Bộ môn Công nghệ Phần mềm
Viện CNTT & TT
Trường Đại học Bách Khoa Hà Nội

# LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG
## Bài 05. Kết tập và kế thừa

## Mục tiêu bài học

- Giải thích về khái niệm tái sử dụng mã nguồn
- Chỉ ra được bản chất, mô tả các khái niệm liên quan đến đến kết tập và kế thừa
- So sánh kết tập và kế thừa
- Biểu diễn được kết tập và kế thừa trên UML
- Giải thích nguyên lý kế thừa và thứ tự khởi tạo, hủy bỏ đối tượng trong kế thừa
- Áp dụng các kỹ thuật, nguyên lý về kết tập và kết thừa trên ngôn ngữ lập trình Java



## Nội dung

1. Tái sử dụng mã nguồn
2. Kết tập (Aggregation)
3. Kế thừa (Inheritance)

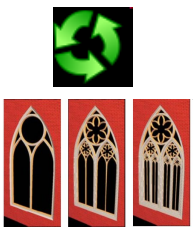


## Nội dung

1. Tái sử dụng mã nguồn
2. Kết tập (Aggregation)
3. Kế thừa (Inheritance)



## 1. Tái sử dụng mã nguồn (Re-usability)





## 1. Tái sử dụng mã nguồn (2)

- Các cách sử dụng lại lớp đã có:

## Ưu điểm của tái sử dụng mã nguồn

7

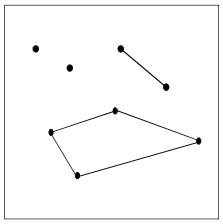
## Nội dung

1. Tái sử dụng mã nguồn
2. Kết tập (Aggregation)
3. Kế thừa (Inheritance)

8

## 2. Kết tập

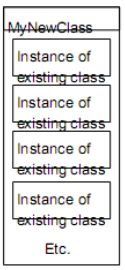
- Ví dụ:



9

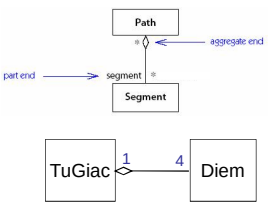
## 2.1. Bản chất của kết tập

- Lớp toàn thể chứa đối tượng của lớp thành phần



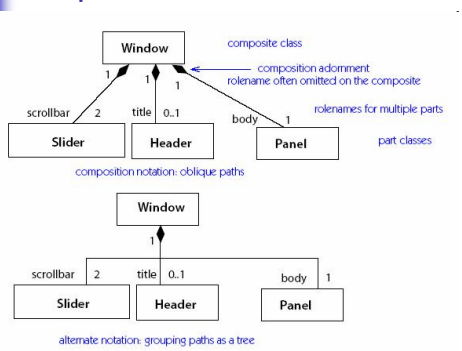

10

## 2.2. Biểu diễn kết tập bằng UML




11

## Ví dụ




12

## 2.3. Minh họa trên Java

```
class Diem {
  private int x, y;
  public Diem(){}
  public Diem(int x, int y) {
     this.x = x; this.y = y;
  }
  public void setX(int x){ this.x = x; }
  public int getX() { return x; }
  public void hienThiDiem(){
      System.out.print("(" + x + ", "
                          + y + ")");
  }
}
```

13

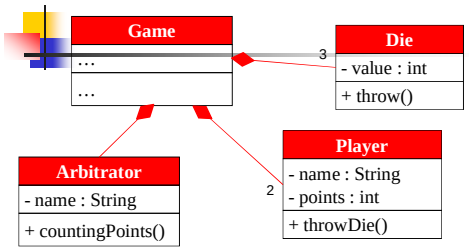
```
class TuGiac {
  private Diem d1, d2;
  private Diem d3, d4;
  public TuGiac(Diem p1, Diem p2,
                    Diem p3, Diem p4){
    d1 = p1; d2 = p2; d3 = p3; d4 = p4;
  }
  public TuGiac(){
    d1 = new Diem();      d2 = new Diem(0,1);
    d3 = new Diem (1,1); d4 = new Diem (1,0);
  }
  public void printTuGiac(){
    d1.printDiem();  d2.printDiem();
    d3.printDiem();  d4.printDiem();
    System.out.println();
  }
}
```

14

## Ví dụ khác về Kết tập

- Một trò chơi gồm 2 đối thủ, 3 quân súc sắc và 1 trọng tài.

15




16

## 2.4. Thứ tự khởi tạo trong kết tập

17

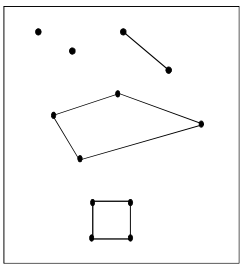
## Nội dung

1. Tái sử dụng mã nguồn
2. Kết tập (Aggregation)
3. Kế thừa (Inheritance)

18

## 3.1. Tổng quan về kế thừa

- Ví dụ:



19

## 3.1.1. Bản chất kế thừa

- Kế thừa (Inherit, Derive)

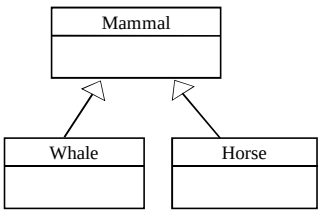
20

## 3.1.1. Bản chất kế thừa (2)

- Lớp con
  - Là một loại (is-a-kind-of) của lớp cha

21

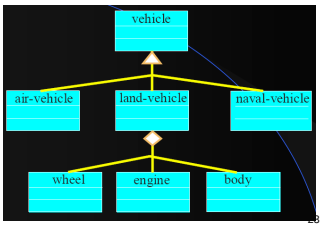
## 3.1.2. Biểu diễn kế thừa trong UML




22

## 3.1.3. Kết tập và kế thừa

- So sánh kết tập và kế thừa?
  - Giống nhau?
  - Khác nhau?



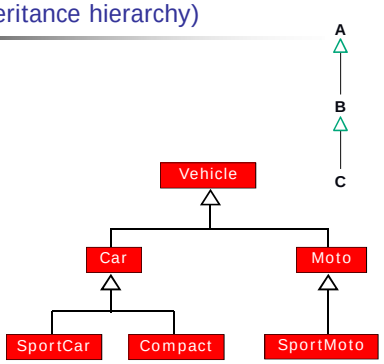

23

## Phân biệt kế thừa và kết tập

Kế thừa Kết tập

24

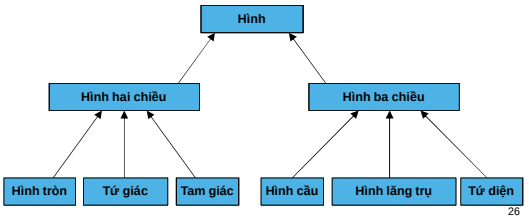
## 3.1.4. Cây phân cấp kế thừa (Inheritance hierarchy)




25

## 3.1.4. Cây phân cấp kế thừa (2)

- Siblings



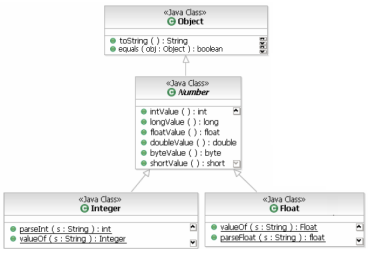

26

## Lớp Object

- Trong gói java.lang

27

## Lớp Object (2)




## 3.2. Nguyên lý kế thừa

| | public | Không có | protected | private |
|---|---|---|---|---|
| Cùng lớp | | | | |
| Lớp con cùng gói | | | | |
| Lớp con khác gói | | | | |
| Khác gói, non-inher | | | | |

29

## 3.2. Nguyên lý kế thừa (2)

- Các trường hợp không được phép kế thừa?

30

## 3.3. Cú pháp kế thừa trên Java

- Cú pháp kế thừa trên Java:
- Ví dụ:

```
class HinhVuong extends TuGiac {
  ...
}
```

31

```
public class TuGiac {
  protected Diem d1, d2, d3, d4;
  public void setD1(Diem _d1) {d1=_d1;}
  public Diem getD1(){return d1;}
  public void printTuGiac(){...}
  …
}

public class HinhVuong extends TuGiac {
  public HinhVuong(){
      d1 = new Diem(0,0); d2 = new Diem(0,1);
      d3 = new Diem(1,0); d4 = new Diem(1,1);
  }
}
public class Test{
  public static void main(String args[]){
      HinhVuong hv = new HinhVuong();
      hv.printTuGiac();
  }
}
```

32

```
public class TuGiac {
  protected Diem d1, d2, d3, d4;
  public void printTuGiac(){...}
  public TuGiac(){...}
  public TuGiac(Diem d1, Diem d2,
                Diem d3, Diem d4) { ...}
}
public class HinhVuong extends TuGiac {
  public HinhVuong(){ super(); }
  public HinhVuong(Diem d1, Diem d2,
                   Diem d3, Diem d4){
      super(d1, d2, d3, d4);
  }
}
public class Test{
  public static void main(String args[]){
      HinhVuong hv = new HinhVuong();
      hv.printTuGiac();
  }
}
```

33

## Ví dụ 2

```
class Person {
  private String name;
  private Date bithday;
  public String getName() {return name;}
  ...
}
class Employee extends Person {
  private double salary;
  public boolean setSalary(double sal){
   salary = sal;
   return true;
  }
  public String getDetail(){
   String s = name+", "+birthday+", "+salary;
  }
}
```

34

## Ví dụ 2

```
class Person {
  protected String name;
  protected Date bithday;
  public String getName() {return name;}
  ...
}
class Employee extends Person {
  private double salary;
  public boolean setSalary(double sal){
   salary = sal;
   return true;
  }
  public String getDetail(){
   String s = name+", "+birthday+", "+salary;
  }
}
```

35

## Ví dụ 2 (tiếp)

```
public class Test {
  public static void main(String args[]){
      Employee e = new Employee();
      e.setName("John");
      e.setSalary(3.0);
  }
}
```

36

## Ví dụ 3 – Cùng gói

```
public class Person {
  Date birthday;
  String name;
  ...
}
public class Employee extends Person {
  ...
  public String getDetail() {
     String s;
     String s = name + "," + birthday;
     s += ", " + salary;
     return s;
  }
}
```

37

## Ví dụ 3 – Khác gói

```
package abc;
public class Person {
  protected Date birthday;
  protected String name;
  ...
}

import abc.Person;
public class Employee extends Person {
  ...
  public String getDetail() {
     String s;
     s = name + "," + birthday + "," + salary;
     return s;
  }
}
```

38

## 3.4. Khởi tạo và huỷ bỏ đối tượng

- Khởi tạo đối tượng:
- Hủy bỏ đối tượng:

39

## 3.4.1. Tự động gọi constructor của lớp cha

```
public class TuGiac {
  protected Diem d1, d2;
  protected Diem d3, d4;
  public TuGiac(){
   System.out.println
      ("Lop cha TuGiac()");
  }
  //…
}
public class HinhVuong
      extends TuGiac {
  public HinhVuong(){
   //Tu dong goi TuGiac()
   System.out.println
   ("Lop con HinhVuong()");
  }
}
```

```
public class Test {
  public static void
  main(String arg[])
  {
   HinhVuong hv =
      new HinhVuong();
  }
}
```

40

## 3.4.2. Gọi trực tiếp constructor của lớp cha

- Bắt buộc nếu lớp cha không có phương thức khởi tạo mặc định

41

## Ví dụ

```
public class TuGiac {
  protected Diem d1, d2;
  protected Diem d3, d4;
  public TuGiac(Diem d1,
  Diem d2, Diem d3, Diem d4){
   System.out.println("Lop cha
    TuGiac(d1, d2, d3, d4)");
   this.d1 = d1; this.d2 = d2;
   this.d3 = d3; this.d4 = d4;
  }
}
public class HinhVuong extends TuGiac {
  public HinhVuong(){
    System.out.println
   ("Lop con HinhVuong()");
  }
}
```

```
public class Test {
  public static
  void main(String
  arg[])
  {
   HinhVuong hv =
  new

  HinhVuong();
   }
}
```

42