

Bài 54: Lớp vô danh

- ✓ Khái niệm và đặc điểm
- ✓ Các loại lớp vô danh
- ✓ Lưu ý
- ✓ Ví dụ minh họa

Khái niệm và đặc điểm

- Lớp vô danh (anonymous class) là một inner class không có tên và chỉ dùng để tạo một đối tượng
- Lớp vô danh không phải là thành phần của bất kì lớp nào
- Lớp vô danh hữu ích trong trường hợp ta muốn tạo đối tượng của lớp mà không muốn tạo lớp độc lập
- Thường sử dụng để tạo đối tượng thực thi của abstract class, interface
- Ví dụ: các đối tượng thực thi interface lắng nghe các sự kiện trong lập trình giao diện

Khái niệm và đặc điểm

- Lớp vô danh thường được tạo bằng hai cách:
 - Qua các lớp, gồm cả lớp thông thường và lớp abstract
 - Qua các interface

- Cú pháp tạo lớp vô danh:

```
Type objectName = new Type() {  
    // các dữ liệu cần thiết  
    // các phương thức  
    ...  
};
```

Ví dụ

➤ Ban đầu ta có interface Calculable:

```
public interface Calculable {  
    int add(int a, int b); // cộng hai số  
  
    int sub(int a, int b); // trừ hai số  
  
    default float div(int a, int b) { // chia hai số  
        return 0;  
    }  
  
    default int mul(int a, int b) {  
        return 1;  
    }  
}
```

Ví dụ

➤ Sau đó ta tạo lớp vô danh thực thi interface này:

```
class Example {  
    public static void main(String[] args) {  
        Calculable myCalculator = new Calculable() {  
            @Override  
            public int add(int a, int b) {  
                return a + b;  
            }  
  
            @Override  
            public int sub(int a, int b) {  
                return a - b;  
            }  
        };  
        // sử dụng đối tượng tạo ra từ Lớp vô danh:  
        System.out.println(myCalculator.add(100, 250));  
        System.out.println(myCalculator.sub(200, 125));  
    }  
}
```



Các loại lớp vô danh

- Lớp vô danh kế thừa từ một lớp khác
- Lớp vô danh thực thi interface
- Lớp vô danh trong tham số của phương thức

Loại 1

Lớp vô danh kế thừa từ một lớp khác

- Ta có thể tạo một lớp vô danh từ một lớp thông thường:

```
public class Father {  
    public void speak() {  
        System.out.println("Father speaking English");  
    }  
}  
  
class FatherExample {  
    public static void main(String[] args) {  
        // tạo lớp vô danh kế thừa lớp Father  
        Father son = new Father() {  
            @Override  
            public void speak() {  
                super.speak(); // gọi phương thức của lớp cha  
                System.out.println("Son is speaking Vietnamese");  
            }  
        };  
  
        son.speak();  
    }  
}
```

Loại 1

Lớp vô danh kế thừa từ một lớp khác

➤ Lớp abstract:

```
public abstract class Animal {  
    private String name;  
  
    public Animal(String name) {  
        this.name = name;  
    }  
  
    protected abstract void move(); // di chuyển  
  
    protected abstract void eat(); // ăn  
  
    public String getName() {  
        return name;  
    }  
}
```


Loại 1

Lớp vô danh kế thừa từ một lớp khác

➤ Tạo lớp vô danh từ lớp abstract:

```
class AnimalExample {  
    public static void main(String[] args) {  
        Animal cat = new Animal("Tom") {  
            @Override  
            protected void move() {  
                System.out.println("Mèo " + getName()  
                    + " đang chạy trên nóc nhà");  
            }  
  
            @Override  
            protected void eat() {  
                System.out.println("Mèo " + getName()  
                    + " đang ăn cá rán giòn");  
            }  
        };  
  
        System.out.println("Tên mèo là: " + cat.getName());  
        cat.move();  
        cat.eat();  
    }  
}
```

Loại 2

Lớp vô danh thực thi interface

➤ Ta có thể tạo lớp vô danh thực thi interface:

```
public interface Speakable {  
    void speak(); // nói chuyện  
  
    void whisper(); // nói thì thầm  
}  
  
class SpeakableExample {  
    public static void main(String[] args) {  
        // tạo Lớp vô danh từ interface  
        Speakable speakable = new Speakable() {  
            @Override  
            public void speak() {  
                System.out.println("Hello...");  
            }  
  
            @Override  
            public void whisper() {  
                System.out.println("I love you...");  
            }  
        };  
        // sử dụng đối tượng speakable  
        speakable.speak();  
        speakable.whisper();  
    }  
}
```



Loại 3

Lớp vô danh trong tham số của phương thức

- Ta có thể tạo lớp vô danh trong tham số của phương thức
- Đây là điều thường thấy trong lập trình giao diện, tạo các callback

Loại 3

➤ Ví dụ ta có interface và lớp sau:

```
public interface Speakable {  
    void speak(); // nói chuyện  
  
    void whisper(); // nói thì thầm  
}  
  
class Speaking {  
    private Speakable mSpeakable;  
  
    public Speaking(Speakable speakable) {  
        mSpeakable = speakable;  
    }  
  
    public void doSpeak() {  
        System.out.println("Inside class Speaking...");  
        mSpeakable.speak();  
        mSpeakable.whisper();  
    }  
}
```

Loại 3

- Ta tạo đối tượng vô danh làm đối số của phương thức:

```
class SpeakableExample {  
    public static void main(String[] args) {  
        // tạo đối tượng của Lớp với tham số là Lớp vô danh  
        Speaking speaking = new Speaking(new Speakable() {  
            @Override  
            public void speak() {  
                System.out.println("Hello...");  
            }  
  
            @Override  
            public void whisper() {  
                System.out.println("I love you...");  
            }  
        });  
        // thực hiện hành động  
        speaking.doSpeak();  
    }  
}
```

Lưu ý

- Một lớp vô danh chỉ có thể kế thừa một lớp khác tại một thời điểm
- Một lớp vô danh chỉ có thể implements 1 interface tại một thời điểm
- Lớp vô danh không thể chứa constructor vì nó không có tên
- Một lớp vô danh có thể truy cập các thành phần của lớp outer chứa nó
- Lớp vô danh không thể chứa interface, khối khởi tạo static, không truy cập được các biến cục bộ không phải final
- Lớp vô danh có thể chứa các hằng static, các trường, phương thức và các local class khác



Minh họa

➤ Thực hiện trong công cụ lập trình

Nội dung tiếp theo

Một số interface thường dùng