Centro Asociado Palma de Mallorca

Introducción Práctica de Programación Java

Antonio Rivero Cuesta

Sesión

II

La Sintaxis de Java I 5

Tipos de datos 6

Tipos de datos simples 7

Operadores 11

Operadores Aritméticos 12

Operadores relacionales 13

Operadores lógicos 14

Operadores de bits 15

Operadores de asignación 16
Precedencia de operadores en Java 17
Secuencias de escape 19
Palabras reservadas 20

La Sintaxis de Java I

Tipos de datos

Tipos de datos simples

Tipo	Descripción	Long	Rango
byte	byte	1 byte	−128 a 127
short	Entero corto	2 bytes	−32768 a 32767
int	Entero	4 bytes	-2^{31} a $2^{31}-1$
long	Entero largo	8 bytes	-2^{63} a $2^{63}-1$
float	Real Coma Flotante	32 bytes	$\pm 3{,}4 \cdot 10^{38}$ a $\pm 1{,}4 \cdot 10^{-38}$
double	Real Coma Flotante DP	64 bytes	$\pm 1{,}8 \cdot 10^{308}$ a $\pm 4{,}9 \cdot 10^{-324}$
char	Carácter	2 bytes	0 a 65.535
boolean	Lógico	1 bit	true o false

El resto de tipos de datos que no son simples, son considerados ***referenciales***.

Estos tipos son básicamente las clases, en las que se basa la programación orientada a objetos.

Al declarar un objeto perteneciente a una determinada clase, se está reservando una zona de memoria donde se almacenarán los atributos y otros datos pertenecientes a dicho objeto.

Lo que se almacena en el objeto en sí, es un puntero, referencia, a dicha zona de memoria.

Dentro de estos tipos pueden considerarse las ***interfaces***, los ***Strings*** y los ***vectores***, que son unas clases un tanto especiales, y que se verán en detalle posteriormente.

A diferencia de otros lenguajes de programación, los ***Strings*** en Java no son un tipo simple de datos sino un objeto.

Los valores de tipo String van entre comillas dobles (“Hola”), mientras que los de tipo char van entre comillas simples

Tipos de datos referenciales

Tipo de datos simple	Clase equivalente
byte	**`java.lang.Byte`**
short	**`java.lang.Short`**
int	**`java.lang.Integer`**
long	**`java.lang.Long`**
float	**`java.lang.Float`**
double	**`java.lang.Double`**
char	**`java.lang.Character`**
boolean	**`java.lang.Boolean`**

Operadores

1. Aritméticos
2. Relaciónales
3. Lógicos
4. De bits
5. Asignación
6. Prioridad

Operadores Aritméticos

Operador	Formato	Descripción
`+`	`op1 + op2`	Suma aritmética de dos operandos
`-`	`op1 - op2`	Resta aritmética de dos operandos
`-`	`-op1`	Cambio de signo
`*`	`op1 * op2`	Multiplicación de dos operandos
`/`	`op1 / op2`	División entera de dos operandos
`%`	`op1 % op2`	Resto de la división entera o módulo
`++`	`++op1` `op1++`	Incremento unitario
`--`	`--op1` `op1--`	decremento unitario

Operadores relacionales

Operador	Formato	Descripción
`>`	`op1 > op2`	Devuelve `true` (cierto) si `op1` es mayor que `op2`
`<`	`op1 < op2`	Devuelve `true` (cierto) si `op1` es menor que `op2`
`>=`	`op1 >= op2`	Devuelve `true` (cierto) si `op1` es mayor o igual que `op2`
`<=`	`op1<= op2`	Devuelve `true` (cierto) si `op1` es menor o igualque `op2`
`==`	`op1 == op2`	Devuelve `true` (cierto) si `op1` es igual a `op2`
`!=`	`op1 != op2`	Devuelve `true` (cierto) si `op1` es distinto de `op2`

Operadores lógicos

Operador	Formato	Descripción
`&&`	`op1 && op2`	Y lógico. Devuelve `true` si son ciertos `op1` y `op2`
`||`	`op1 || op2`	O lógico. Devuelve `true` si son ciertos `op1` o `op2`
`!`	`!op1`	Negación lógica. Devuelve `true` si es `false op1`.

Estos operadores actúan sobre operadores o expresiones lógicas, es decir, aquellos que se evalúan a cierto o falso (true / false).

Operadores de bits

Operador	Formato	Descripción
`>>`	`op1 >> op2`	Desplaza `op1`, `op2` bits a la derecha
`<<`	`op1 << op2`	Desplaza `op1`, `op2` bits a la izquierda
`>>>`	`op1 >>> op2`	Desplaza `op1`, `op2` bits a la derecha (sin signo)
`&`	`op1 & op2`	Realiza un Y (AND) a nivel de bits
|	`op1 | op2`	Realiza un O (OR) a nivel de bits
`^`	`op1 ^ op2`	Realiza un O exclusivo (XOR) a nivel de bits
`~`	`~op1`	Realiza el complemento de `op1` a nivel de bits.

Operadores de asignación

Operador	Formato	Equivalencia
+=	op1 += op2	op1 = op1 + op2
-=	op1 -= op2	op1 = op1 - op2
*=	op1 *= op2	op1 = op1 * op2
/=	op1 /= op2	op1 = op1 / op2
%=	op1 %= op2	op1 = op1 % op2
&=	op1 &= op2	op1 = op1 & op2
|=	op1 |= op2	op1 = op1 | op2
^=	op1 ^= op2	op1 = op1 ^ op2
>>=	op1 >>= op2	op1 = op1 >> op2
<<=	op1 <<= op2	op1 = op1 << op2
>>>=	op1 >>>= op2	op1 = op1 >>> op2

Precedencia de operadores en Java

Operadores postfijos	`[] . (paréntesis)`
Operadores unarios	`++expr --expr -expr ~ !`
Creación o conversión de tipo	`new (tipo)expr`
Multiplicación y división	`* / %`
Suma y resta	`+ -`
Desplazamiento de bits	`<< >> >>>`
Relacionales	`< > <= >=`
Igualdad y desigualdad	`== !=`
AND a nivel de bits	`&`
XOR a nivel de bits	`^`
OR a nivel de bits	`|`
AND lógico	`&&`
OR lógico	`||`
Condicional al estilo C	`? :`
Asignación	`= += -= *= /= %= ^= &= |= >>= <<= >>>=`

Las secuencias de escape son combinaciones del símbolo contrabarra \ seguido de una letra, y sirven para representar caracteres que no tienen una equivalencia en forma de símbolo.

Las posibles secuencias de escape son:

Secuencias de escape

Secuencia	Significado
\’	Comillas simples
\”	Dobles comillas
\\	Contrabarra
\b	Retroceso
\n	Línea siguiente
\f	Form feed
\r	Retorno de carro
\t	Tabulador
\a	Alarma
\xxx	Carácter en octal
\0	Carácter nulo
\uxxxx	Carácter en hexadecimal Unicode

Palabras reservadas

abstract	continue	for	new	switch
assert	default	goto	package	synchronized
boolean	do	if	private	this
break	double	implements	protected	throw
byte	else	import	public	throws
case	enum	instanceof	return	transient
catch	extends	int	short	try
char	final	interface	static	void
class	finally	long	strictfp	volatile
const	float	native	super	while