Inheritance and Composition

John Sonmez
http://simpleprogrammer.com
John.Sonmez@gmail.com

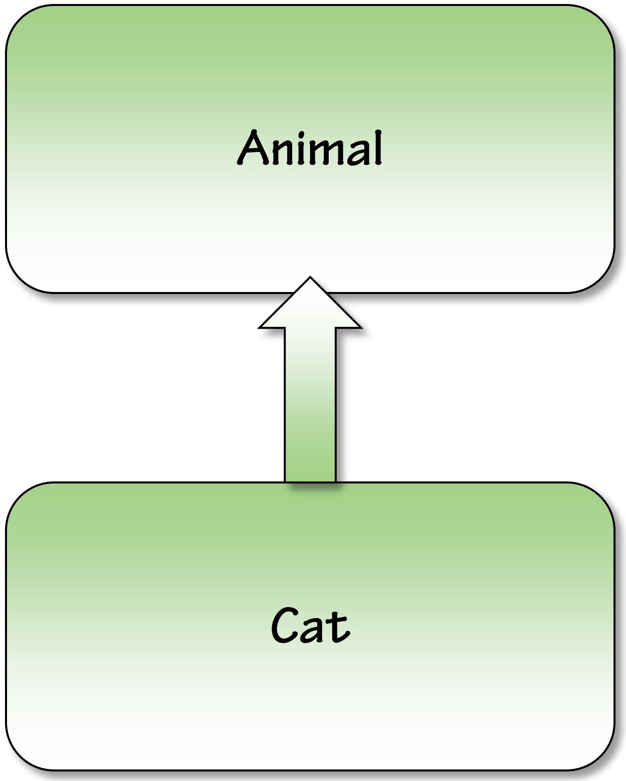
Outline

- Is-A and Has-A
- Basic Inheritance
- Basic Composition
- Poly-What?
- Favor Composition
- Interfaces

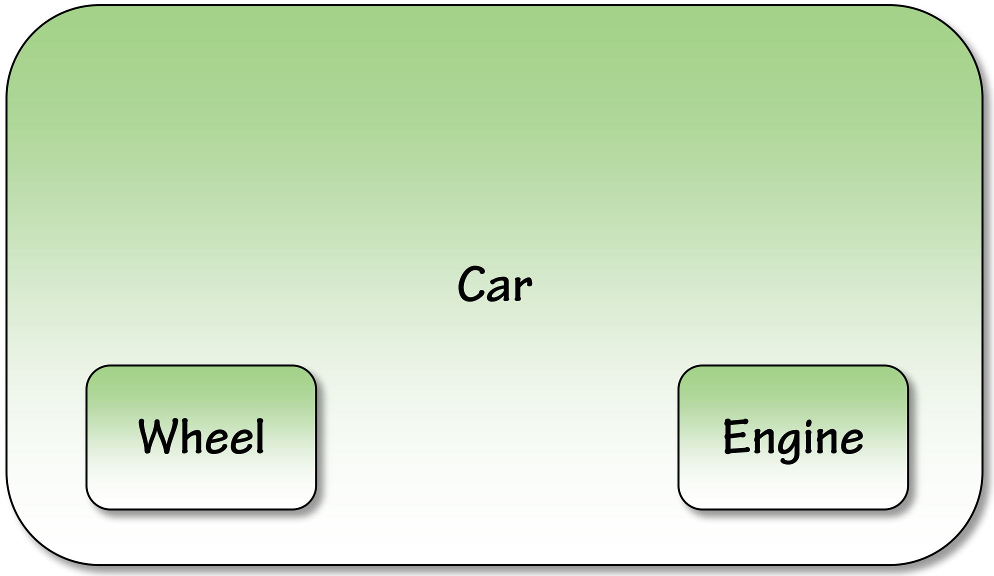
Is-A and Has-A

- **Classes do not stand alone**
- **Classes are related**
- **We need ways to model real relationships between things**
  - Cat -> Animal
  - Car -> Wheels, Engine
  - Train -> Wheels, Engine
  - Car, Truck -> Vehicle
- **Need to be able to reuse code**

Inheritance (Is-A)



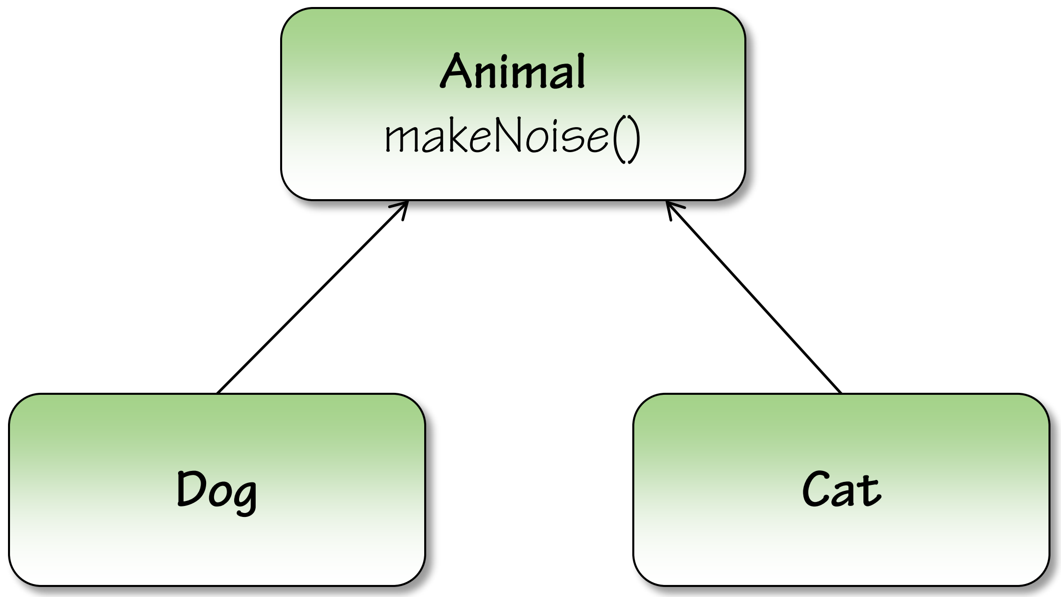
Composition (Has-A)



Poly-what?

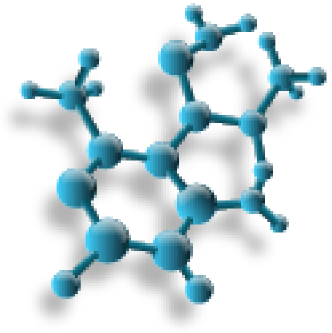
- **Polymorphism**
  - Many
  - Form
- **2 Components**
  - Code is dependent on an interface
  - The behavior is determined by the actual class implementing that interface

Example



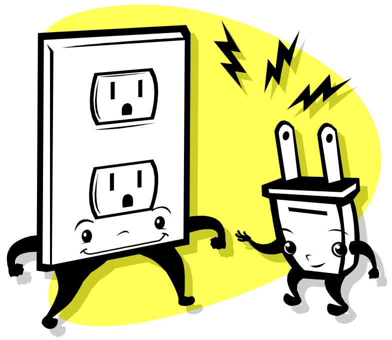
Favor Composition

- **Almost all inheritance can be rewritten**
- **Object oriented programming changes**
  - Less about modeling the real world
  - More about modeling
    - Interactions in a system
    - Roles and responsibilities
- **Inheritance is difficult to maintain**

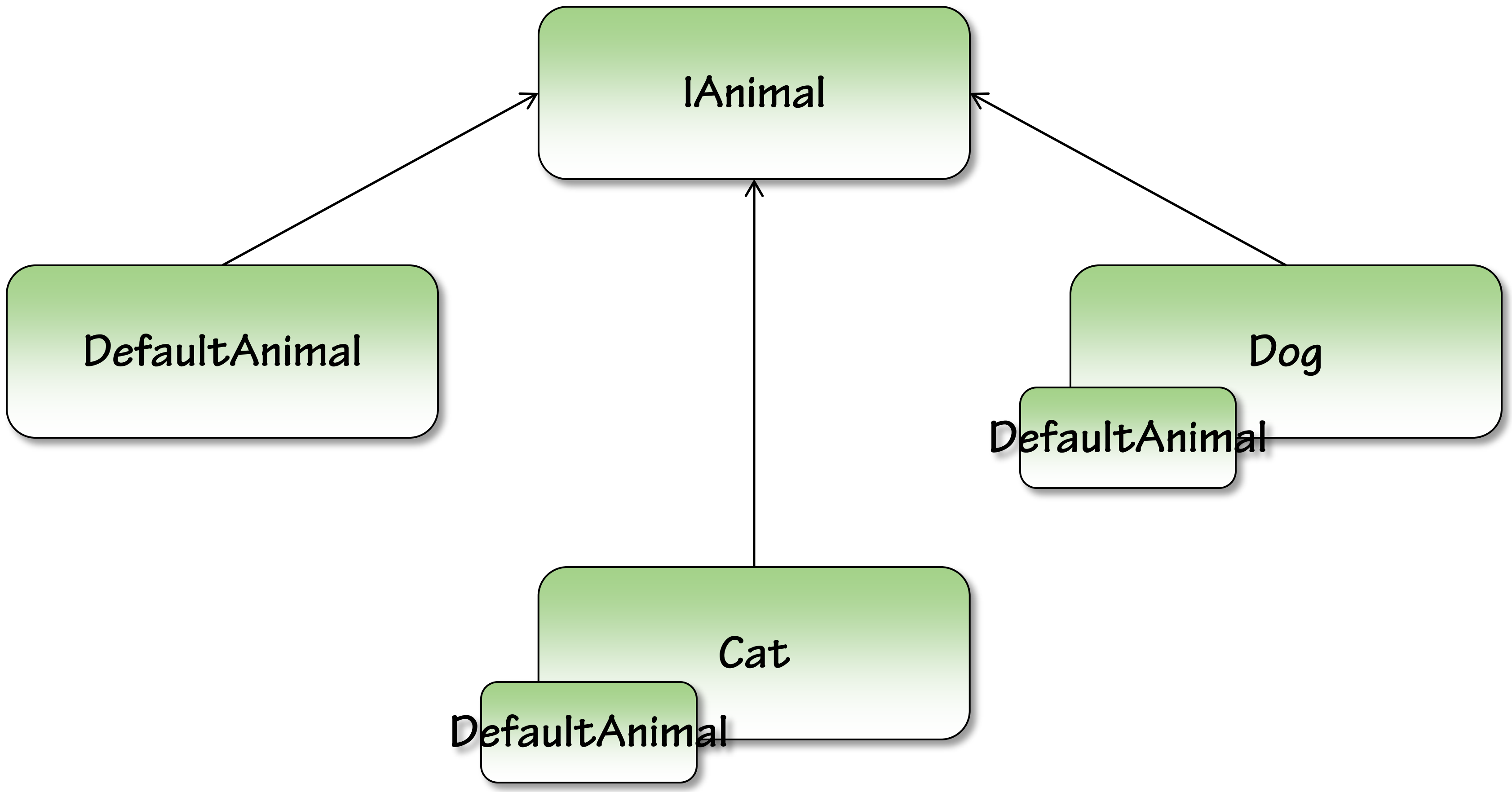


Interfaces

- Inheritance without the baggage!
- No implementation details
- Multiple implementations
- Works with polymorphism



Example


IAnimal
DefaultAnimal
Dog
DefaultAnimal
Cat
DefaultAnimal

Summary

- Is-A and Has-A
- Basic Inheritance
- Basic Composition
- Poly-What?
- Favor Composition
- Interfaces