Herencia en el Ámbito de Clase

Dpto. Lenguajes y Sistemas Informáticos
Universidad de Granada

Programación y Diseño Orientado a Objetos

(Curso 2021-2022)

Créditos

- Las siguientes imágenes e ilustraciones son libres y se han obtenido de:
  - Emojis, `https://pixabay.com/images/id-2074153/`
- El resto de imágenes e ilustraciones son de creación propia, al igual que los ejemplos de código

Objetivos

- Entender las diferencias existentes entre Java y Ruby relacionadas con la herencia en el ámbito de clase

Contenidos

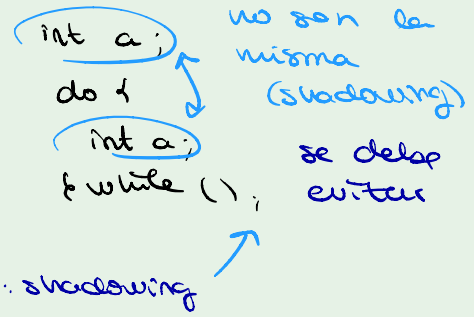
Java

- No permite la redefinición de métodos de clase al mismo nivel que de instancia
- Aunque pueden existir métodos de clase con el mismo nombre en una jerarquía de clases, no se obtienen los mismos resultados que a nivel de instancia

Ejemplo

Java: Ejemplo de herencia en el ámbito de clase

```
1  class Padre {
2    public static final int DECLASE = 1;
3    public static int getDECLASE() { return DECLASE; }
4  }
5
6  class Hija extends Padre {
7    public static final int DECLASE = 2; // Variable shadowing
8  }
9
10 class Nieta extends Hija{
11   public static int getDECLASE() { // No es una redefinición
12       // super.getDECLASE() No permitido
13       return DECLASE;
14   }
15 }
16
17 public static void main(String[] args) {
18   System.out.println (Padre.DECLASE);   // 1
19   System.out.println (Hija.DECLASE);    // 2
20   System.out.println (Nieta.DECLASE);   // 2
21   System.out.println (Padre.getDECLASE()); // 1
22   System.out.println (Hija.getDECLASE());  // 1
23   //porque "redefine" el método de clase
24   System.out.println (Nieta.getDECLASE()); // 2
25 }
```



no se permite a nivel de clase en Java

busca el método en el padre y ya se queda en ese método

encuentra el método en Nieta. Busca el atributo en nieta, va a Hija. Imprime Hija

Ejemplo

Java: Ejemplo de herencia en el ámbito de clase

```
1  public static void main(String[] args) {
2
3    // El tipo estático de las instancias influye
4
5    // Aunque Java lo permite, no se debe invocar a métodos de clase así
6    // Lo digo en serio
7
8    Padre p=new Padre(); tipo estático == tipo dinámico
9    System.out.println (p.getDECLASE()); // 1
10
11   p = new Nieta(); mismo p, asigno nieta ↘ tipo e = padre
12   System.out.println (p.getDECLASE()); // 1   ↘ tipo d = nieta → no tengo el mismo comportamiento que cuando lo llamaba correctamente
13
14   Nieta n = new Nieta();
15   System.out.println (n.getDECLASE()); // 2 → es el tipo estático quien determina dónde buscamos el método (!= polimorfismo)
16 }
```

Observamos que, EN ÁMBITO DE CLASE, no funciona el polimorfismo como hemos visto en el pwp anterior (a nivel de instancia).

Podríamos decir, entre comillas, que "no hay herencia a nivel de clase".

Ruby

- La clases son *first class citizens* y en el ámbito de clase todo funciona como es de esperar

Ejemplo

Ruby: Ejemplo de herencia en el ámbito de clase

```ruby
class Padre
  @atributo_clase1  = 1
  @atributo_clase2  = 2
  @@atributo_clase3 = 5
  def self.salida
    puts @atributo_clase1+1
    puts @atributo_clase2+1 unless @atributo_clase2.nil?
    puts @@atributo_clase3+1
  end
  def self.salida2
    salida
  end
end
Padre.salida # 2 3 6
class Hija < Padre
  @atributo_clase1  = 3
  @@atributo_clase3 = 7
  def self.salida2
    super # Las clases son "first class citizens"
    puts @atributo_clase1+1
  end
end
Padre.salida  # 2 3 8
Hija.salida   # 4 8
Padre.salida2 # 2 3 8
Hija.salida2  # 4 8 4
```

Los atributos de instancia de clase SIEMPRE se buscan en la propia clase (nunca ancestros) pq no puedo acceder a los de otra clase → @atr1 lo busca en Hija *
→ @atr2 lo busca en Hija, no lo encuentra, no lo coge de Padre.

→ antes de la declaración de Hija (la duda)

→ evitar atributos de clase salvo que tengamos claro que no habrá herencias.

→ la clase Hija cambia el comportamiento de Padre por

→ no tiene definido @atr2 (está a nil → no lo muestra)

→ mismo valor pq misma def. *

Herencia en el Ámbito de Clase

Dpto. Lenguajes y Sistemas Informáticos
Universidad de Granada

Programación y Diseño Orientado a Objetos

(Curso 2021-2022)