



Method Overriding

Method Overriding

- Giving different implementations to the method
- One method having **multiple** different implementations
- Overriding a method **must** take place in subclass
- Less memory usage and Improves the reusability of our code

Method Overriding

DOG

eat()

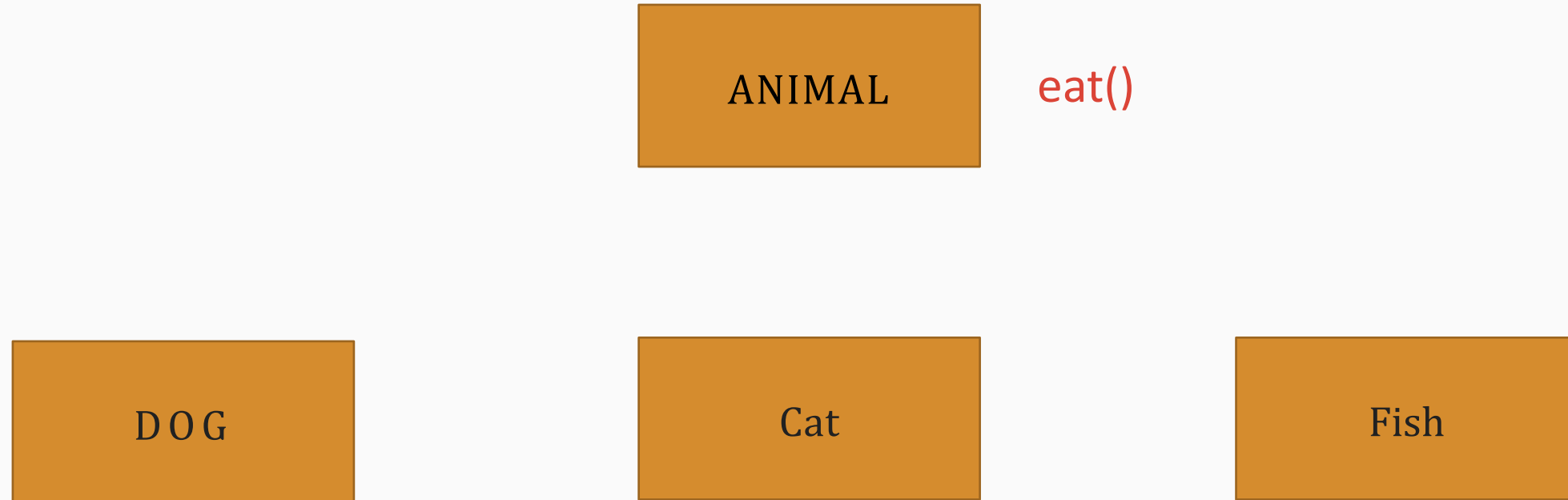
Cat

eat()

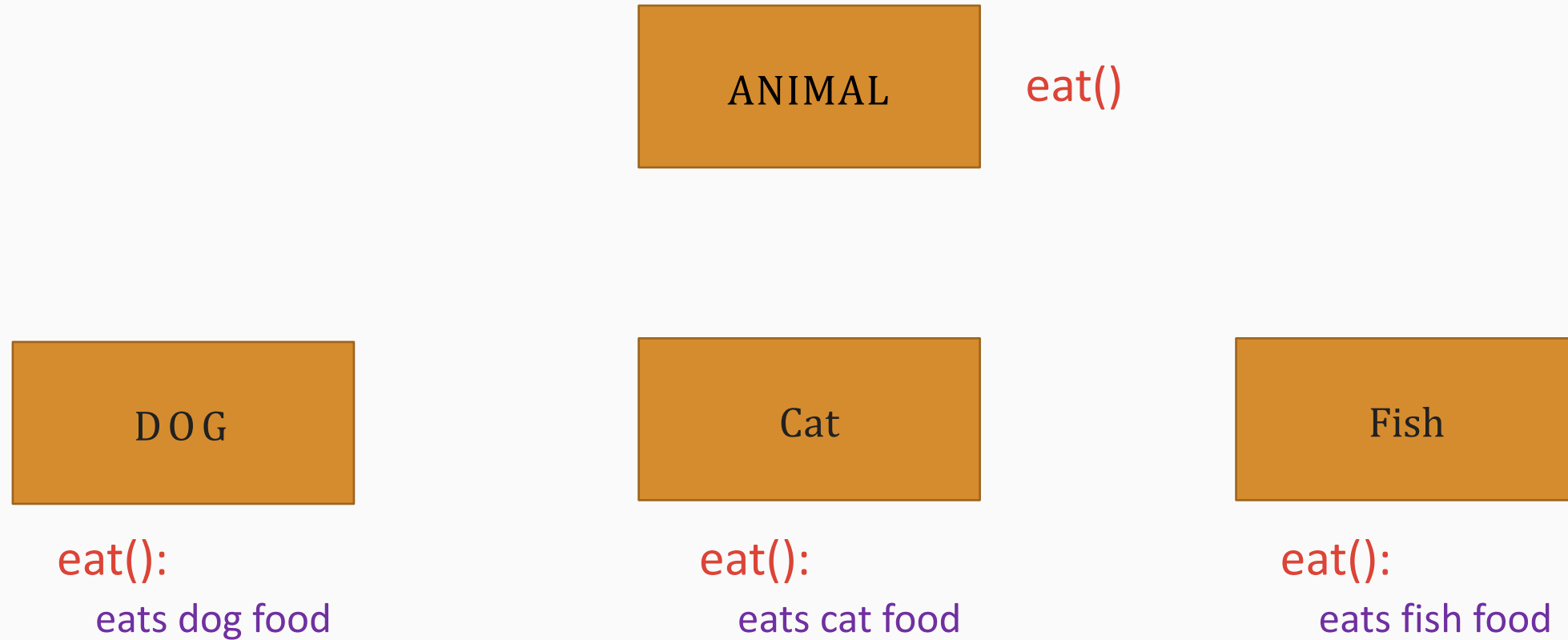
Fish

eat()

Method Overriding



Method Overriding



Method Overriding

Circle

area()
perimeter()

Square

area()
perimeter()

Rectangle

area()
perimeter()

Method Overriding

Shape

area()
perimeter()

Circle

Square

Rectangle

Method Overriding

Shape

area()
perimeter()

Circle

area():
 $\text{radius} * \text{radius} * \pi$

perimeter():
 $2 * \text{radius} * \pi$

Square

area():
 $\text{side} * \text{side}$

perimeter():
 $\text{side} * 4$

Rectangle

area():
 $\text{width} * \text{length}$

perimeter():
 $2 * (\text{width} + \text{length})$

Method Overriding Rules

- Must happen in the sub class
- Return type, method name and parameters of the overridden method **must** be same
- Access modifier of the overridden method needs to be **same** or **more visible**
- We can not override methods with private access modifier, or with static & final specifiers
- Only the instance methods (not private & not final) can be overridden

Method Overriding Example

```
public class Shape{  
    public double area(){  
        return 0;  
    }  
  
    public double perimeter(){  
        return 0;  
    }  
}
```

```
public class Circle extends Shape{  
    public double radius;  
    public static double pi = 3.14;  
  
    @Override  
    public double area(){  
        return radius * radius * pi;  
    }  
  
    @Override  
    public double perimeter(){  
        return 2 * radius * pi;  
    }  
}
```

Overriding vs Overloading

Method Overloading	Method Overriding
Method overloading is performed within class	Method overriding occurs in two classes that have IS-A relationship
parameter must be different	parameter must be same
Access specifier can be changed	Access specifier must not be more restrictive than original method
private and final methods can be overloaded	private and final methods can not be overridden
Return type of method does not matter, it can be same or different	Return type must be same in method overriding