Đa hình

Nội dung

1. Lớp và phương thức trừu tượng
2. Mẫu thiết kế
3. Khái niệm về đa hình
4. Toán tử instanstof

Lớp và phương thức trừu tượng

Phương thức trừu tượng

- Là phương thức đặc biệt.
- Chỉ có chữ ký mà không có cài đặt cụ thể.
- Không thể khai báo là **final** hoặc **static**

Lớp và phương thức trừu tượng

Cú pháp

```
abstract <return_type> method_name([params]);
```

Ví dụ

```
abstract void calTotalPrice();
```

Lớp và phương thức trừu tượng

Lớp trừu tượng (Abstract Class) có đặc điểm:

- Không thể tạo thể hiện trực tiếp
- Có thể kế thừa
- Chưa đầy đủ, thường được sử dụng làm lớp lớp cha.

  Lớp con kế thừa nó sẽ hoàn thiện nốt.
- Chứa các phương thức trừu tượng

Lớp và phương thức trừu tượng

Cú pháp

```
abstract class class_name
```

Ví dụ

```
abstract class Shape{
    private String name;
    ...
    public abstract float calculateArea();
}
```

Giao diện

Giao diện (Interface) có đặc điểm:

- Một lớp có thể kế thừa (thực thi) nhiều giao diện cùng lúc
- Không thể tạo thể hiện trực tiếp từ giao diện
- Toàn bộ các phương thức trong giao diện không có nội dung.
- Các thuộc tính phải khai báo hằng (static hoặc final)

Giao diện

Cú pháp

```
interface interface_name
```

Ví dụ

```
interface IShape{
    private String name;
    ...
    public float calculateArea();
}
```

Giao diện

Thực thi một giao diện (interface)

- Có thể là Lớp hoặc là Lớp trừu tượng (Abstract Class)
- Bắt buộc phải cài đặt chi tiết toàn bộ các phương thức trong giao diện

Giao diện

Cú pháp

```
class class_name implements interface_name
```

Ví dụ

```
class Square implements IShape{
    private String name;
    public float calculateArea(){
        ...
    }
}
```

Giao diện vs Lớp trừu tượng

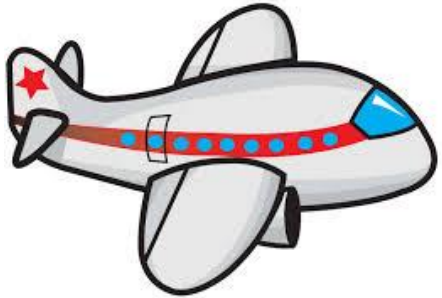
Giao diện	Lớp trừu tượng
Tất cả các phương thức đều không chứa nội dung	Có ít nhất 1 phương thức abstract, còn các phương thức còn lại vẫn có thể chứa nội dung
Chỉ có thể chứa các phương thức **public**	Có thể chứa các phương thức **protected** và **static**
Chỉ có thể chứa các thuộc tính hằng	Có thể chứa các thuộc tình final và non-final
Một lớp có thể thực thi nhiều giao diện	Một lớp chỉ có thể kế thừa từ 1 lớp trừu tượng

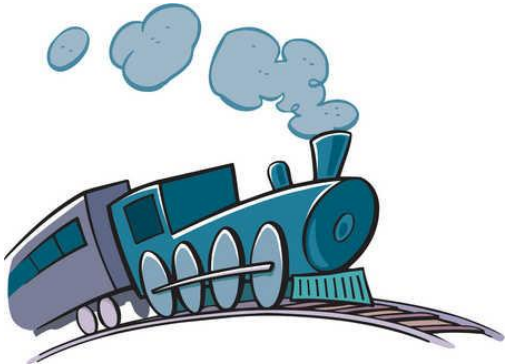
Đa hình - Polymophism

Đa hình

Ví dụ: Nếu đi du lịch từ Hà Nội - Hồ Chí Minh, bạn có thể chọn 1 loại phương tiện: ô tô, máy bay, tàu hoả

- Dù đi bằng phương tiện gì, kết quả cũng giống nhau là bạn sẽ đến được HCM
- Cách thức đáp ứng của các phương tiện khác nhau





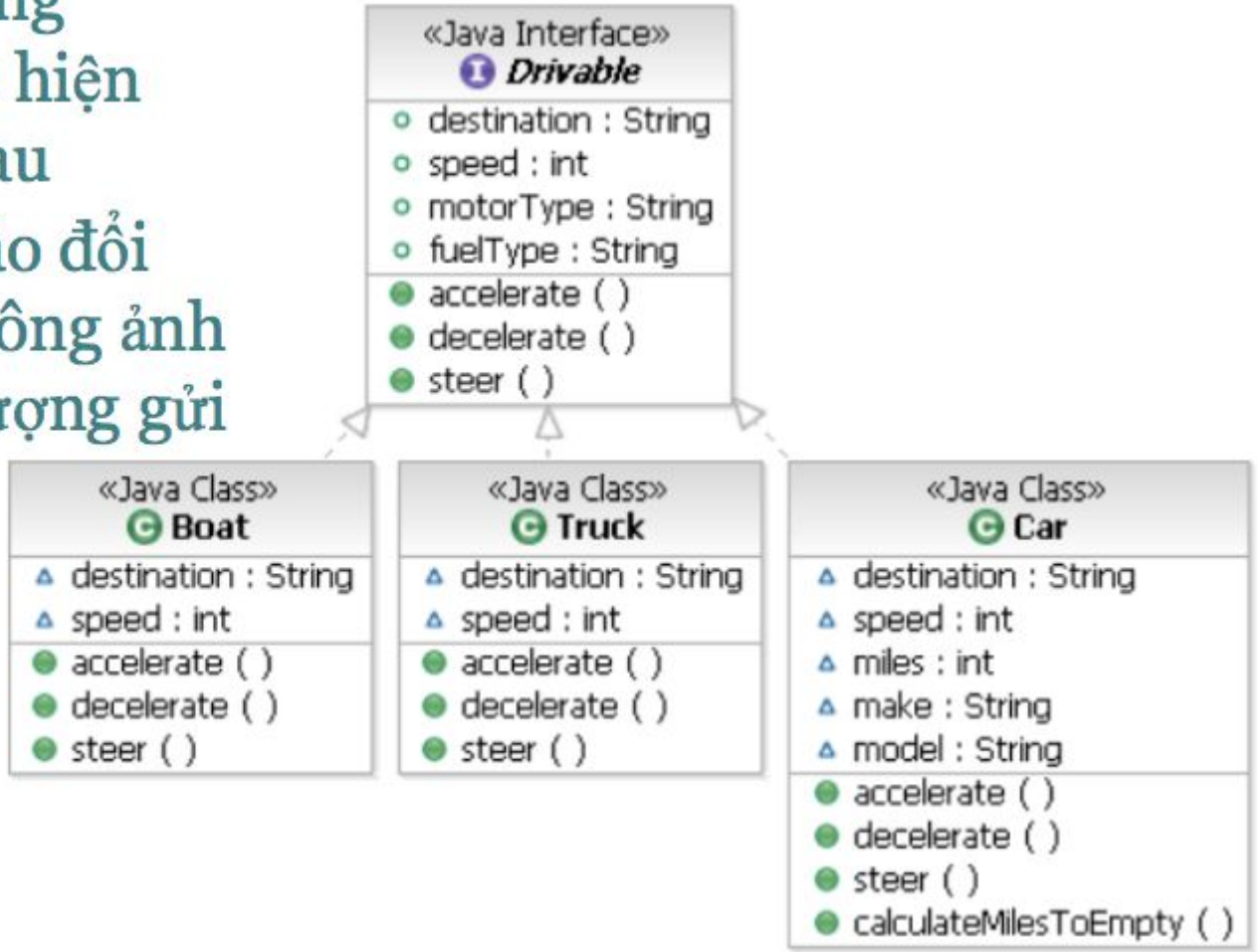
Đa hình

Định nghĩa

Các lớp khac 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



Đa hình

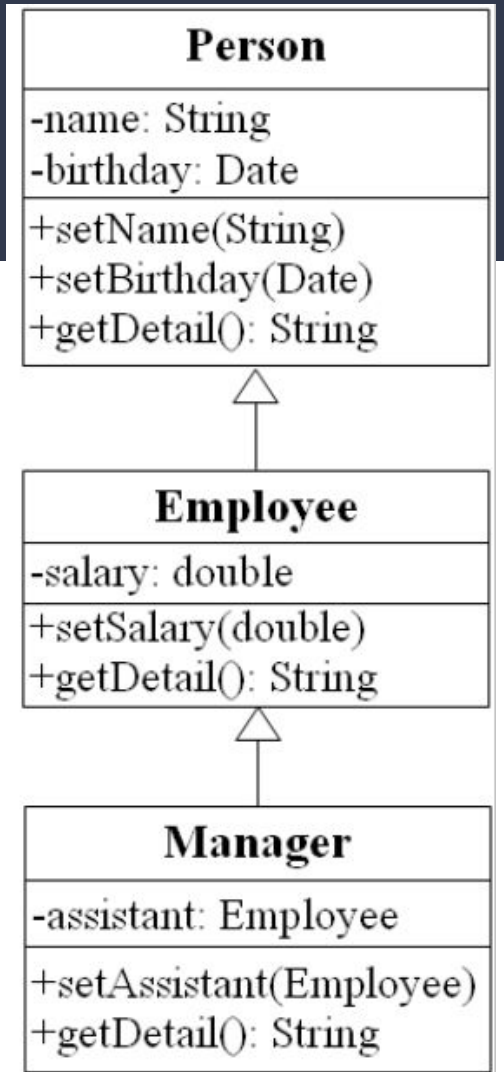
Đa hình trong lập trình

1. **Đa hình phương thức**
    - Phương thức trùng tê, phân biệt bởi danh sách tham số (overloading nạp chồng)
2. **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

Đa hình

Ví dụ

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



Ví dụ khác


```
class EmployeeList {
  Employee list[];
  ...
  public void add(Employee e) {...}
  public void print() {
    for (int i=0; i<list.length; i++) {
          System.out.println(list[i].getDetail());
    }
}
 ...
  EmployeeList list = new EmployeeList();
  Employee e1; Manager m1;
  ...
  list.add(e1); list.add(m1);
  list.print();
```

Toán tử `instanceof`

Kiểm tra xem đối tượng có là instance (thể hiện) của một kiểu cụ thể: lớp hoặc lớp con hoặc giao diện (interface) hay không.

Toán tử `instanceof`

Cú pháp

```
thể_hiện instanceof data_type
```

- Câu lệnh trả về giá trị true / false

Ví dụ

```
if (e instanceof Employee) {
    ...
}
```