Programació I - Tema 3 - Mètodes d'objecte i mètodes de classe

UNIVERSITAT DE BARCELONA

Grau en Enginyeria Informàtica
Facultat de Matemàtiques i Informàtica
Curs 18-19

Programació I - Tema 3 - Mètodes d'objecte i mètodes de classe

1. 3.1 Introducció
2. 3.2 Mètodes d'objecte
3. 3.3 Mètodes de classe

1 3.1 Introducció

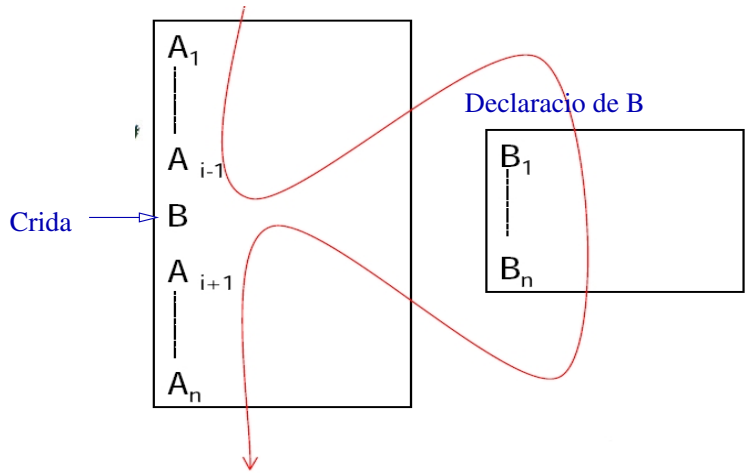
2 3.2 Mètodes d'objecte

3 3.3 Mètodes de classe

3.1 Introducció

Recordatori concepte de mètode

- Mètode: acció que pot realitzar un objecte d'una classe (veurem més endavant que aquesta definició es refereix a un tipus concret de mètode!).
- Una altre definició (més general): un mètode és la definició d'un conjunt de sentències (i dades) que realitzen una tasca determinada (el concepte anàlog a mètode en llenguatges estructurats, com ara el llenguatge C, és *funció*).
- **CRIDA a un mètode**: és la utilització del mètode per part d'un altre conjunt d'instruccions (per exemple: quan s'utilitza el Math.sqrt(x)).



3.1 Introducció

Recordatori concepte de mètode

- Tots els mètodes s'han de **definir** (què fan). Si hi han paràmetres formals, aquests indiquen el seu tipus de dades.
  - `<tipus_retorn> nomMetode(<tipus> <parametre_formal1> ... <tipus> <parametre_formaln>){...}`
- Una vegada definits, els mètodes es poden **cridar** (utilitzar!) les vegades que es vulgui:
  - Si no retorna res (s'ha definit amb tipus de retorn `void`):
    - `nomMetode (<parametre_actual1> ... <parametre_actualn> );`
  - Si retorna un valor:
    - `variable = nomMetode(<parametres_actuals>);`
      `variable` ha de ser del mateix tipus de dades que retorna el mètode.
  - A la crida, han d'haver tants paràmetres actuals com paràmetres formals hi han a la definició.
  - A la crida MAI es posa el tipus dels paràmetres, només el nom.

3.1 Introducció

Recordatori concepte de mètode

- Existeixen mètodes ja definits per la API de Java o per altres programadors, nosaltres ens limitem doncs a utilitzar-los (cridar-los).
  - Ex. `Scanner teclat = new Scanner(System.in); teclat.nextInt();`
  - Ex. `String s1 = new String("Hola"); String s2 = new String("Hola"); System.out.println(s1.equals(s2));`

3.2 Mètodes d'objecte

- **Mètode d'objecte**: mètode que manipula els atributs d'un objecte 'concret' d'una classe.
  - IMPORTANT: Ès imprencisdible tenir un objecte creat (amb new) per a poder cridar un mètode d'objecte.
    - Notació per accedir (quan l'utilitzem!!): `objecte.nomMetodeDObjecte()`
    - Recordeu què **this** permet referenciar, dintre d'un mètode d'objecte, a l'objecte amb qui s'ha cridat al mètode.
  - Ex. La crida al mètode `elMeuGos.menjar()` fa que la gana del `elMeuGos` tingui un valor menor que abans de menjar. Només l'atribut `gana` d'aquest objecte (`elMeuGos`) es modifica amb la crida al mètode!

- **Mètode de classe**: serveixen per manipular els atributs de classe (s'utilitza la paraula reservada **static**).
  - IMPORTANT: No es necessari crear un objecte de la classe per a cridar un mètode de classe.
    - Notació per accedir (quan l'utilitzem!!): `NomClasse.nomMetodeDeClasse()`
    - Ex. A la classe **Circle** podem definir un mètode `mostrarNumCircles()` per a imprimir per pantalla el nombre de circles creats d'aquesta classe. Recordeu que numCircles és un atribut de classe (static).
    - Ex. El mètode `main()` és un mètode static de la classe principal d'un programa Java.

Exemple: Càlcul de l'àrea d'un circle (ExempleAreaV1.java) I

```
import java.util.Scanner;
public class ExempleAreaV1 {
  public static void main (String[] args) {
    Scanner teclado = new Scanner(System.in);
    Circle circle0 = new Circle();
    Circle circle1 = new Circle(1.6);
    Circle circle2;
    double ra; //Podem anomenar-la radi enlloc de ra?

    System.out.println("Num de circles: " + Circle.numCircles);

    System.out.println("Area del circle0 " + "amb radi "
                       + circle0.radi + " = " + circle0.calcularArea()
                          );

    System.out.println("Area del circle1 " + "amb radi "
                       + circle1.radi + " = " + circle1.calcularArea()
                          );

    System.out.print("Introdueix el radi del circle2: ");
    ra = teclado.nextDouble();
    circle2 = new Circle (ra);
```

Exemple: Càlcul de l'àrea d'un circle (ExempleAreaV1.java) II

```
    System.out.println("Area␣del␣circle2␣"+ "amb␣radi␣"
                       + circle2.radi + "␣=␣" + circle2.calcularArea()
                           );

    //System.out.println("Num de circles: " + Circle.numCircles);
    Circle.mostraNumCircles();
    circle0.mostrarCircle();
    circle1.mostrarCircle();
    circle2.mostrarCircle();

  }
}

class Circle{
  double radi;
  int id;
  static int numCircles = 0;

  Circle(){
    radi = 0.0;
    id = ++numCircles;
```

Exemple: Càlcul de l'àrea d'un circle (ExempleAreaV1.java) III

```
  }
  Circle(double r){
    id = ++numCircles;
    radi = r;
  }
  double calcularArea(){
    return Math.PI * radi * radi;
  }
  void mostrarCircle(){
    System.out.println("El␣meu␣id:␣" + id + "␣i␣el␣meu␣radi:␣" + radi);
  }
  static void mostrarNumCircles(){
    System.out.println("S'han␣creat␣" + numCircles + "␣circles␣de␣la␣
        classe␣Circle");
  }
}
```