Програмування множинного наслідування реалізації на основі анонімних внутрішніх класів Java

Сергій Іщеряков, Юрій Яновський, Микола Козленко

Кафедра інформаційних технологій
Прикарпатський національний університет імені Василя Стефаника
м. Івано-Франківськ, Україна

Abstract—Запропоновано спосіб програмування множинного наслідування реалізації на основі анонімних внутрішніх класів Java.

Keywords—множинне наслідування, інтерфейс, анонімний внутрішній клас, фабричний метод, Java.

I. Вступ

Основною технологією множинного наслідування в об'єктних мовах є використання абстрактних конструкцій із повним відокремленням елементів реалізації. В Java такими конструкціями є класичні (до Java SE7 включно) інтерфейси, до складу яких можуть входити тільки абстрактні методи та статичні незмінні (`final`) поля. Використання в інтерфейсах дефолтних та статичних методів із наявними елементами реалізації, починаючи із Java SE8, погіршило "кришталеву" прозорість цих конструкцій, яка дозволяла розробникам здійснювати множинне наслідування без будь-яких штучних обмежень та встановлення рівнів пріоритетності.

Проте, навіть в класичних інтерфейсах відокремлення від реалізації надає наслідуванню – одній з фундаментальних основ об'єктної парадигми – декларативного характеру. Застосування багаторівневої ієрархії із використанням абстрактних класів надає розробникам можливість здійснювати повноцінне наслідування підкласами елементів суперкласів, але це не стосується множинного наслідування реалізації, обмеженого або повністю забороненого в об'єктних мовах.

II. Огляд літератури

В [1] описано технологію множинного наслідування класів із повноцінним наслідуванням реалізації суперкласів завдяки наявності в Java унікальних елементів – анонімних внутрішніх класів. Проте, наведені приклади програмних кодів із використанням фабричних методів, в яких формуються об'єкти анонімних внутрішніх класів, не демонструють можливості одночасного наслідування методів суперкласів з однаковою сігнатурою.

III. Виклад основного матеріалу

Нижче наведено Java-код, в якому класом `Smartphone` здійснюється наслідування методів `process()` двох суперкласів `Computer` та `Phone`. Клас Smartphone напряму наслідує реалізацію класу `Computer`, а наслідування методу `process()` класу `Phone` здійснюється через фабричний метод `makePhone()`, в якому формується об'єкт анонімного внутрішнього класу, що є підкласом класу `Phone`.

Ключовим елементом коду є лінійка з коментарем // key line, яка забезпечує звертання зсередини анонімного внутрішнього класу до методу зовнішнього класу. Логічне поле flag здійснює переключення між методами підкласу, що перевизначають методи суперкласів однакової сігнатури. За наявності більшої кількості суперкласів необхідно використовувати замість логічного (boolean) поля ціле (int) поле.

```
public class Dispatcher{
     public static void main(String[] args){
          Smartphone smart = new Smartphone();
          smart.process();
          smart.makePhone().process();
     }
}
class Phone{
     void process(){System.out.println("Phone");}
}
class Computer{
     void process(){System.out.println("Computer");}
}
class Smartphone extends Computer{
    boolean flag = true;
    void process(){
        super.process();
        System.out.println("Mobile as Computer");
        if (flag){
            flag = false;
            makePhone().process();
        }
        flag = true;
    }
    Phone makePhone(){
        return new Phone(){
            void process(){
                super.process();
                System.out.println("Mobile as Phone");
                if (flag){
                    flag = false;
                    Smartphone.this.process(); // key line
                }
                flag = true;
            }
        };
    }
}
```

IV. Висновки

Запропоновано спосіб програмування множинного наслідування реалізації на основі анонімних внутрішніх класів Java.

Література

[1] B. Eckel, *Thinking in Java*. Prentice Hall, 2006.