Programació en Java: Literals

Poden existir diverses maneres de representar un valor. Per exemple, en tindre el número 100, significa exactament 100 en una base decimal, però parlem de 4 en una base binària o 256 en hexadecimal. Per a evitar ambigüitats, a Java es van establir algunes normes.

1. Literals enters
2. Literals de nombres reals
3. Literals char
4. Literals boolean
5. Literals String
6. Exemples de valors literals

Literals enters

Existeixen 3 maneres de representar el valor d'un nombre enter. Si s'escriu el número solament, aquest representa un número en base decimal. Si té un zero davant estem escrivint octals i si té el prefix "0x" el valor està representat en base hexadecimal. Per exemple:

Base de numeración	valor	valor decimal
Decimal	100	100
Octal	0100	64
Hexadecimal	0x100	256

Per a indicar-li al compilador que un valor donat és de tipus long hem d'incloure el sufix L:
long llarg= 800L;

Literals de nombres reals

En tractar amb nombres reals introduïm el punt decimal o utilitzem la notació científica.

```
double v = 300000.0;  // notación normal
double v = 3.0e+5;     // notación -científica
```

En l'exemple el valor 3.0e+5 significa: la lletra e denota la mantissa, i és igual si l'escrivim en majúscula o minúscula.
Per a determinar si estem parlant de float o double, al número se li agrega un sufix: **F** o **f** per a float i **D** o **d** per a double. Si no col·loquem un sufix el compilador el pren com double.

Literals char

Una variable de tipus char emmagatzema un caràcter Unicode. Aquest sistema de codificació amplia l'escassetat d'opcions que té la codificació ASCII per a incloure altres alfabets. Podem escriure en grec, hebreu o en vietnamita sense que el compilador proteste.
Un literal de tipus char s'expressa tancant el caràcter entre cometes simples:
char caracter = 'z';

Una altra manera de representar un caràcter és indicant el codi Unicode en hexadecimal, anteposant els símbols **\o**, tot entre cometes simples.
D'aquesta manera el símbol arrova (@) s'escriu:
char arrova = '\u0040';
El primer byte de la codificació Unicode és la mateixa que el vell i benvolgut ASCII

Literals boolean

Senzillament true o false (vertader o fals) sense cometes i en minúscula.
No tenen correlació amb cap número com ocorre en el llenguatge C que teníem al 0 com false i a l'1 com true.

Literals String

El tipus String no és un tipus de dada primitiva. Però es comporta com si ho fóra.
Per a indicar una cadena de caràcters es tanquen entre cometes dobles.
String cadena = "Time is money";

Exemples de valors literals

Literal	Tipo de dato
"Gato"	String
"G"	String
'G'	char
123	int
8000L	long
3.14	double
3.14D	double
3.14f	float
26.45e-45	double
'\u00ff'	char
'c'	char
"true"	String
true	boolean

En els sufixos que tenim per a identificar el tipus de dada no importa si són en minúscula o majúscula.
Però per convenció s'utilitzen en majúscula, perquè es pot confondre en llegir el codi una ela minúscula (l) amb un u (1).
Cal anar amb compte de no oblidar-nos els sufixos quan siguen necessaris.
El següent codi dóna un error en temps de compilació:
float f = 3.14;
El compilador ens donarà l'error de "possible loss of precision" (possible pèrdua de precisió).
Això es deu al fet que sense la F darrere, estem indicant un valor de tipus double.
El compilador es nega a comprimir un double de 64 bits en un float de tan sols 32 bits.
Ho solucionem d'aquesta manera:
float f = 3.14f;