flotante tiene las siguientes partes en este orden (que algunas son opcionales según el caso):

1. Una parte entera.

2. Un punto (.).
3. Una parte fraccionaria.
4. Un exponente.
5. Un sufijo de tipo.

Los literales de coma flotante se pueden expresar en decimal o hexadecimal (usando el prefijo `0x` o `0X`).

Todas las partes numéricas del literal (la entera, la fraccionaria y el exponente) deben ser decimales o hexadecimales, sin mezclar algunas de un tipo y otras de otro.

Se permiten caracteres de subrayado (`_`) para separar los dígitos de la parte entera, la parte fraccionaria o el exponente.

El exponente, si aparece, se indica mediante el carácter `e` o `E` (si el número es decimal) o el carácter `p` o `P` (si es hexadecimal), seguido por un número entero con signo.

Un literal de coma flotante será de tipo `float` si lleva un sufijo `f` o `F`; si no lleva ningún sufijo (o si lleva opcionalmente el sufijo `d` o `D`), será de tipo `double`.

El literal positivo finito de tipo `float` más grande es `3.4028235e38f`.

El literal positivo finito de tipo `float` más pequeño distinto de cero es `1.40e-45f`.

El literal positivo finito de tipo `double` más grande es `1.7976931348623157e308`.

El literal positivo finito de tipo `double` más pequeño distinto de cero es `4.9e-324`.

Ejemplos de literales de tipo `float`:

```
1e1f
2.f
.3f
0f
3.14f
6.022137e+23f
```

Ejemplos de literales de tipo `double`:

```
1e1
2.
.3
0.0
3.14
1e-9d
1e137
```

El estándar IEEE-754 incluye números positivos y negativos formados por un signo y una magnitud.

También incluye:

- Ceros positivo y negativos:

`+0`

`-0`

- Infinitos positivos y negativos:

`Float.POSITIVE_INFINITY`

`Float.NEGATIVE_INFINITY`

`Double.POSITIVE_INFINITY`

`Double.NEGATIVE_INFINITY`

- Valores especiales *Not-a-Number* (o NaN), usados para representar ciertas operaciones no válidas como dividir entre cero:

`Float.NaN`

`Double.NaN`

1.2.3. Booleanos

El tipo booleano (`boolean`) contiene dos valores, representados por los literales booleanos `true` (verdadero) y `false` (falso).

Un literal booleano siempre es de tipo `boolean`.

1.2.4. Subtipado

1.2.4.1. Subtipado entre tipos primitivos

1.2.5. Conversiones entre datos primitivos

1.2.5.1. Casting

1.2.5.2. De ampliación (*widening*)

1.2.5.3. De restricción (*narrowing*)

1.2.6. Promociones numéricas

1.3. Tipos por referencia

2. Variables en Java

2.1. Variables de tipos primitivos

2.2. Variables de tipos por referencia

2.3. Declaraciones de variables

3. Estructuras de control

3.1. Bloques

3.2. `if`

3.3. `switch`

3.4. `while`

3.5. `for`

3.6. `do ... while`

4. Entrada/salida

4.1. Flujos `System.in`, `System.out` y `System.err`

4.2. `java.util.Scanner`