

UD2.- Introducció a JAVA (II)

"Sempre codifica com si el tipus que acaba mantenint el teu codi fora un psicòpata violent que sap on vius"

Rick Osborne

8.- Literals.

A l'hora de tractar amb **valors dels tipus de dades simples (i Strings)** s'utilitza el que es denomina "**literals**".

Els literals són elements que serveixen per a representar un valor en el codi font del programa, es a dir, els valors que podem assignar a les variables.

A Java existeixen literals per als següents tipus de dades:

- **Lògics** (boolean).
- **Caràcter** (char).
- **Enters** (byte, short, int i long).
- **Reals** (double i float).
- **Cadenes de caràcters** (String).

• Literals lògics

Són únicament dos, les paraules reservades **true** i **false**. Exemple:

```
boolean activitat = false;
```

• Literals enters

Els literals de tipus enter: **byte**, **short**, **int** i **long** poden expressar-se en decimal (base 10), octal (base 8) o hexadecimal (base 16).

A més, pot afegir-se al final del mateix la lletra **L** per a indicar que l'enter és considerat com **long** (64bits).

En Java, el compilador identifica un **enter** decimal (base 10) en trobar un número el primer dígit del qual és qualsevol símbol decimal excepte el zero (de l'1 al 9). A continuació poden aparèixer dígits del 0 al 9.

La lletra **L** al final d'un literal de tipus enter pot aplicar-se a qualsevol sistema de numeració i **indica que el nombre decimalsiga tractat com un enter llarg (de 64 bits)**.

Aquesta lletra **L pot ser majúscula o minúscula**, encara que és aconsellable utilitzar la majúscula ja que en cas contrari pot confondre's amb el dígit un (1) en els llistats.

Exemple:

```
long max1 = 9223372036854775807L; //valor màxim per a un enter llarg
```

• Literals reals

Els literals de tipus real serveixen per a indicar valors **float** o **double**. A diferència dels literals de tipus enter, no poden expressar-se en octal o hexadecimal.

Existeixen **dos formats** de representació: mitjançant la seua part sencera, el punt decimal (.) i la part fraccionària; omitjançant **notació exponencial o científica**:

Exemples equivalents per a representar el número 3,1415

```
3.1415
0.31415e1
.31415e1
0.031415E+2
.031415e2
314.15e-2
31415E-4
```

Al igual que els literals que representen enters, **es pot posar una lletra com a sufix**. Aquesta lletra pot ser una **F** o una **D** (majúscula o minúscula indistintament).

F -> Tracta el literal com si fora de tipus **float**.

D -> Tracta el literal com si fora de tipus **double**.

Exemple:

```
3.1415F
.031415d
```

• Literals caràcter

Els literals de tipus caràcter es representen sempre entre **cometes simples (')**. Entre les cometes simples pot aparèixer:

- **Un símbol** (lletra) sempre que el caràcter estiga associat a un codi Unicode.

Exemples: 'a', 'B', '{', 'ñ', 'á'.

- Una **"seqüència de fuga o escapada"**. Les seqüències de fuga o escapada (de "escape") són combinacions del **símbol contrabarra \ seguit d'una lletra**, i serveixen per a **representar caràcters que no tenen una equivalència en forma de símbol**.

Les possibles seqüències d'escapada són:

```
\n → Nova Línia.
\t → Tabulador.
\r → Retorn de Carro. (Enter)
\f → Començament de Pàgina.
\b → Esborrat a l'Esquerra.
\\ → El caràcter barra inversa ( \ ).
\' → El caràcter prima simple ( ' ).
\" → El caràcter prima doble o bi-prima ( " ).
```

Per exemple :

```
Per a imprimir una diagonal inversa s'utilitza: \\
Per a imprimir cometes dobles en un String s'utilitza: \"
```

• Literals cadenes

Els **Strings o cadenes de caràcters** no formen part dels tipus de dades elementals a Java, sinó que són instanciats a partir de la classe `java.lang.String`, però accepten la seua inicialització a partir de literals d'aquest tipus, per la qual cosa es tracten en aquest punt.

Un literal de tipus string **va tancat entre cometes dobles (")** i ha d'estar inclòs completament en una sola línia del programa font (**no pot dividir-se en diverses línies**).

Entre les cometes dobles pot incloure's **qualsevol caràcter del codi Unicode** (o el seu codi precedit del caràcter \) **a més de les seqüències de fuga vistes anteriorment en els literals de tipus caràcter**.

Així, per exemple, **per a incloure un canvi de línia dins d'un literal de tipus string haurà de fer-se mitjançant laseqüència de fuga \n :**

Exemple:

```
System.out.println("Primera línia\nSegona línia de la  
cadena\n"); System.out.println("Hola");
```

La visualització del **string** anterior mitjançant **println()** produiria la següent eixida per pantalla:

```
Primera línia
Segona línia de la cadena

Hola
```

La manera d'incloure els caràcters: cometes dobles (") i contrabarra (\) és mitjançant les seqüències d'escapament \" i \\ respectivament (o mitjançant el seu codi Unicode precedit de \).

Si l'**String** és massa llarg i ha de dividir-se en diverses línies en el fitxer font, pot utilitzar-se **l'operador de concatenació d'Strings (+)** de la següent forma:

```
"Aquest String és massa llarg per a estar en una línia del " +  
"fitxer font i s'ha dividit en dues."
```

Exemple.- Programa que mostra per pantalla diversos literals de cadena combinats amb seqüències d'escapada

```
public class SeqüenciesEscapada {

    public static void main(String[] args) {

        // Per imprimir una línia utilitzem: \n

        System.out.println("Lorena\nVerònica\nDébora\nPasqual\nJaume");
        System.out.print("\n");
        System.out.println();

        // Per enviar un retorn de carro (ENTER): \r

        System.out.println("Dilluns\rDimarts, Dimecres\rDijous, Divendres");
        System.out.print("\n");
        System.out.println();

        // Per esborrar un caràcter a l'esquerra s'utilitza: \b
        // (SOLS EN MODE ADMINISTRADOR)

        System.out.println("Dilluns\bDimarts");
        System.out.print("\n");
        System.out.println();

        // Per a tabular s'utilitza: \t

        System.out.println("\tLunes\tMartes\tMiércoles");
        System.out.println("\tDilluns\tDimarts\tDimecres");
        System.out.println("\tMonday\tTuesday\tWednesday");
        System.out.print("\n");
        System.out.println();

        /*
        * Per imprimir comilles dobles (") en un String s'utilitza: \"
        * i per cometes (') simples: \'
        */

        System.out.println("\"Dilluns\", \"Dimarts\", 'Dimecres'");
        System.out.print("\n");
        System.out.println();

        //Per imprimir una diagonal inversa (\) s'utilitza: \\

        System.out.println("Pagina:\\ http://iesbenigaslo.es \\");

    } //Del main()
} //De la classe
```

Lorena
Verònica
Débora
Pasqual
Jaume

Dilluns
Dimarts, Dimecres
Dijous, Divendres

DillunDimarts

Lunes	Martes	Miércoles
Dilluns	Dimarts	Dimecres
Monday	Tuesday	Wednesday

"Dilluns", "Dimarts", 'Dimecres'

Pagina:\ http://iesbenigaslo.es \

9.- Eixida i Entrada estàndard.

• Eixida estàndard

Ja hem vist l'ús de **System.out** per a mostrar informació per pantalla:

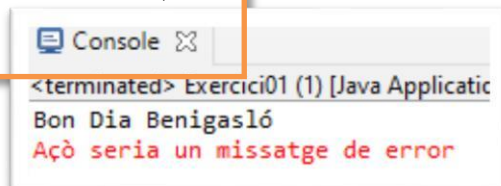
- `print("...");` // imprimeix text per pantalla
- `println("...");` // imprimeix text per pantalla i introdueix un salt de línia.

La utilització de **System.err** seria totalment anàloga per a enviar **els missatges produïts per errors** en l'execució (és el canal que usa també el compilador per a notificar els errors trobats).

Per exemple , per a presentar el missatge de salutació habitual per pantalla, i després un missatge d'error, tindríem la següent classe (encara que en realitat tota la informació va a la consola de comandos on estem executant el programa):

```
public static void main(String[] args) {  
  
    System.out.print("Bon Dia ");  
    System.out.println("Benigasló");  
    System.err.println("Açò seria un missatge de error");  
  
}
```

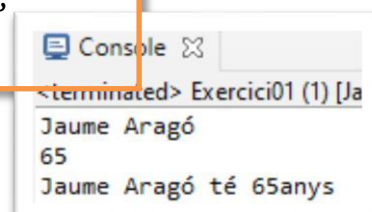
L'eixida serà:



També podem imprimir per pantalla variables de qualsevol tipus, així com combinacions de text i variables concatenades amb l'operador +

```
public static void main(String[] args) {  
  
    String nom = "Jaume Aragó";  
    int edat = 65;  
  
    System.out.println(nom);  
    System.out.println(edat);  
    System.out.println(nom + " té " + edat + "anys");  
  
}
```

L'eixida serà:



• Entrada estàndard

La entrada estàndard (llegir informació del teclat, escrita per l'usuari) és un poc més complexa. Hi ha diverses maneres de fer-ho però la més senzilla és utilitzar la classe **Scanner**.

Per a poder utilitzar la classe Scanner és necessari **importar-la** des del paquet **java.util** de Java.

```
import java.util.Scanner;
```

Sempre que vullgam llegir informació del teclat primer haurem de **declarar un objecte Scanner** que llegisca de l'entradaestandar **System.in** així:

```
Scanner elMeuObjecteScanner = new Scanner(System.in);
```

NOTA: En aquest exemple hem creat un objecte Scanner anomenat elMeuObjecteScanner però podríemposar-li qualsevol nom.

Ara podem utilitzar **elMeuObjecteScanner** tantes vegades com vulguem per a llegir informació del teclat. Per exemple:

```
String introduccioText = elMeuObjecteScanner.nextLine();  
int variableEntera = elMeuObjecteScanner.nextInt();
```

El mètode `reader.nextLine()` recollirà el text que l'usuari escriga per teclat (fins a pressionar la tecla Intro) i ho guardarà en `introduccioText` (de tipus `String`).

Existeixen molt altres mètodes segons el tipus de dada que es vullga llegir:

- **nextByte():** obté un nombre enter tipus **byte**.
- **nextShort():** obté un nombre enter tipus **short**.
- **nextInt():** obté un nombre enter tipus **int**.
- **nextLong():** obté un nombre enter tipus **long**.
- **nextFloat():** obté un nombre real **float**.
- **nextDouble():** obté un nombre real **double**.
- **next():** obté el següent **token** (text fins a un espai).
- **nextLine():** obté el text introduït per teclat dins que es pressione **INTRO**.

No existeixen mètodes de la classe Scanner per a obtenir directament booleans ni per a obtenir un sol caràcter.

Exemple 1.- Llegim una cadena de text i la mostrem per consola:

```
import java.util.Scanner;

public class ExempleScanner {

    public static void main(String[] args) {

        String nom;
        Scanner teclat = new Scanner(System.in);

        System.out.print("Introdueix el teu nom: ");
        nom = teclat.nextLine();
        System.out.print("Bon dia "+ nom);

    } //Del main
} // De la classe
```

Exemple 2: Llegim un valor tipus double. El programa demana a l'usuari que introdueixca el radi d'un cercle, després calcula la seua àrea i circumferència, finalment el mostra per pantalla.

```
import java.util.Scanner;

public class ExempleScanner2 {

    public static void main(String[] args) {

        double radi, area, longitud;
        Scanner entrada = new Scanner(System.in);

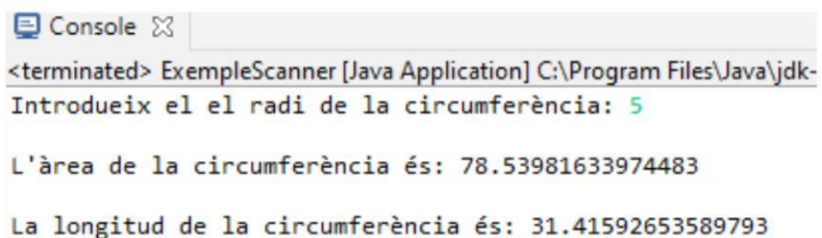
        System.out.print("Introdueix el el radi de la circumferència: ");
        radi = entrada.nextDouble();

        area= Math.PI * Math.pow(radi, 2);    // area = 3.14 * radi * radi

        longitud = 2 * Math.PI * radi;

        System.out.println("\nL'àrea de la circumferència és: "+ area);
        System.out.println("");
        System.out.println("La longitud de la circumferència és: "+ longitud);

    } //Del main
} // De la classe
```



```
Console
<terminated> ExempleScanner [Java Application] C:\Program Files\Java\jdk-
Introdueix el el radi de la circumferència: 5

L'àrea de la circumferència és: 78.53981633974483

La longitud de la circumferència és: 31.41592653589793
```

NOTA: A l'anterior exemple s'utilitzen constats i mètodes de la classe **Math**.

10.- Classe MATH.

Es troben a faltar operadors matemàtics més potents a Java. Per això s'ha inclòs una classe especial anomenada **Math** dins del paquet **java.lang**.

```
import java.lang.Math;
```

Aquesta classe conté molts mètodes interessants per a realitzar càlculs matemàtics complexos com a càlcul de potències, arrels quadrades, valors absoluts, si, cosinus, etc.

NOTA: Per poder invocar qualsevol constant o funció de la classe **Math** s'ha de posar el nom de la classe seguit d'un punt i el nom del mètode o constant a usar.

Per exemple:

```
double x = Math.pow(3,3);      // Potència 3 ^ 3
double i = Math.sqrt(9);      // Arrel quadrada de 9
```

També posseeix constants com:

```
double PI    // El número Π (3,14159265...)
double E     // El número e (2, 7182818245...)
```

Les funcions que més usarem de la classe **Math** són:

Funció Matemàtica	Significat	Exemple	Resultat
abs	Valor absolut	int x = Math.abs(2.3);	x = 2;
pow	Potència	double x = Math.pow(2.3);	x = 8.0;
sqrt	Arrel quadrada	double x = Math.sqrt(9)	x = 3.0;
round	Redondeig	double x = Math.round(2.5);	x = 3;
random	Número aleatori	double x = Math.random();	x = 0.20614522323378;
floor	Redondeig l'enter menor	double x = Math.floor(2.5);	x = 2.0;
ceil	Redondeig l'enter major	double x = Math.ceil(2.5);	x = 3.0;
max	Retorna el major de 2 números	int x = Math.max(5,4)	x=5
min	Retorna el major de 2 números	int x = Math.min(5,4)	x=4

Amb més detall:

• CONSTANTS

Constant	Descripció
E	Retorna el valor més aproximat del número e .
PI	Retorna el valor més aproximat del número pi .

• MÈTODES

Mètode	Descripció
abs(double a)	Retorna el valor absolut d'un valor double introduït com a paràmetre.
abs(float a)	Retorna el valor absolut d'un valor float introduït com a paràmetre.
abs(int a)	Retorna el valor absolut d'un valor enter introduït com a paràmetre.
abs(long a)	Retorna el valor absolut d'un valor long introduït com a paràmetre.
acos(double a)	Retorna l' arc-cosinus d'un valor introduït com a paràmetre.
addExact(int x, int y)	Retorna la suma dels seus arguments, llançant una excepció si el resultat desborda unint .
addExact(long x, long y)	Retorna la suma dels seus arguments, llançant una excepció si el resultat es desborda along .
asin(double a)	Retorna l' arc-sinus d'un valor introduït.
atan(double a)	Retorna l' arc-tangent d'un valor introduït.
cbrt(double a)	Retorna l' arrel cúbica d'un valor.
cos(double a)	Retorna el cosinus trigonomètric d'un angle.
exp(double a)	Retorna el número e de Euler elevat a la potència d'un valor de tipus double .
log(double a)	Retorna el logaritme natural (base e) d'un valor de tipus double .
log10(double a)	Retorna el logaritme de base 10 d'un d'un valor de tipus double .
max(double a, double b)	Retorna el major de dos valors de tipus double .
max(float a, float b)	Retorna el major de dos valors de tipus float .
max(int a, int b)	Retorna el major de dos valors Enters .
max(long a, long b)	Retorna el major de dos valors long .
min(double a, double b)	Retorna el menor de dos valors double .
min(float a, float b)	Retorna el menor de dos valors float .
min(int a, int b)	Retorna el menor de dos valors enters .
min(long a, long b)	Retorna el menor de dos valors long .
multiplyExact(int x, int y)	Retorna el producte dels arguments, llançant una excepció si el resultat desborda un int .
multiplyExact(long x, long y)	Retorna el producte dels arguments, llançant una excepció si el resultat desborda un long .
pow(double a, double b)	Retorna el valor del primer argument elevat a la potència del segon argument.
random()	Retorna un valor de tipus double amb un signe positiu, major o igual que 0.0 i menor que 1.0.
round(double a)	Retorna el long arredonit més pròxim al tipus double .
round(float a)	Retorna el int més pròxim i arredonit al float introduït.
sin(double a)	Retorna el sinus trigonomètric d'un angle.
sqrt(double a)	Retorna l' arrel quadrada positiva correctament arredonida d'un valor double .
tan(double a)	Retorna la tangent trigonomètrica d'un angle.

API DE JAVA ORACLE

<https://docs.oracle.com/javase/8/docs/api/>

<https://docs.oracle.com/en/java/javase/21/docs/api/index.html>