



Java™

LẬP TRÌNH JAVA 1

BÀI 8: LỚP VÀ ĐỐI TƯỢNG

PHẦN 1

- ⊙ Hiểu rõ khái niệm lớp và đối tượng
- ⊙ Khai báo lớp và tạo được đối tượng
- ⊙ Khai báo trường, phương thức
- ⊙ Khai báo và sử dụng hàm tạo
- ⊙ Khai báo và sử dụng package
- ⊙ Sử dụng các đặc tả truy xuất

- ❑ Biểu diễn đối tượng trong thế giới thực
- ❑ Mỗi đối tượng được đặc trưng bởi các thuộc tính và các hành vi riêng của nó



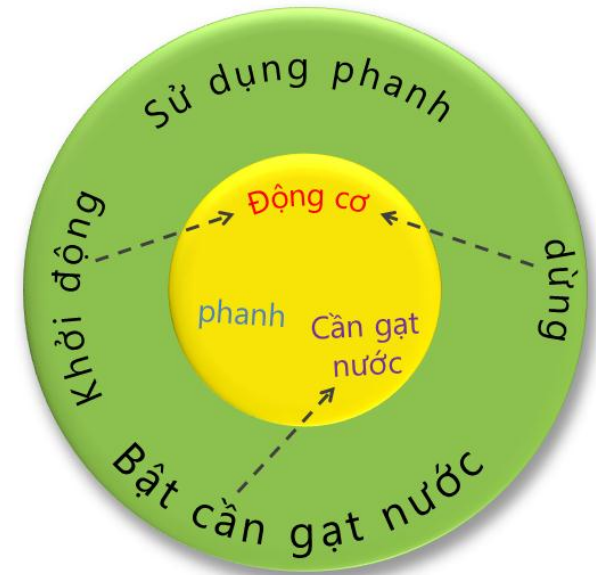
□ Đặc điểm

- Hãng sản xuất
- Model
- Năm
- Màu

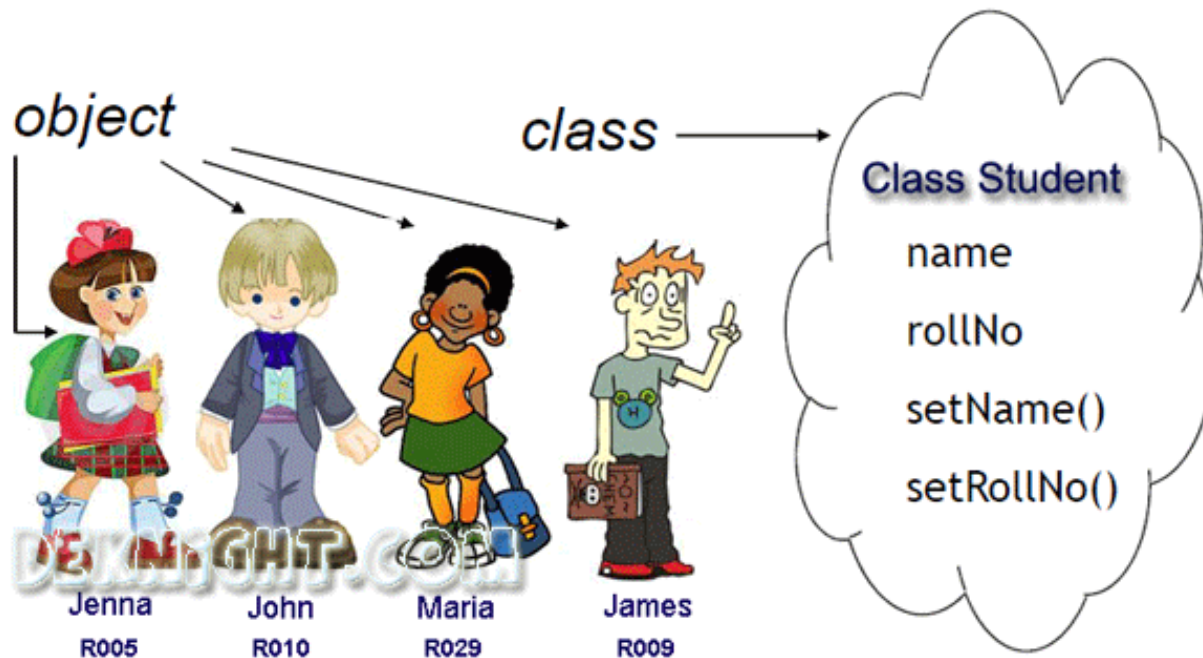


□ Hành vi (Ô tô có thể làm gì?)

- Khởi động
- Dừng
- Phanh
- Bật cần gạt nước



- ❑ Lớp là một khuôn mẫu được sử dụng để mô tả các đối tượng cùng loại.
- ❑ Lớp bao gồm các thuộc tính (trường dữ liệu) và các phương thức (hàm thành viên)



❑ Thuộc tính (field)

- ❖ Hãng sản xuất
- ❖ Model
- ❖ Năm
- ❖ Màu

Danh từ



❑ Phương thức (method)

- ❖ Khởi động()
- ❖ Dừng()
- ❖ Phanh()
- ❖ Bật cần gạt nước()

Động từ



```
class <<ClassName>>
```

```
{
```

```
<<type>> <<field1>>;
```

Khai báo các trường

```
...
```

```
<<type>> <<fieldN>>;
```

Khai báo các phương thức

```
<<type>> <<method1>>([parameters]) {
```

```
    // body of method
```

```
}
```

```
...
```

```
<<type>> <<methodN>>([parameters]) {
```

```
    // body of method
```

```
}
```

```
}
```

VÍ DỤ LỚP MÔ TẢ HÌNH TRÒN

```
package com.poly;
```

```
public class HìnhTron {  
    public double banKinh;  
    public double getChuVi() {  
        return 2*Math.PI*this.banKinh;  
    }  
    public double getDienTich() {  
        return Math.PI*Math.pow(this.banKinh, 2);  
    }  
    public void print() {  
        System.out.println("> Bán kính: " + this.banKinh);  
        System.out.println("> Chu vi: " + this.getChuVi());  
        System.out.println("> Diện tích: " + this.getDienTich());  
    }  
}
```

- ✓ Lớp HìnhTron
- ✓ Trường banKinh
- ✓ 3 phương thức
 - getChuVi()
 - getDienTich()
 - print()

- ❑ Đoạn mã sau sử dụng lớp `HinhTron` để tạo một hình tròn sau đó gán dữ liệu cho trường `banKinh` và gọi các phương thức `print()`.

```
public static void main(String[] args) {  
    HinhTron ht = new HinhTron();  
    ht.banKinh = 100;  
    ht.print();  
}
```

- ❑ Chú ý:

- ❖ Toán tử **new** được sử dụng để tạo đối tượng
- ❖ Biến `ht` chứa tham chiếu tới đối tượng
- ❖ Sử dụng dấu chấm (.) để truy xuất các thành viên của lớp (trường và phương thức).



DEMO

Tạo lớp mô tả sinh viên bao gồm họ
tên, điểm và các phương thức nhập,
xuất và xếp loại học lực



- ❑ Phương thức hay còn gọi là hàm thành viên
- ❑ Khai báo phương thức giống hệt khai báo hàm tiện ích trước đây, chỉ khác ở chỗ là không sử dụng **static**
- ❑ Cú pháp

```
<<kiểu trả về>> <<tên phương thức>> ([danh sách tham số])  
{  
    // thân phương thức  
}
```

```
public class Employee{  
    public String fullname;  
    public double salary;
```

```
    public void input(){...}  
    public void output(){...}
```

Kiểu trả về là **void** nên thân phương thức **không chứa lệnh return giá trị**

```
    public void setInfo(String fullname, double salary) {  
        this.fullname = fullname;  
        this.salary = salary;  
    }
```

Kiểu trả về là **double** nên thân phương thức phải chứa lệnh **return số thực**

```
    public double incomeTax(){  
        if(this.salary < 5000000){  
            return 0;  
        }  
        double tax = (this.salary - 5000000) * 10/100;  
        return tax;  
    }  
}
```

- ❑ Hàm tạo là một phương thức đặc biệt được sử dụng để tạo đối tượng.
- ❑ Đặc điểm của hàm tạo
 - ❖ Tên trùng với tên lớp
 - ❖ Không trả về giá trị
- ❑ Ví dụ

Lớp

```
public class ChuNhat{  
    double dai, rong;  
    public ChuNhat(double dai, double rong){  
        this.dai = dai;  
        this.rong = rong;  
    }  
}
```

Đối tượng

```
ChuNhat cn1 = new ChuNhat(20, 15);  
ChuNhat cn2 = new ChuNhat(50, 25);
```

- ❑ Trong một lớp có thể định nghĩa nhiều hàm tạo khác tham số. Mỗi hàm tạo cung cấp 1 cách tạo đối tượng.
- ❑ Nếu không khai báo hàm tạo thì Java tự động cung cấp hàm tạo mặc định (không tham số)

```
public class ChuNhat {  
    public double dai, rong;  
    public ChuNhat(double dai, double rong) {  
        this.dai = dai;  
        this.rong = rong;  
    }  
    public ChuNhat(double canh) {  
        this.dai = canh;  
        this.rong = canh;  
    }  
}
```

```
public static void main(String[] args) {  
    ChuNhat cn = new ChuNhat(20, 15);  
    ChuNhat vu = new ChuNhat(30);  
}
```




DEMO



Hiện thực hóa ví dụ ở slide trước

- ❑ **this** được sử dụng để đại diện cho đối tượng hiện tại.
- ❑ **this** được sử dụng trong lớp để tham chiếu tới các thành viên của lớp (field và method)
- ❑ Sử dụng **this.field** để phân biệt field với các biến cục bộ hoặc tham số của phương thức

```
public class MyClass{  
    int field;  
    void method(int field){  
        this.field = field;  
    }  
}
```



Trường

Tham số

SinhVien

+ hoTen: String
+ diemTB: double

+ xepLoai(): String
+ xuất(): void
+ nhập(): void

+ SinhVien()
+ SinhVien(hoTen, diemTB)

DEMO



Xây dựng lớp mô tả sinh viên như mô hình trên.
Trong đó nhập() cho phép nhập họ tên và điểm từ bàn phím; xuất() cho phép xuất họ tên, điểm và học lực ra màn hình; xepLoai() dựa vào điểm để xếp loại học lực
Sử dụng 2 hàm tạo để tạo 2 đối tượng sinh viên



Java™

LẬP TRÌNH JAVA 1

BÀI 8: LỚP VÀ ĐỐI TƯỢNG

PHẦN 2

- ❑ Package được sử dụng để chia các class và interface thành từng gói khác nhau.
 - ❖ Việc làm này tương tự quản lý file trên ổ đĩa trong đó class (như file) và package (như folder)
- ❑ Mục đích sử dụng package là để quản lý tên class và interface
 - ❖ Trong một package không có 2 class hoặc interface trùng tên
- ❑ Ví dụ: lớp MyClass thuộc gói com.poly

```
package com.poly;  
public class MyClass{...}
```

- ❑ Trong Java có rất nhiều gói được sử dụng để phân các class và interface theo chức năng
 - ❖ java.util: chứa các lớp tiện ích
 - ❖ java.io: chứa các lớp vào/ra dữ liệu
 - ❖ java.lang: chứa các lớp thường dùng...
- ❑ Bạn có thể sử dụng package để phân class hoặc interface theo ý riêng
 - ❖ Dự án
 - ❖ Chức năng
 - ❖ Vùng miền...

- ❑ Lệnh import được sử dụng để chỉ ra lớp hoặc interface đã được định nghĩa trong một package.
- ❑ Các lớp trong các gói sau sẽ tự động import
 - ❖ Các lớp cùng gói với lớp sử dụng chúng
 - ❖ java.lang sẽ được import
- ❑ Cách thức import
 - ❖ **com.poly.MyClass**
 - Chỉ import class có tên MyClass
 - ❖ **com.poly.***
 - Import tất cả các class và interface có trong gói com.poly

❑ Xét ví dụ sau ta thấy

- ❖ Lớp HelloWorld sử dụng các lớp MyClass và Scanner
- ❖ Lớp HelloWorld không cùng gói với lớp MyClass

❑ Ví vậy phải cần import các lớp này một cách tường minh

```
package com.polyhcm;  
import com.poly.MyClass;  
import java.util.Scanner;  
  
public class HelloWorld{  
    public static void main(String[] args){  
        MyClass obj = new MyClass();  
        Scanner scanner = new Scanner(System.in);  
    }  
}
```

❑ Đặc tả truy xuất được sử dụng để định nghĩa khả năng cho phép truy xuất đến các thành viên của lớp. Trong java có 4 đặc tả khác nhau:

❖ **private**: chỉ được phép sử dụng nội bộ trong class

❖ **public**: công khai hoàn toàn

❖ **{default}**:

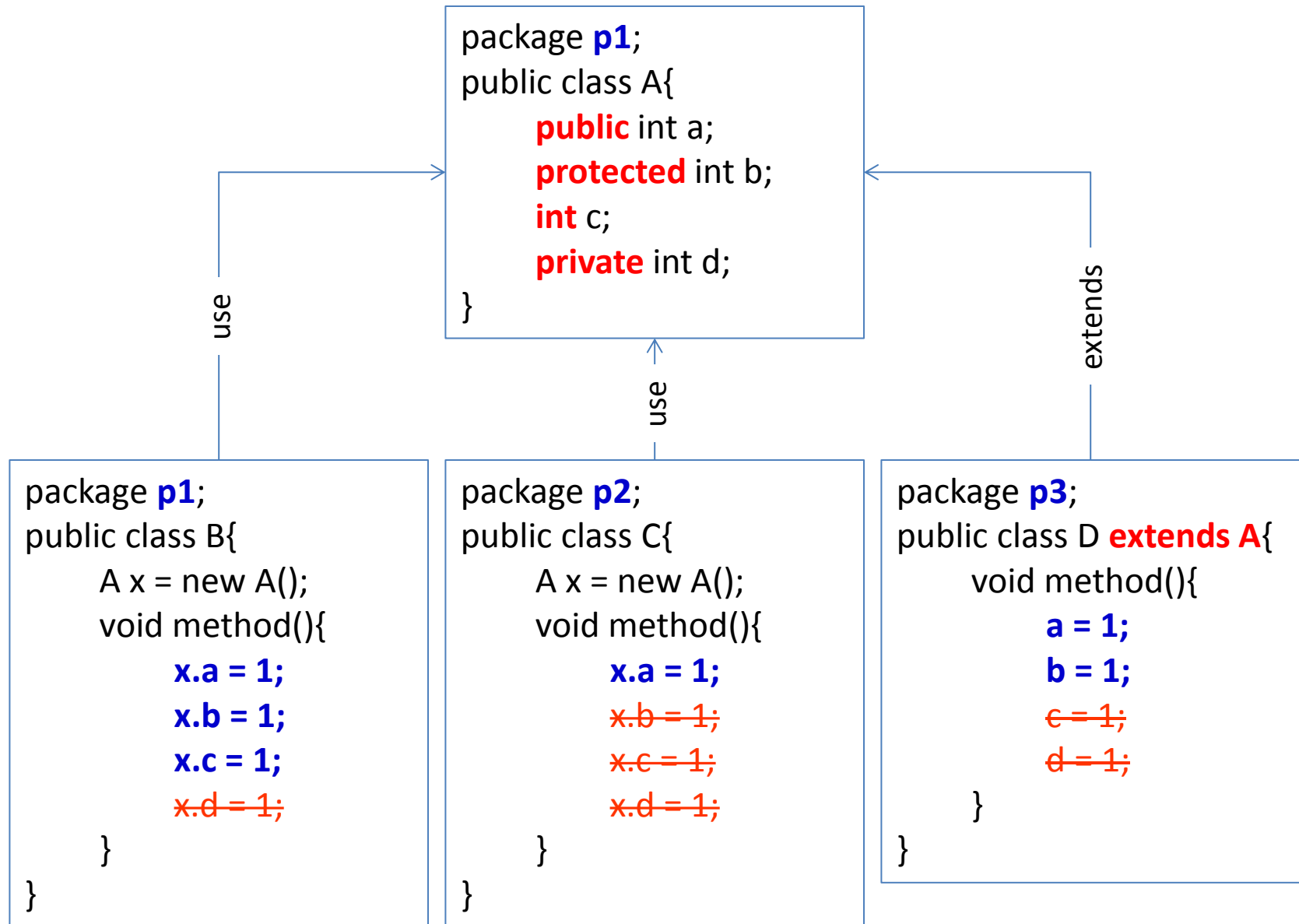
➤ Là public đối với các lớp truy xuất cùng gói

➤ Là private với các lớp truy xuất khác gói.

❖ **protected**: tương tự {default} nhưng cho phép kế thừa dù lớp con và cha khác gói.

❑ Mức độ che dấu tăng dần theo chiều mũi tên

public → **protected** → **{default}** → **private**



- ❑ Encapsulation là tính che dấu trong hướng đối tượng.
 - ❖ Nên che dấu các trường dữ liệu
 - ❖ Sử dụng phương thức getter/setter để truy xuất các trường dữ liệu
- ❑ Mục đích của che dấu
 - ❖ Bảo vệ dữ liệu
 - ❖ Tăng cường khả năng mở rộng

- ❑ Xét lớp SinhVien và công khai hoTen và diem như sau

```
public class SinhVien{  
    public String hoTen;  
    public double diem;  
}
```

```
public class MyClass{  
    public static void main(String[] args){  
        SinhVien sv = new SinhVien();  
        sv.hoTen = "Nguyễn Văn Tèo";  
        sv.diem = 20.5;  
    }  
}
```

- ❑ Khi sử dụng người dùng có thể gán dữ liệu cho các trường một cách tùy tiện
- ❑ Điều gì sẽ xảy ra nếu điểm **hợp lệ chỉ từ 0 đến 10**

- ❑ Để che dấu hoàn toàn một trường, sử dụng đặc tả truy xuất **private**.

private double diem;

- ❑ Bổ sung các phương thức getter và setter để đọc ghi các trường đã che dấu

```
public void setDiem(double diem){  
    this.diem = diem;  
}  
  
public String getDiem() {  
    return this.diem;  
}
```

```
public class SinhVien{
    private String hoTen;
    private double diem;
    public void setHoTen(String hoTen){
        this.hoTen = hoTen;
    }
    public String getHoTen(){
        return this.hoTen;
    }
    public void setDiem(double diem){
        if(diem < 0 || > 10){
            System.out.println("Điểm không hợp lệ");
        }
        else{
            this.diem = diem;
        }
    }
    public String getDiem(){
        return this.diem;
    }
}
```

❑ Chỉ cần thêm mã vào phương thức setDiem() để có những xử lý khi dữ liệu không hợp lệ

```
public class MyClass{
    public static void main(String[] args){
        SinhVien sv = new SinhVien();
        sv.setHoTen("Nguyễn Văn Tèo");
        sv.setDiem(20);
    }
}
```

- ☑ Hiểu rõ khái niệm lớp và đối tượng
- ☑ Khai báo lớp và tạo được đối tượng
- ☑ Khai báo trường, phương thức
- ☑ Khai báo và sử dụng hàm tạo
- ☑ Khai báo và sử dụng package
- ☑ Sử dụng các đặc tả truy xuất

