

Polymorphism

Hsuan-Tien Lin

Department of CSIE, NTU

OOP Class, April 26-27, 2010

One Thing, Many Shapes.....

One Variable, Many Values

short

```
1 char a;  
2 switch(a){  
3 case 1:  
4     return 1 * 1;  
5 case 2:  
6     return 2 * 2;  
7 case 3:  
8     return 3 * 3;  
9 ...  
10 case 127:  
11     return 127 * 127;  
12 }
```

*} return a*a;*

- better ways?
- mechanism? raw memory interpretation

One Class, Many Instances

for(if s = 100 [i])

```
1 Student s;  
2 if (s equals student1) ↗  
3   show(score1);  
4 else if (s equals student2) ↗  
5   show(score2);  
6 ...  
7 else if (s equals student100)  
8   show(score100);
```

```
class Student {  
  int score;  
  ↵  
  show(s.score);  
}
```

- better ways?
- mechanism? data abstraction

One Method Name, Many Parameter Lists

```
1 //no overloading
2 System.out.println("abc");
3 System.out.println(3);
4 System.out.println(4.0);
5 //overloading
6 System.out.print("abc");
7 System.out.print(3);
8 System.out.print(4.0);
```

- mechanism? signature by name + parameter types

A Twist in Method Overloading

```
1 // overloading
2 System.out.print("abc");
3 System.out.print(3);
4 System.out.print(4.0);
5 //a twist (of course, not exactly workable)
6 String("abc").print();
7 Integer(3).print();
8 Double(4.0).print();
```

- mechanism? just write print() for every class (we'll see)

Polymorphic Behavior

- all cases above: some polymorphic behavior
- **BUT** almost no one calls them real “polymorphism”

Why Not?

Polymorphic Behavior on Known Stuff

- polymorphic behavior of **known** primitive-type variables
- polymorphic behavior of **known** classes (extended types)
- polymorphic behavior of **known** parameter types

Unknown Stuff: Future Extensions

Backward Compatibility

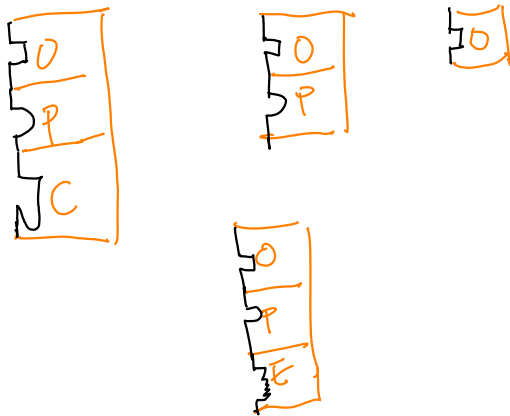
inheritance hierarchy

Forward Advance

newly added variables/methods

Polymorphism of Instance Content

one **advanced** content, many **compatible** ways to access



Polymorphism of Instance References

one **compatible** reference, many **advanced** contents to point to



Reference Upcast versus Reference Downcast

```
1 CSIEProfessor c = new CSIEProfessor();  
2 Professor p = c;  
3 // Professor p = new CSIEProfessor();
```

Upcast

simple (backward compatibility)

```
1 CSIEProfessor c = new CSIEProfessor();  
2 Professor p = c;  
3 CSIEProfessor c2 = (CSIEProfessor)p;
```

Downcast

need to check whether content fits (forward advance)

need RTTI (run-time type information/identification) to make downcast work (**where did we see it?**)

Content/Reference Polymorphism: Summary

backward compatibility handled

forward advance: only via downcast (RTTI)

simpler mechanism for forward advance?