



Introduction to Objects

- Short Introduction to the Concepts of Object-Oriented Programming
- Overview over the most important terminology

OOAD 1998/99

Claudia Niederée, Joachim W. Schmidt
Software Systems Institute

c.niederee@tu-harburg.de <http://www.sts.tu-harburg.de>

The Progress of Abstraction

- Problem space & solution space
- Alan Kay's 5 rules for a pure OOP language
 - 1) Everything is an object
 - 2) A program is a bunch of objects telling each other what to do by sending messages
 - 3) Each object has its own memory made up of other objects
 - 4) Every object has a type
 - 5) All objects of a particular type can receive the same messages.

An Object has an Interface

Type name

Light

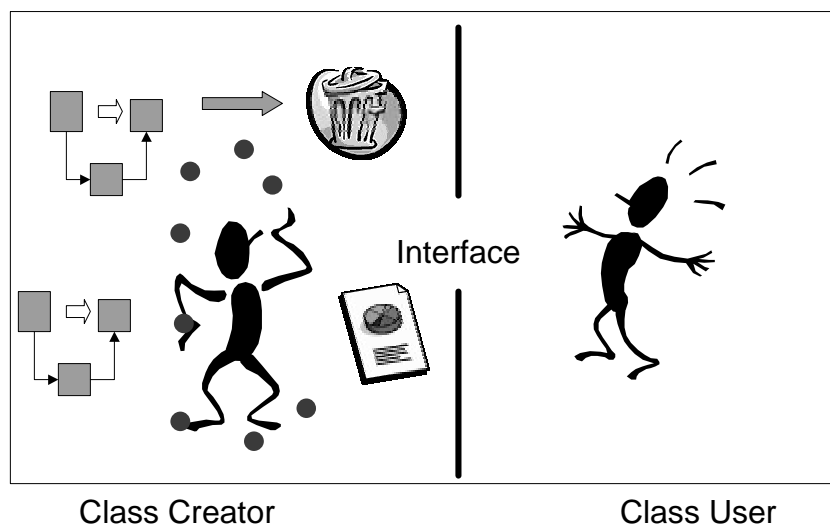
Interface

```
on()
off()
brighten()
dim()
```



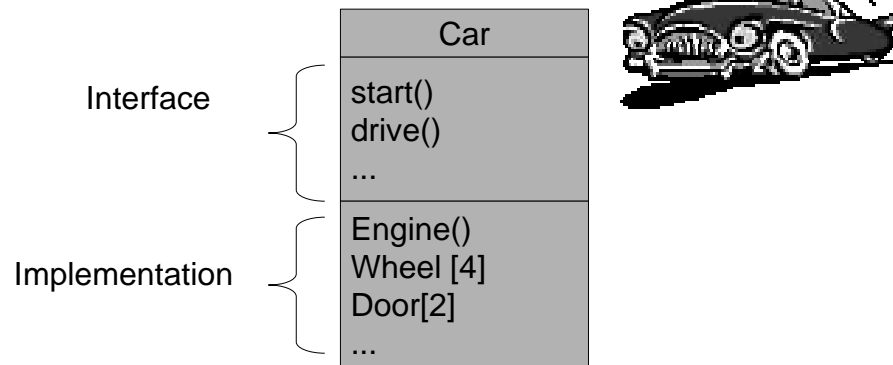
- Object : Characteristics and Behavior
 - Object Creation: `Light lt = new Light();`
 - Message: `lt.on();`
- That's programming with objects!

The Hidden Implementation



Reusing the Implementation

- **Composition:** Make new objects by combining other objects

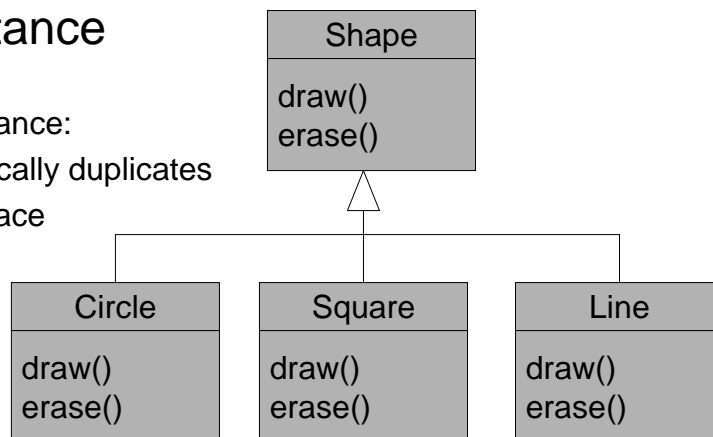


OOAD98/99-ST5-Introduction to Objects

5

Inheritance

- Inheritance:
Automatically duplicates
the interface



- Reusing interface (and implementation)
- Objects related by inheritance all have the same type (interface)

OOAD98/99-ST5-Introduction to Objects

6

Polymorphism

One piece of code ...

```
draw()  
erase()
```

Shape

... works with all this objects

Circle

Square

Line

An Amazing Trick

```
void doStuff(Shape s) {  
    s.erase();  
    ...  
    s.draw();  
}
```

```
Circle c = new Circle();  
Triangle t = new Triangle();  
Line l = new Line();
```

```
doStuff(c); // "dynamic binding"  
doStuff(t);  
doStuff(l);
```

Object-Oriented Programs

- Made up of objects sending messages to each other's interfaces

