JAVA I - Introducció

Introducció a Java
CIFO Programació JAVA, Juny 2018

El programa i el codi font

Codi font

```
System.out.println("Hello world!");
```

Computers execute different operations, or actions, based on the commands. For example, when printing the text "Hello world!" on the screen, it is done by the command **System.out.println**.

The **System.out.println** command prints the string given inside the brackets on the screen. The suffix **ln** is short for the word line. Therefore, this command prints out a line. This means that after the given string has been printed, the command will also print a **line break.**

Cos del programa principal

```
public class Example {
    public static void main(String[] args) {
        System.out.print("Text to be printed");
    }
}
```

The program is stored in a text file named after the program with the **.java** extension. For a program named Example, the file should be named **Example.java**.

When we are talking about commands such as printing, we need to write the commands into the program body. For example: **System.out.print("Text to be printed");**

Variables

Variables i assignació

```
String text = "includes text";
int wholeNumber = 123;
double decimalNumber = 3.141592653;
boolean isTrue = true;

System.out.println("The variable's type is text. Its value is " + text);
System.out.println("The variable's type is integer. Its value is  " + wholeNumber);
System.out.println("The variable's type is decimal number. Its value is " + decimalNumber);
System.out.println("The variable's type is truth value. Its value is " + isTrue);
```

A **variable** is one of the most important concepts in computer programming. A variable should be imagined as a **box in which you can store information**. The information stored in a variable always has a **type**. These types include text (**String**), whole numbers (**int**), decimal numbers (**double**), and truth values (**boolean**).

A value can be assigned to a variable using the equals sign (=).

Càlcul

```
int first = 3;
int second = 2;
double result1 = (double)first / second;  // result is: 1.5

double result2 = first / (double)second;  // result is: 1.5

double result3 = (double)(first / second);  // result is: 1
```

The calculation operations are pretty straightforward: **+**, **-**, ***** and **/.**

A more peculiar operation is the modulo operation **%**, which calculates the remainder of a division. The order of operations is also pretty straightforward: the operations are calculated **from left to right taking the parentheses into account**.

Concatenació o combinació de cadenes

```
int x = 10;

System.out.println("variable x has the following value: " + x);

int y = 5;
int z = 6;

System.out.println("y has the value  " + y + " and z has the value " + z);
```

This program obviously prints:

```
variable x has the following value: 10
y has the value 5 and z has the value 6
```

Input

Llegir l'entrada de l'usuari

```java
import java.util.Scanner;

public class ProgramBody {
    public static void main(String[] args) {
        Scanner reader = new Scanner(System.in);

        // program code
    }
}
```

The **Java Scanner** class breaks the input into tokens using a delimiter that is whitespace by default. It provides **many methods to read and parse various primitive values**.

Java **Scanner class is widely used** to parse text for string and primitive types using regular expression.

Java Scanner class **extends Object class and implements Iterator and Closeable interfaces**.

Llegir l'entrada de l'usuari: string

```java
import java.util.Scanner;

public class Greeting {
    public static void main(String[] args) {
        Scanner reader = new Scanner(System.in);

        System.out.print("Who is greeted: ");
        String name = reader.nextLine(); // Reads a line of input from the user and assigns it
                                         //     to the variable called name

        System.out.print("Hi " + name);
    }
}
```

Llegir l'entrada de l'usuari: integer

```java
import java.util.Scanner;

public class NameAndAgeGreeting {
    public static void main(String[] args) {
        Scanner reader = new Scanner(System.in);

        System.out.print("Your name: ");
        String name = reader.nextLine();   // Reads a line from the users keyboard

        System.out.print("How old are you: ");
        int age = Integer.parseInt(reader.nextLine()); // Reads a string variable from the keyboard and transfers it to an integer

        System.out.println("Your name is: " + name + ", and you are " + age + " years old, nice to meet you!");
    }
}
```

Llegir l'entrada de l'usuari: methods

Method	Description
public String next()	it returns the next token from the scanner.
public String nextLine()	it moves the scanner position to the next line and returns the value as a string.
public byte nextByte()	it scans the next token as a byte.
public short nextShort()	it scans the next token as a short value.
public int nextInt()	it scans the next token as an int value.
public long nextLong()	it scans the next token as a long value.
public float nextFloat()	it scans the next token as a float value.
public double nextDouble()	it scans the next token as a double value.

més

Control de flux: if

Condicionals i valors de veritat: if

```
int first = 1;
int second = 3;

boolean isLesser = first < second;

if (isLesser) {
    System.out.println(first + " is less than " + second + "!");
}
```

The **comparison operators** are:

- > Greater than
- >= Greater than or equal to
- < Less than
- <= Less than or equal to
- == Equals
- != Not equal

Condicionals i valors de veritat: nested if

```
int x = 45;
int number = 55;

if (number > 0) {
    System.out.println("The number is positive!");
    if (number > x) {
        System.out.println(" and greater than the value of variable x");
        System.out.println("after all, the value of variable x is " + x);
    }
}
```

A block can contain any code **including** other if statements.

Condicionals i valors de veritat: if-else

```
int number = 4;

if (number > 5) {
    System.out.println("Your number is greater than five!");
} else {
    System.out.println("Your number is equal to or less than five!");
}
```

```
Your number is equal to or less than five!
```

If the truth value of the comparison is **false**, another **optional block** can be executed using the else command.

Condicionals i valors de veritat: else if

```
int number = 3;

if (number == 1) {
    System.out.println("The number is one.");
} else if (number == 2) {
    System.out.println("The number is two.");
} else if (number == 3) {
    System.out.println("The number is three!");
} else {
    System.out.println("Quite a lot!");
}
```

```
The number is three!
```

If there are more than two conditions for the program to check, it is recommended to use the **else if command**. It works like the else command, but with an additional condition. else if comes after the if command. There can be multiple else if commands.

Operacions lògiques

```
int number = 99;

if ((number > 0 && number < 10) || number > 100 ) {
    System.out.println("The number was in the range 1-9 or it was over 100");
} else {
    System.out.println("The number was equal to or less than 0 or it was in the range 10-99");
}
```

```
The number was equal to or less than 0 or it was in the range 10-99
```

The **condition statements** can be made more complicated using logical operations. The logical operations are:

- condition1 && condition2 is true if both conditions are true.
- condition1 || condition2 is true if either of the conditions are true.
- !condition is true if the condition is false.

Control de flux: while

Introducció als bucles: while

```
while (true) {
    System.out.println("I can program!");

    System.out.print("Continue? ('no' to quit)? ");
    String command = reader.nextLine();
    if (command.equals("no")) {
        break;
    }
}

System.out.println("Thank you and see you later!");
```

Now the loop is like this:

First, the program prints **I can program!**. Then, the program will ask the user if it should continue. If the user types **no**, the break command is executed and the loop is interrupted and **Thank you and see you again!** is printed.

Introducció als bucles

```
System.out.println("welcome to the calculator");

while (true) {
    System.out.print("Enter a command (sum, difference, quit): ");
    String command = reader.nextLine();
    if (command.equals("quit")) {
        break;
    }

    System.out.print("enter the numbers");
    int first = Integer.parseInt(reader.nextLine());
    int second = Integer.parseInt(reader.nextLine());

    if (command.equals("sum") ) {
        int sum = first + second;
        System.out.println( "The sum of the numbers is " + sum );
    } else if (command.equals("difference")) {
        int difference = first - second;
        System.out.println("The difference of the numbers is " + difference);
    } else {
        System.out.println("Unknown command");
    }

}

System.out.println("Thanks, bye!");
```

Introducció als bucles

```java
int number = 1;

while (number < 11) {
    System.out.println(number);
    number++;  // number++ means the same as number = number + 1
}
```

In the following example, we print the numbers 1, 2, ..., 10. When the **value of the variable number increases above 10**, the condition of the while statement is no longer true and the loop ends.

Introducció als bucles

```java
int number = 1024;

while (number >= 1) {
    System.out.println(number);
    number = number / 2;
}
```

In the previous example, the variable number **was incremented in each iteration of the loop**.

Generally the change **can be anything**, meaning that the variable used in the condition does not always need to be incremented.

For example: **number = number / 2**

Introducció als bucles

```
int result = 0;

int i = 0;
while (i < 4) {
   result += 3;  // this is the same as result = result + 3;
   i++;          // means the same as i = i+1;
}
```

In the beginning **result = 0**. During the loop, the value of the variable is incremented **by 3 on each iteration**. Because there are 4 iterations, the value of the variable is 3*4 in the end.

```
int length = 100;

length += 10;  // same as length = length + 10;
length -= 50;  // same as length = length - 50;
length *= 10;   // same as length = length * 10;
length /= 100;  // same as length = length / 100;
length %= 3;    // same as length = length % 3;
```

Introducció als bucles: bucle infinit

```
int i = 0;

while (i < 10) {
    System.out.println("Never again shall I program an eternal loop!");
}
```

One of the **classic errors** in programming is to accidentally create an **infinite loop**. In the example we try to print "**Never again shall I program an eternal loop!**" 10 times.

The variable **i**, which determines is supposed to index the loops, is initially set to 0. The block is looped as long as the condition i < 10 is true. But something **funny** happens. Because the value of the variable i is never changed, the condition stays true forever.