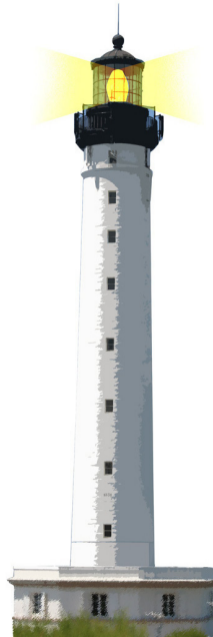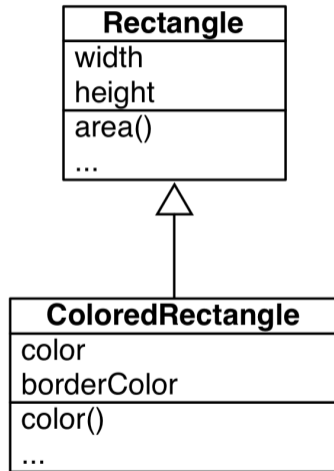# Inheritance Basics

S. Ducasse and L. Fabresse

# Goal

- What is inheritance?
- When to use it?
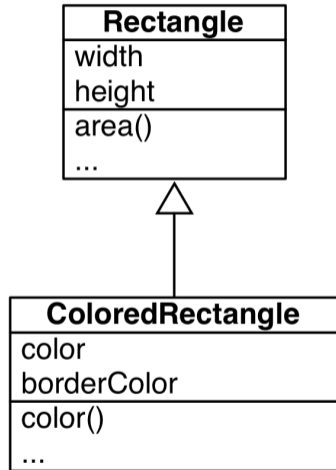- BTW, Pharo has the same inheritance than Java

# The basics

- Often we want small adaptations
- We want to extend existing behavior and state
- Solution: **class inheritance**
- a class extends the definition of its superclass

**Rectangle**
| |
|---|
| width |
| height |
| area() |
| ... |

**ColoredRectangle**
| |
|---|
| color |
| borderColor |
| color() |
| ... |

# Basic subclass behavior

a subclass

- can **add** state and behavior:
  - color, borderColor, ...
- can **use** superclass behavior and state
- can **specialize** and **redefine** superclass behavior

```
┌─────────────────────┐
│     Rectangle       │
├─────────────────────┤
│ width               │
│ height              │
├─────────────────────┤
│ area()              │
│ ...                 │
└─────────────────────┘
          △
          │
┌─────────────────────┐
│  ColoredRectangle   │
├─────────────────────┤
│ color               │
│ borderColor         │
├─────────────────────┤
│ color()             │
│ ...                 │
└─────────────────────┘
```

# Root of inheritance hierarchy

- Object is the root of most classes
  - defines the common behavior of any object
- ProtoObject (Object's superclass) has a special purpose:
  - e.g. raising as much as errors as possible
  - so that the system can catch such errors and do something with them
  - useful for building advanced techniques such as proxy objects
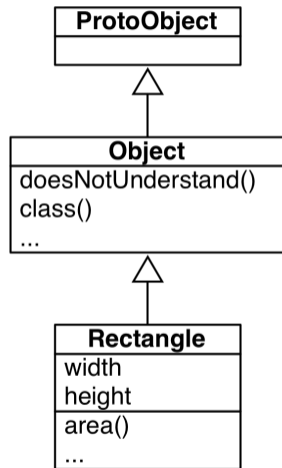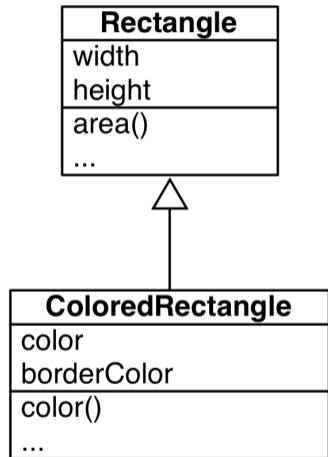
# Two aspects of inheritance

Inheritance is

- **static** for state (i.e., during class creation)
- **dynamic** for behavior (i.e., during execution)

# Inheritance of instance variables

- Happens during class definition
- Computed from
  - the class own instance variables
  - the ones of its superclasses
- ColoredRectangle has a width, height, color, and borderColor

```
┌─────────────────────┐
│     Rectangle       │
├─────────────────────┤
│ width               │
│ height              │
├─────────────────────┤
│ area()              │
│ ...                 │
└─────────────────────┘
          △
          │
┌─────────────────────┐
│  ColoredRectangle   │
├─────────────────────┤
│ color               │
│ borderColor         │
├─────────────────────┤
│ color()             │
│ ...                 │
└─────────────────────┘
```
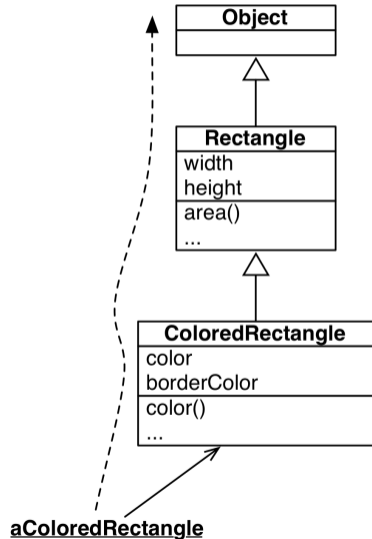
# Inheritance of behavior



- Happens at runtime
- the method is looked up
  - starting from the receiver's class
  - then going up the superclass chain
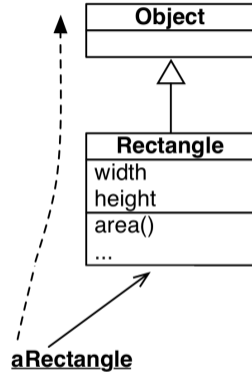
# Inheritance of behavior

- Happens at runtime
- The method is looked up
  - starting from the receiver's class
  - then going to the superclass

# What You Should Know

- Inheritance allows a class to refine state and behavior
- A class has 1 and only 1 superclass
- A class eventually inherits from Object
- Inheritance of state is static
- Inheritance of behavior is dynamic

A course by

S. Ducasse, L. Fabresse, G. Polito, and Pablo Tesone