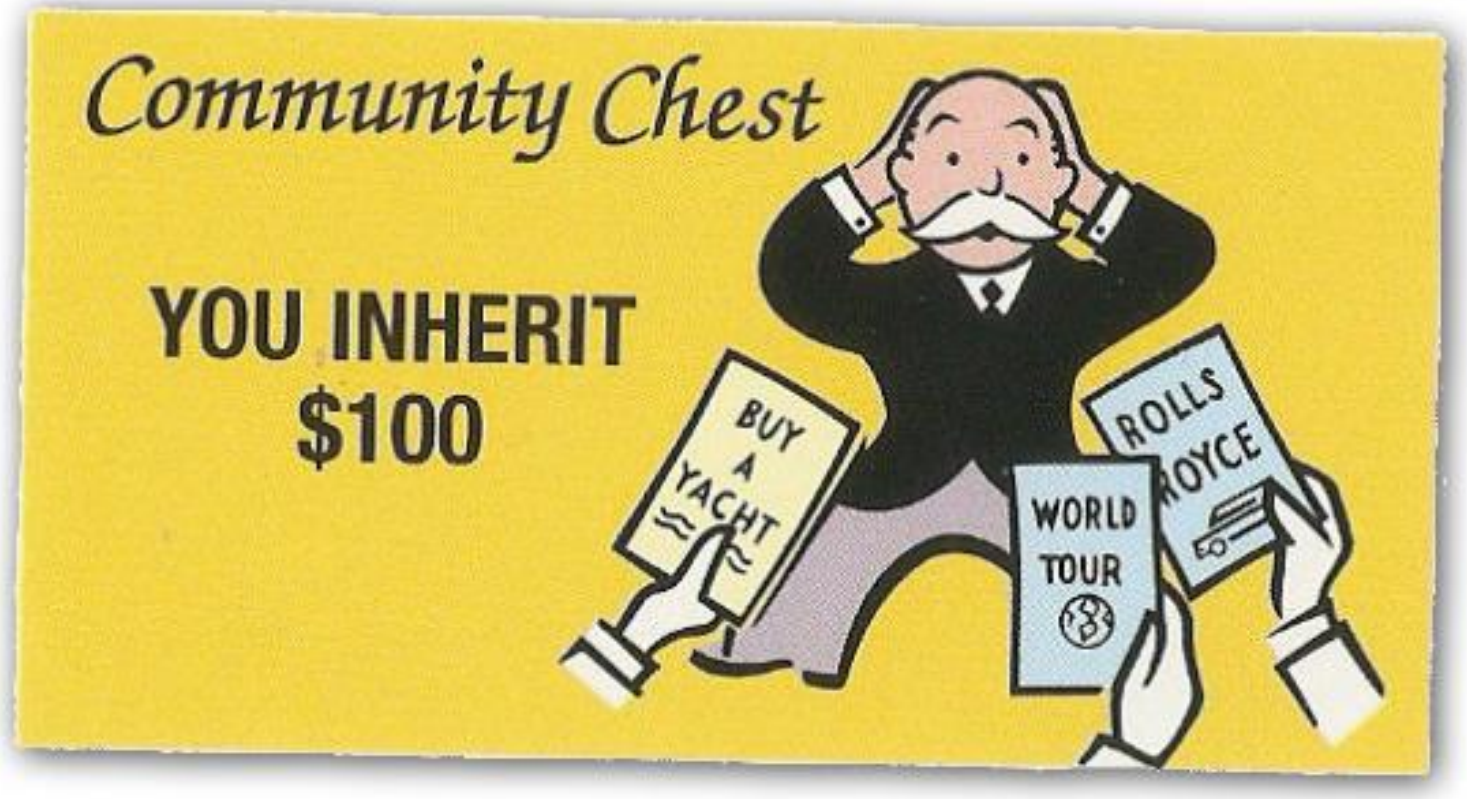
Inheritance

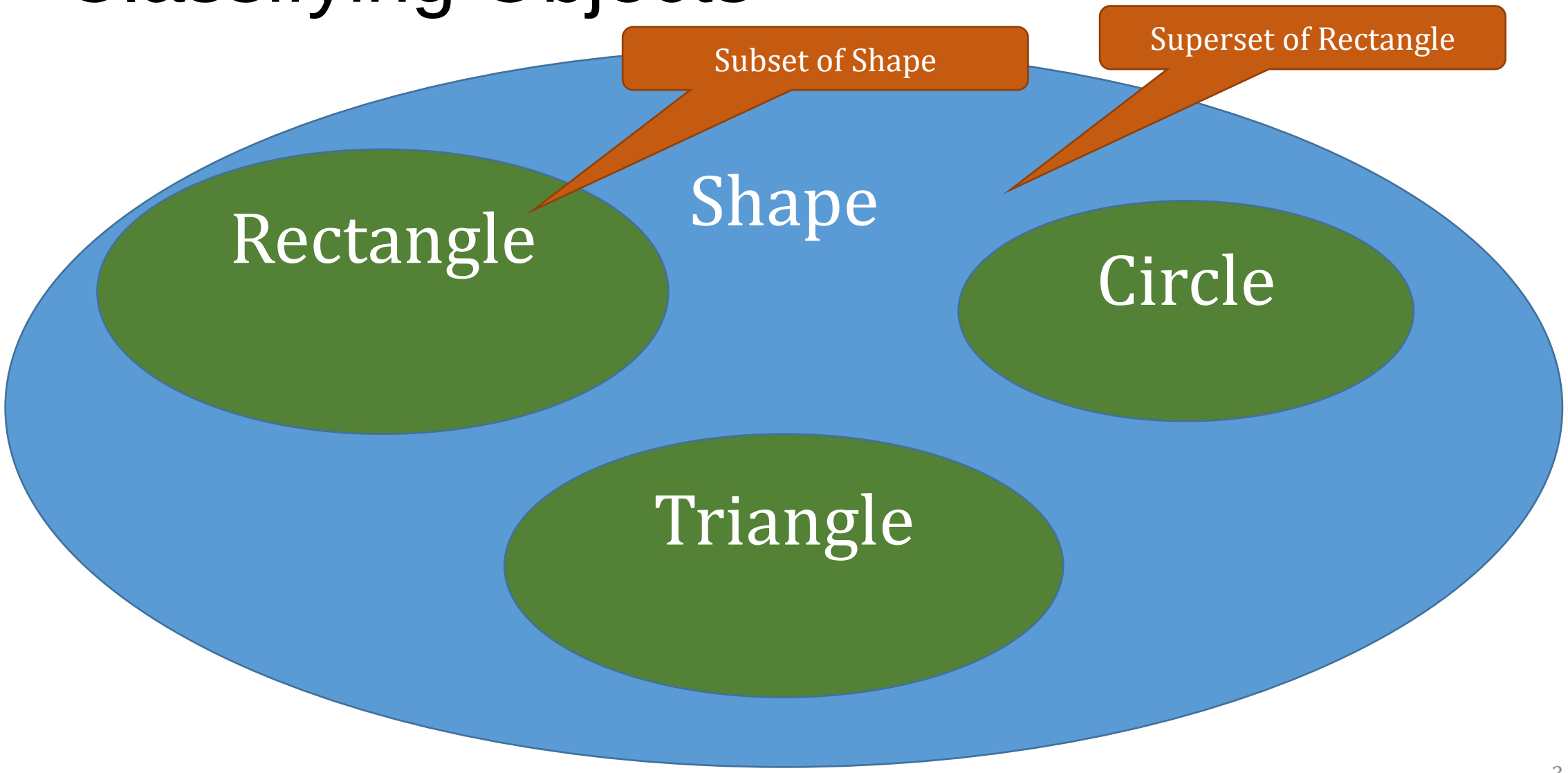
Chapter 9



Problem: Shapes

- Keep track of Rectangles, Circles, and right Triangles in a cartesian coordinate system
- Need a Point (x,y), a min point, a max point, a perimeter, and an area for each kind of shape
- Each shape should be movable

Classifying Objects



Why Inherit?

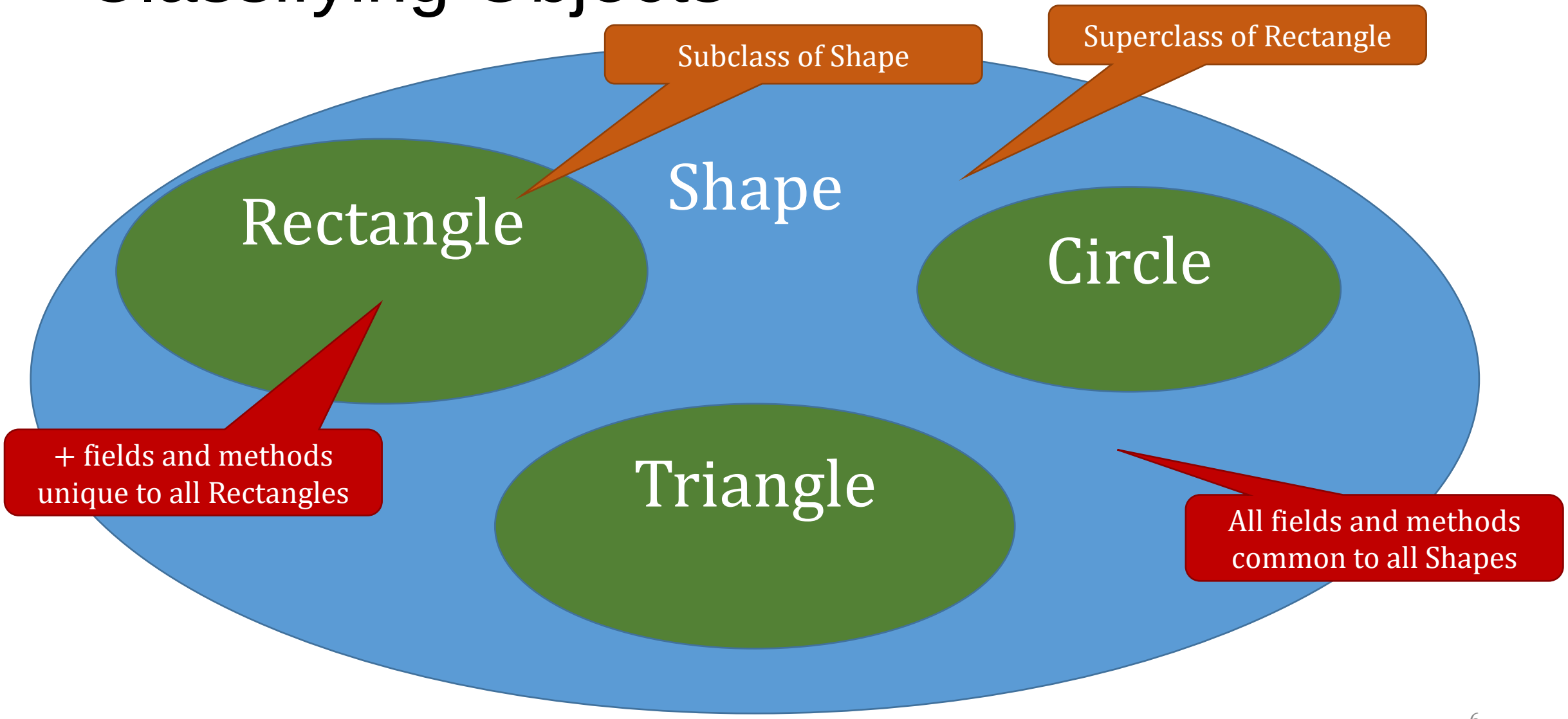
- There are some things that are true for all shapes
  - They all have a lower left point
  - They all move by modifying the lower left point
  - They all have the same minimum point
- It would be nice to be able to deal with all the common stuff once
  - I shouldn't have to duplicate a move method or ll field for rectangles, circles, triangles, …
- But there is some stuff which is pretty specific
  - Only circles have a radius

The inheritance concept

- Define all the stuff that pertains to all shapes in a Shape class
  - including fields and methods

- Then define a "sub" class, like Rectangle that "inherits" the Shape class
  - That means that Rectangle automatically contains all the fields and methods defined in the Shape class
  - That means a Rectangle can do anything a Shape can do
  - But we can add Rectangle specific fields and methods like "width" or "perimeter"

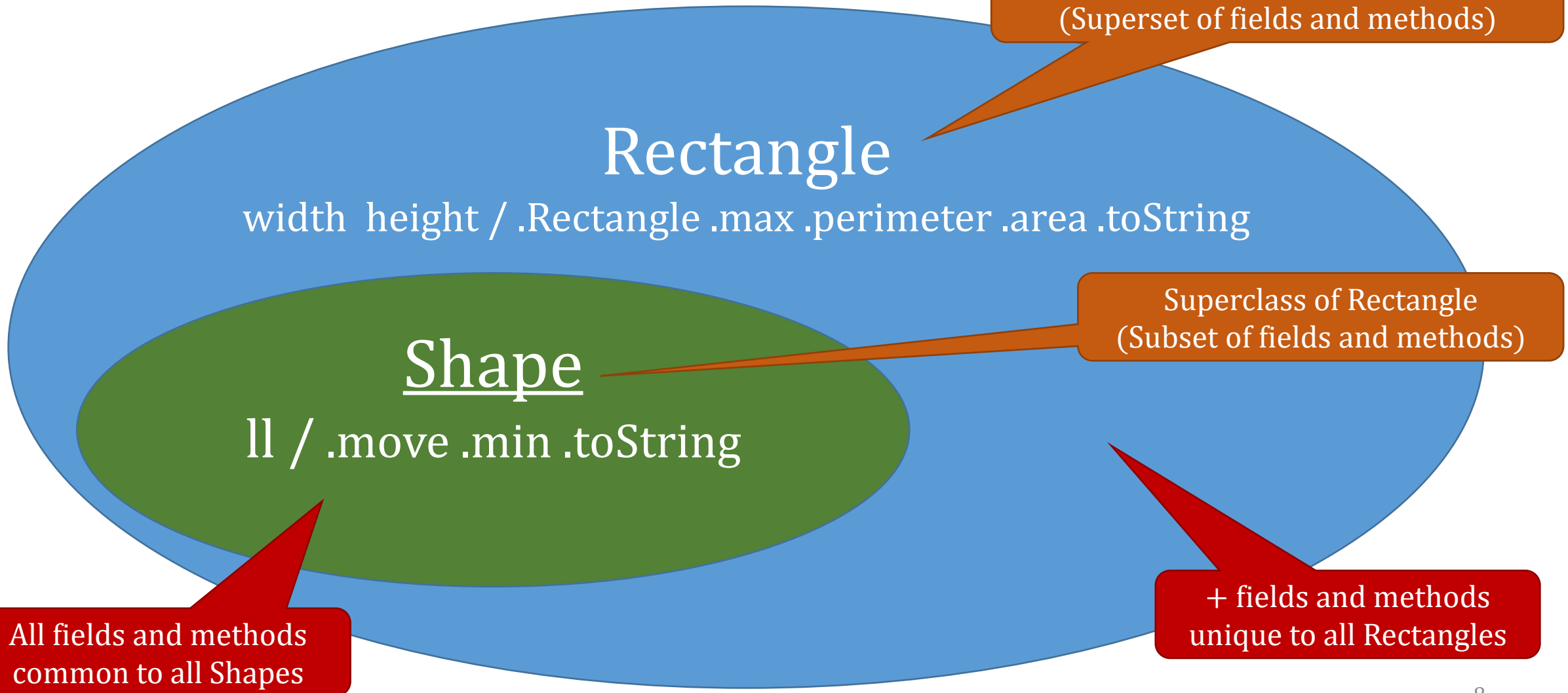
Classifying Objects



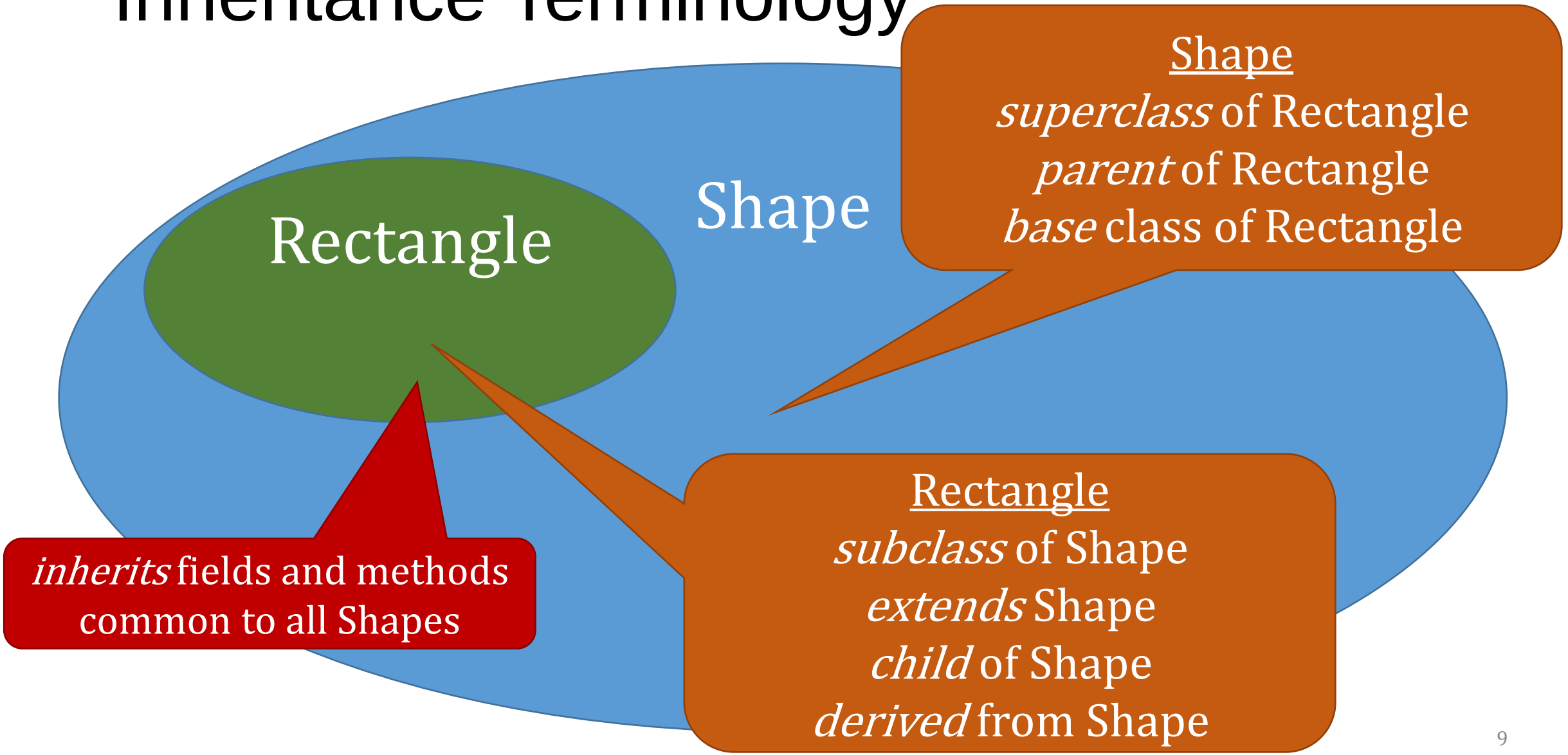
Extending the concept of “Shape”

- Classes: Rectangle, Circle, and Triangle “extend” a Shape
- “Extend” means do everything a Shape does **and** do more
  - All “inherit” the fields defined by Shape (ll)
  - All “inherit” the methods defined by Shape (min,move,toString)
  - May have their own fields (extend with: width,height,radius,…)
  - May have their own methods (extend with: max, perimeter, area, …)
  - May re-define (override) Shape methods (toString)

Sub-class Naming Paradox



Inheritance Terminology



Class Inheritance Syntax

```
class classname extends basename {
        …
}
```

classname : Child class name

basename : Parent class name

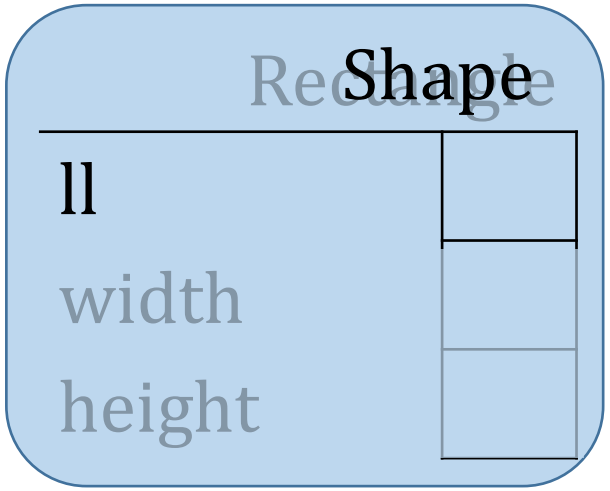
```
public class Rectangle extends Shape {
  double base;
  double height;
  public Rectangle(Point ll,double base,double height) {
    super(ll); // Contains this.ll=ll
    this.base = base;
    this.height = height;
  }
  …
```

Inheritance Access

- The subclass "inherits" all the fields from its parent
  - If a field is private in the parent, the child cannot access it
- The subclass "inherits" all the methods from its parent class
  - If a method is private in the parent, the child cannot use it

Duality of Parent/Child Objects

- Can be thought of as a Shape object
  - With field “ll”
  - With methods min, move, and toString



- Can be though of as a Rectangle object
  - With fields “ll”, “width” and “height”
  - With methods min, move, max, perimeter, area, and toString

- WARNING: Don’t make a child class field with the same name as a parent class field!

The “super” keyword

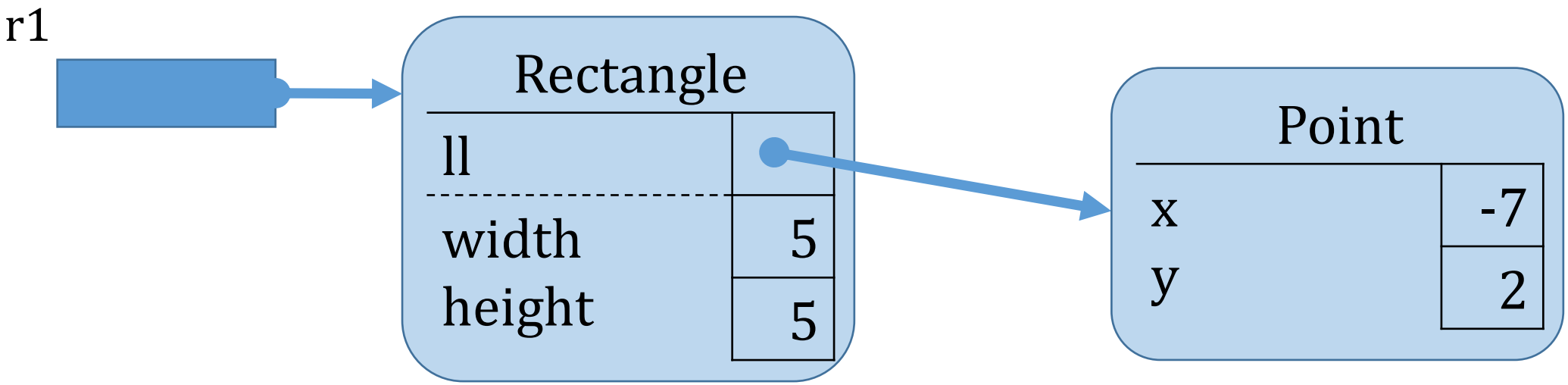
- In a child class, the keyword “super” refers to the parent class

- For instance, if Rectangle is a subclass of Shape, in Rectangle:
  - “super()” refers to the Shape constructor,
  - “toString()” refers to the Rectangle toString method
  - “super.toString()” refers to the Shape toString method

Parent/Child Object Construction

1. Space is allocated for the child object
2. All fields (child and parent) are initialized with 0 or null
3. The child constructor is invoked
   - The child constructor may explicitly invoke the parent constructor
     - with or without arguments!
     - If so, super(…) must come first!
   - If the child constructor does ***not*** invoke a parent constructor, Java invokes the parent no-argument constructor
   - The rest of the child constructor is executed

Memory Image of a Rectangle



Static Methods and Inheritance

- Static (class) methods are NOT inherited/overridden
- You can access a static method by specifying
  ClassName.*methodName*(…)
- No need for inheritance… you can get at class methods without inheritance

Example of using Inheritance

- After Steve Jobs left Apple (for a while), he created the "NeXT" computer
- The "NeXT" operating system and developer libraries use a lot of Objective C (this code was brought back to the Apple Macs and the iPhone)
- The developer toolkits include a full scale text editor.
- Tim Berners-Lee was able to use the editor framework, combined with a few of his own subclasses, to create the first web browser

Quote: https://www.w3.org/People/Berners-Lee/WorldWideWeb.html

… I could do in a couple of months what would take more like a year on other platforms, because on the NeXT, a lot of it was done for me already. … I just had to add hypertext, *(by subclassing the Text object)*

Tim Berners-Lee