
HUST
ĐẠI HỌC BÁCH KHOA HÀ NỘI
HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY
ONE LOVE. ONE FUTURE.

1


ĐẠI HỌC
BÁCH KHOA HÀ NỘI
HANOI UNIVERSITY
OF SCIENCE AND TECHNOLOGY
Bài 8: Đa hình
ONE LOVE. ONE FUTURE.

2

Mục tiêu

- Giới thiệu về upcasting và downcasting
- Phân biệt liên kết tĩnh và liên kết động
- Nắm vững kỹ thuật đa hình
- Ví dụ và bài tập về các vấn đề trên với ngôn ngữ lập trình Java

3

Nội dung

1. Upcasting và Downcasting
2. Liên kết tĩnh và Liên kết động
3. Đa hình (Polymorphism)
4. Ví dụ và bài tập

4

Nội dung

1. **Upcasting và Downcasting**
2. Liên kết tĩnh và Liên kết động
3. Đa hình (Polymorphism)
4. Ví dụ và bài tập

5

1. Upcasting và Downcasting

- Chuyển đổi kiểu dữ liệu nguyên thủy
  - Java tự động chuyển đổi kiểu khi
    - Kiểu dữ liệu tương thích
    - Chuyển đổi từ kiểu hẹp hơn sang kiểu rộng hơn

```
int i;
double d = i;
```

  - Phải ép kiểu khi
    - Kiểu dữ liệu không tương thích
    - Chuyển đổi từ kiểu rộng hơn sang kiểu hẹp hơn

```
int i;
byte b = i;   byte b = (byte)i;
```

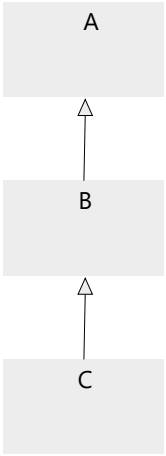

6

1. Upcasting và Downcasting

- Chuyển đổi kiểu dữ liệu tham chiếu
  - Kiểu dữ liệu tham chiếu có thể được chuyển đổi kiểu khi
    - Kiểu dữ liệu tham chiếu (lớp) *tương thích*
      - *Nằm trên cùng một cây phân cấp kế thừa*

```
A var1 = new B();

A var1 = new A();
C var2 = (C)var1;
```

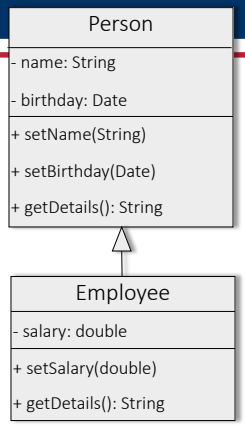
  - Hai loại chuyển đổi
    - Up-casting
    - Down-casting



1.1 Upcasting

- Upcasting: đi lên trên cây phân cấp thừa kế (moving up the inheritance hierarchy)
- Upcasting là khả năng nhìn nhận đối tượng thuộc lớp dẫn xuất như là một đối tượng thuộc lớp cơ sở.
- Tự động chuyển đổi kiểu

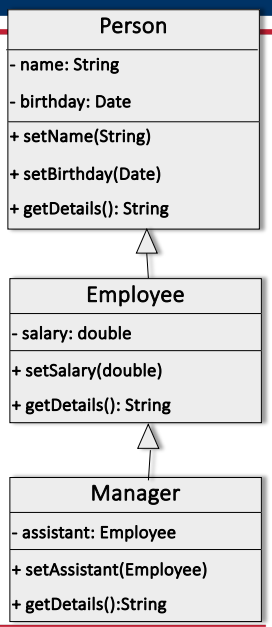
Ví dụ Upcasting




```
public class Test1 {
  public static void main(String arg[]) {
        Employee e = new Employee();
        Person p;
        p = e;
        p.setName("Hoa");
        p.setSalary(350000); // compile error
}
```


9

Ví dụ Upcasting

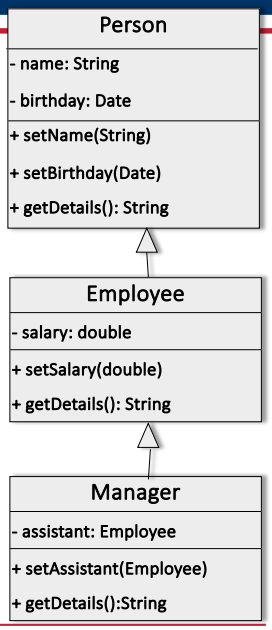



```
class Manager extends Employee {
  Employee assistant;
  // ...
  public void setAssistant(Employee e) {
        assistant = e;
  }
  // ...
}
public class Test2 {
  public static void main(String arg[]) {
        Manager junior, senior;
        // ...
        senior.setAssistant(junior);
  }
}
```


10

Ví dụ Upcasting

```
public class Test3 {
  String static teamInfo(Person p1,
                                  Person p2) {
        return "Leader: " + p1.getName() +
                ", member: " + p2.getName();
  }
  public static void main(String arg[]) {
        Employee e1, e2;
        Manager m1, m2;
        // ...
        System.out.println(teamInfo(e1, e2));
        System.out.println(teamInfo(m1, m2));
        System.out.println(teamInfo(m1, e2));
  }
}
```



1.2 Downcasting

- Down casting: đi xuống cây phân cấp thừa kế (move back down the inheritance hierarchy)
- Down casting là khả năng nhìn nhận một đối tượng thuộc lớp cơ sở như một đối tượng thuộc lớp dẫn xuất.
- Không tự động chuyển đổi kiểu
  → Phải ép kiểu.

Ví dụ Downcasting

```
public class Test2 {
   public static void main(String arg[]) {
      Employee e = new Employee();
      Person p = e; // upcasting
      Employee ee = (Employee) p; // downcasting
      Manager m = (Manager) ee; // run-time error

      Person p2 = new Manager();
      Employee e2 = (Employee) p2;

      Person p3 = new Employee();
      Manager e3 = (Manager) p3;
   }
}
```


Toán tử `instanceof`

- Kiểm tra xem một đối tượng có phải là thể hiện của một lớp nào đó không
- Trả về: true | false (nếu đối tượng là null thì trả về false)

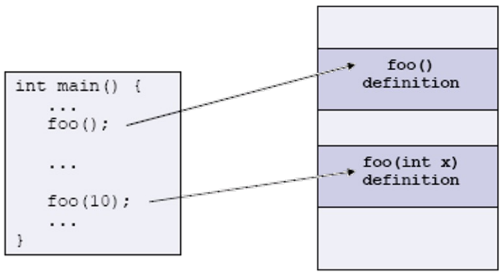
```
public class Employee extends Person {}
public class Student extends Person {}

public class Test{
  public doSomething(Person e) {
    if (e instanceof Employee) {...
    } else if (e instanceof Student) {...
    } else {...
    }
  }
}
```

Nội dung

1. Upcasting và Downcasting
2. **Liên kết tĩnh và Liên kết động**
3. Đa hình (Polymorphism)
4. Ví dụ và bài tập

Liên kết lời gọi hàm

- Liên kết lời gọi hàm (function call binding) là quy trình xác định khối mã hàm cần chạy khi một lời gọi hàm được thực hiện
  - Ví dụ xử lý liên kết lời gọi hàm trong C: đơn giản vì mỗi hàm có duy nhất một tên



Trong ngôn ngữ Hướng đối tượng

- Liên kết lời gọi phương thức (method call binding): quá trình liên kết lời gọi phương thức tới đoạn code thực thi phương thức
- Có 2 loại:
  - Liên kết tĩnh (static binding)
  - Liên kết động (dynamic binding)

2.1 Liên kết tĩnh

- Liên kết tại thời điểm biên dịch
  - Early Binding/Compile-time Binding
  - Lời gọi phương thức được quyết định khi biên dịch, do đó chỉ có một phiên bản của phương thức được thực hiện
  - Nếu có lỗi thì sẽ có lỗi biên dịch
  - Ưu điểm về tốc độ
- Ví dụ trong Java: các phương thức static

Ví dụ liên kết tĩnh trong Java

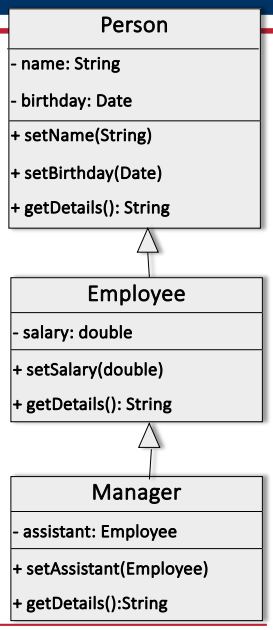
```
class Human {
    public static void walk() {
        System.out.println("Human walks");
    }
}
public class Boy extends Human {
    public static void walk() {
        System.out.println("Boy walks");
    }
    public static void main(String args[]) {
        // Reference is of Human type and object is Boy type
        Human obj1 = new Boy();

        // Reference is of Human type and object is Human type.
        Human obj2 = new Human();

        // Reference is of Human type and object is Human type.
        Boy obj3 = new Boy();

        obj1.walk();
        obj2.walk();
        obj3.walk();

        obj1 = obj3;
        obj1.walk();
    }
}
```


2.2 Liên kết động

- Lời gọi phương thức được quyết định khi thực hiện (run-time)
  - Late binding/Run-time binding
  - Phiên bản của phương thức phù hợp với đối tượng được gọi
  - Java trì hoãn liên kết phương thức cho đến thời gian chạy (run-time) - đây được gọi là liên kết động hoặc liên kết trễ
    - Java mặc định sử dụng liên kết động

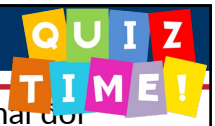
Ví dụ

```
public class Test {
  public static void main(String arg[]){
    Person p = new Person();
    // ...
    Employee e = new Employee();
    // ...
    Manager m = new Manager();
    // ...
    Person pArr[] = {p, e, m};
    for (int i=0; i< pArr.length; i++){
      System.out.println(
              pArr[i].getDetail());
    }
  }
}
```



21

Bài tập 1



- Giả sử lớp Sub kế thừa từ lớp cha Sandwich. Tạo hai đối tượng từ các lớp này:

```
Sandwich x = new Sandwich();
Sub y = new Sub();
```

- Phép gán nào sau đây là hợp lệ?

```
1. x = y;
2. y = x;
3. y = new Sandwich();
4. x = new Sub();
```


22

Nội dung

1. Upcasting và Downcasting
2. Liên kết tĩnh và Liên kết động
3. **Đa hình (Polymorphism)**
4. Ví dụ và bài tập

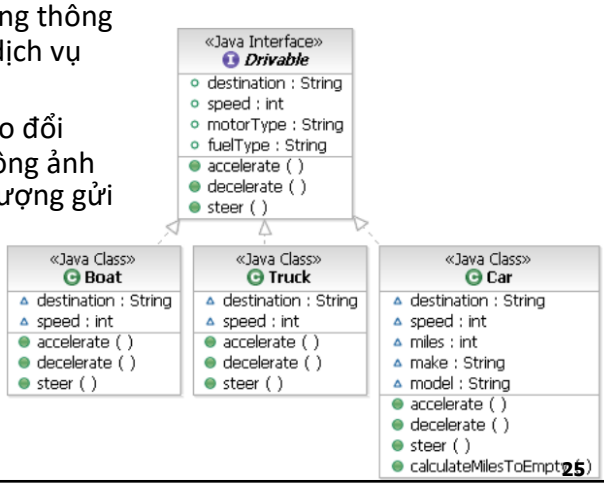
3. Đa hình (Polymophism)

- Ví dụ: Nếu đi du lịch, bạn có thể chọn ô tô, thuyền, hoặc máy bay
  - Dù đi bằng phương tiện gì, kết quả cũng giống nhau là bạn đến được nói cần đến
  - Cách thức đáp ứng các dịch vụ có thể khác nhau



3. Đa hình

- Các lớp khác nhau có thể đáp ứng danh sách các thông điệp giống nhau, vì vậy cung cấp các dịch vụ giống nhau
  - Cách thức đáp ứng thông điệp, thực hiện dịch vụ khác nhau
  - Chúng có thể tráo đổi cho nhau mà không ảnh hưởng đến đối tượng gửi thông điệp
  - → Đa hình



25

3. Đa hình

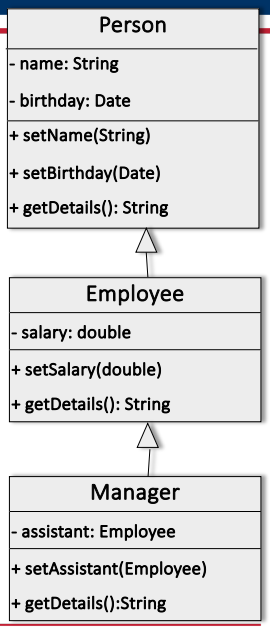
- Polymorphism: Nhiều hình thức thực hiện, nhiều kiểu tồn tại
  - Khả năng của một biến tham chiếu thay đổi hành vi theo đối tượng mà nó đang tham chiếu tới
- Đa hình trong lập trình
  - Đa hình phương thức:
    - Phương thức trùng tên, phân biệt bởi danh sách tham số.
  - Đa hình đối tượng
    - Nhìn nhận đối tượng theo nhiều kiểu khác nhau
    - Các đối tượng khác nhau cùng đáp ứng chung danh sách các thông điệp có giải nghĩa thông điệp theo cách thức khác nhau.

26

3. Đa hình

- Nhìn nhận đối tượng theo nhiều kiểu khác nhau → Upcasting và Downcasting

```
public class Test3 {
  public static void main(String
                          args[]){
    Person p1 = new Employee();
    Person p2 = new Manager();

    Employee e = (Employee) p1;
    Manager m = (Manager) p2;
  }
}
```



3. Đa hình

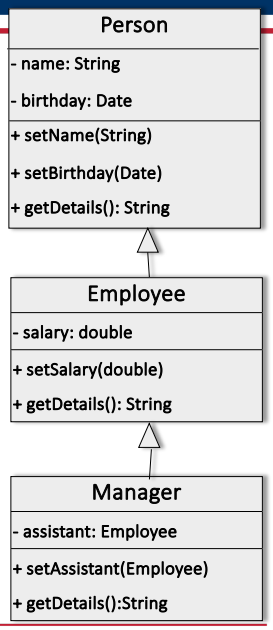
- Các đối tượng khác nhau giải nghĩa các thông điệp theo các cách thức khác nhau → Liên kết động
- Ví dụ:

```
Person p1 = new Person();
Person p2 = new Employee();
Person p3 = new Manager();
// ...
System.out.println(p1.getDetail());
System.out.println(p2.getDetail());
System.out.println(p3.getDetail());
```

3. Đa hình

- Ví dụ:
- Các đối tượng khác nhau giải nghĩa các thông điệp theo các cách thức khác nhau
  - Liên kết động (Java)

```
Person p1 = new Person();
Person p2 = new Employee();
Person p3 = new Manager();
// ...
System.out.println(p1.getDetail());
System.out.println(p2.getDetail());
System.out.println(p3.getDetail());
```



Toán tử instanceof

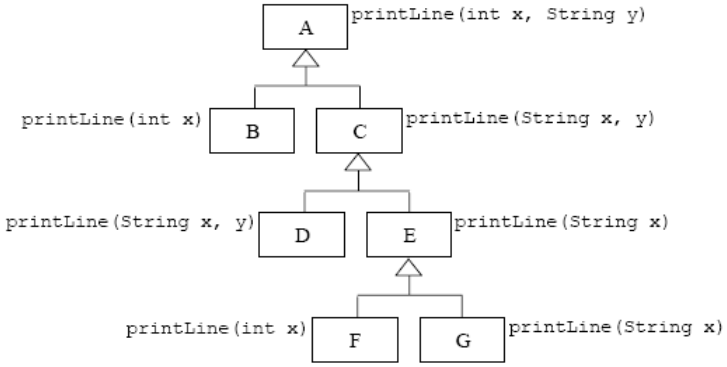
```
public class Employee extends Person {}
public class Student extends Person {}

public class Test{
  public doSomething(Person e) {
    if (e instanceof Employee) {...
    } else if (e instanceof Student) {... ){
    } else {...}
  }
}
```

Nội dung

1. Upcasting và Downcasting
2. Liên kết tĩnh và Liên kết động
3. Đa hình (Polymorphism)
4. **Ví dụ và bài tập**

Bài tập 2

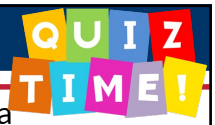
- Cho biểu đồ lớp:



Phương thức **`printLine()`** của lớp nào sẽ được sử dụng trong mỗi trường hợp dưới đây, biết rằng **z** là một đối tượng của lớp **F**? Giải thích ngắn gọn?

```
1. z.printLine(1)
2. z.printLine(2, "Object-Oriented Programming")
3. z.printLine("Java")
4. z.printLine("Object-Oriented Programming", "Java")
5. z.printLine("Object-Oriented Programming", 3)
```

Bài tập 3



- Những điều kiện nào trả về true? (Có thể xem Java documentation để biết các quan hệ thừa kế giữa các lớp) Biết rằng System.out là một đối tượng của lớp PrintStream.

```
1. System.out instanceof PrintStream
2. System.out instanceof OutputStream
3. System.out instanceof LogStream
4. System.out instanceof Object
5. System.out instanceof String
6. System.out instanceof Writer
```


Bài tập 4

- Kiểm tra các đoạn mã sau đây và vẽ sơ đồ lớp tương ứng

```
abstract public class Animal {
   abstract public void greeting();
}
public class Cat extends Animal {
   public void greeting() {
      System.out.println("Meow!");
   }
}
public class Dog extends Animal {
   public void greeting() {
      System.out.println("Woof!");
   }
   public void greeting(Dog another) {
      System.out.println("Wooooooooof!");
   }
}
public class BigDog extends Dog {
   public void greeting() {
      System.out.println("Woow!");
   }
   public void greeting(Dog another) {
      System.out.println("Wooooowwwww!");
   }
}
```

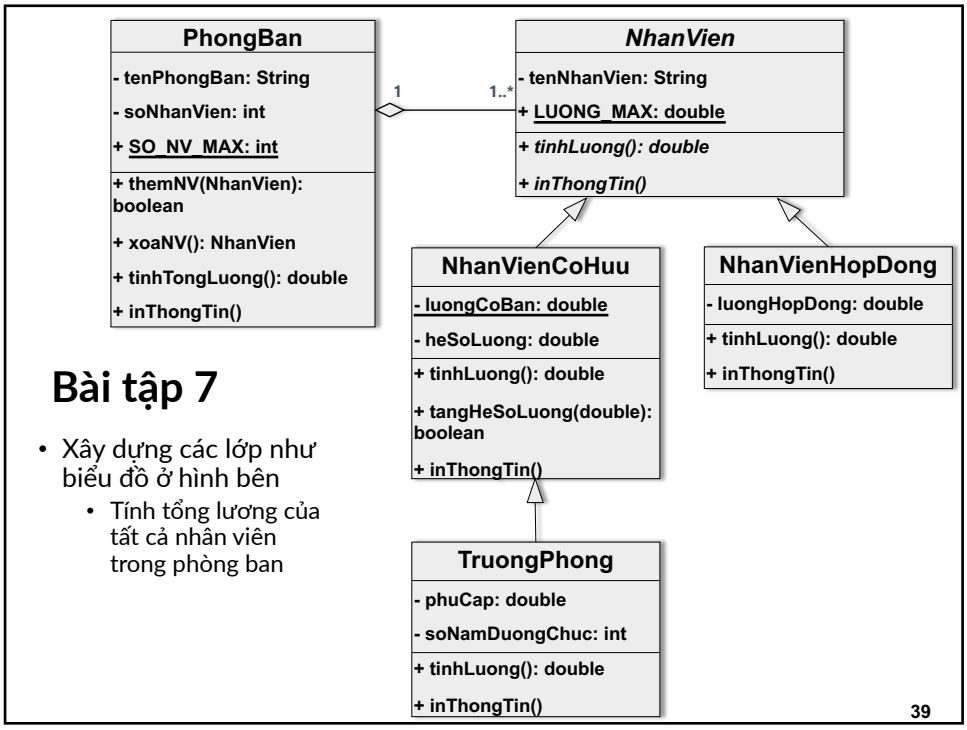
Bài tập 5

- Giải thích các đầu ra (hoặc các lỗi nếu có) cho chương trình thử nghiệm sau:

```
public class TestAnimal {
   public static void main(String[] args) {
      // Using the subclasses
      Cat cat1 = new Cat();
      cat1.greeting();
      Dog dog1 = new Dog();
      dog1.greeting();
      BigDog bigDog1 = new BigDog();
      bigDog1.greeting();

      // Using Polymorphism
      Animal animal1 = new Cat();
      animal1.greeting();
      Animal animal2 = new Dog();
      animal2.greeting();
      Animal animal3 = new BigDog();
      animal3.greeting();
      Animal animal4 = new Animal();

      // Downcast
      Dog dog2 = (Dog)animal2;
      BigDog bigDog2 = (BigDog)animal3;
      Dog dog3 = (Dog)animal3;
      Cat cat2 = (Cat)animal2;
      dog2.greeting(dog3);
      dog3.greeting(dog2);
      dog2.greeting(bigDog2);
      bigDog2.greeting(dog2);
      bigDog2.greeting(bigDog1);
   }
}
```


Bài tập 6

- Phân tích xây dựng các lớp như mô tả sau:
  - Hàng điện máy <mã hàng, tên hàng, nhà sản xuất, giá, thời gian bảo hành, điện áp, công suất>
  - Hàng sành sứ < mã hàng, tên hàng, nhà sản xuất, giá, loại nguyên liệu>
  - Hàng thực phẩm <mã hàng, tên hàng, nhà sản xuất, giá, ngày sản xuất, ngày hết hạn dùng>
- Viết chương trình tạo mỗi loại một mặt hàng cụ thể. Xuất thông tin về các mặt hàng này.


PhongBan
- tenPhongBan: String
- soNhanVien: int
+ SO_NV_MAX: int
+ themNV(NhanVien): boolean
+ xoaNV(): NhanVien
+ tinhTongLuong(): double
+ inThongTin()
1
1..*
NhanVien
- tenNhanVien: String
+ LUONG_MAX: double
+ tinhLuong(): double
+ inThongTin()
NhanVienCoHuu
- luongCoBan: double
- heSoLuong: double
+ tinhLuong(): double
+ tangHeSoLuong(double): boolean
+ inThongTin()
NhanVienHopDong
- luongHopDong: double
+ tinhLuong(): double
+ inThongTin()
TruongPhong
- phuCap: double
- soNamDuongChuc: int
+ tinhLuong(): double
+ inThongTin()
Bài tập 7
• Xây dựng các lớp như biểu đồ ở hình bên
• Tính tổng lương của tất cả nhân viên trong phòng ban
39

39


ĐẠI HỌC BÁCH KHOA HÀ NỘI
HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY
41

41

42