

POLYMORPHISM

# TODAY'S OBJECTIVES

- Polymorphism
  - Using inheritance
  - Using interfaces
- Why use interfaces?

# POLYMORPHISM IN JAVA

In object-oriented programming, **polymorphism** is the idea that something can be assigned a different meaning or usage based on the context it is referred to as. Put another way, different objects can be treated as the same type of thing within a program.

- Polymorphism using inheritance is the concept that any object which is a subclass can be treated as the superclass type.
  - `Auction buyOutAuction = new BuyOutAuction("Super cool thing", 150);`
- Polymorphism can also be implemented using **interfaces**.

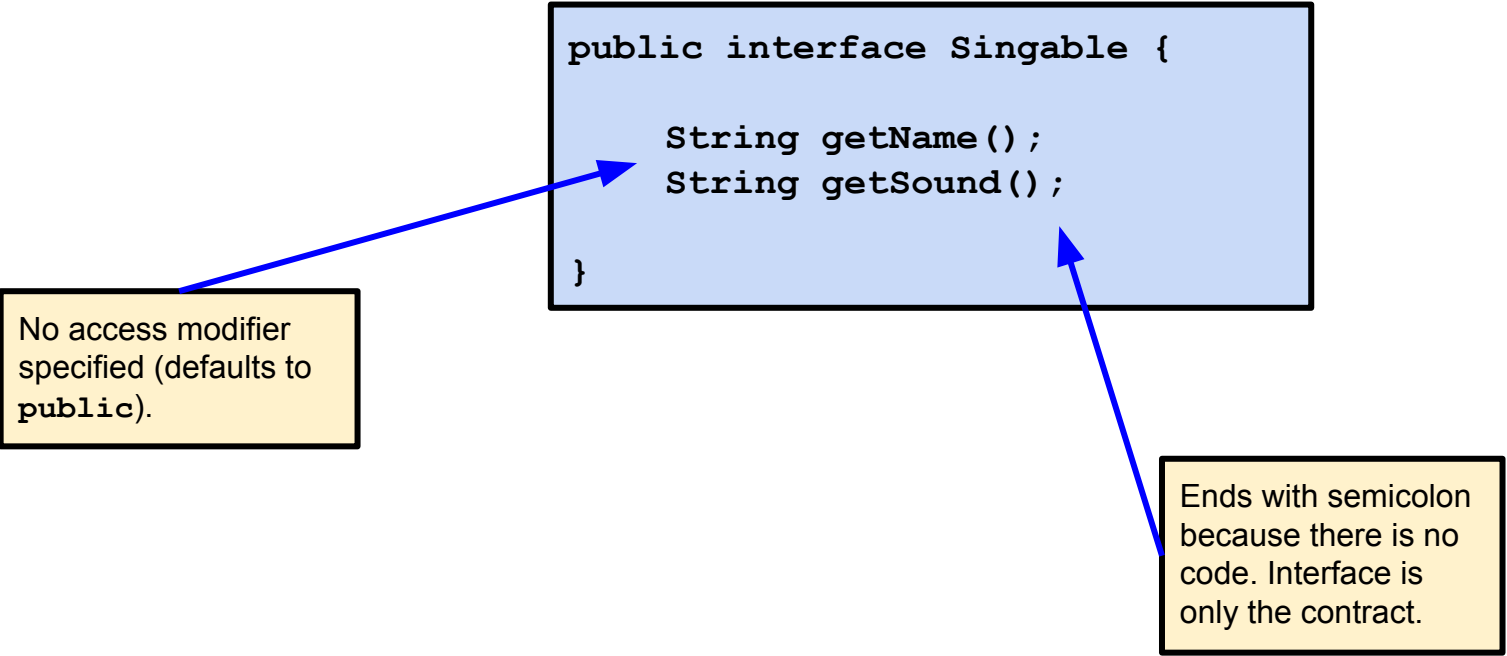
OLD MACDONALD...

# INTERFACES

- Interfaces define behaviors that objects must implement.
  - You can think of them as contracts that must be followed.
- Keyword: `implement`
- Defines what (the method signature) but not how (the actual method body).
- An interface is a contract that defines which methods a user of the interface can expect.
- Cannot be instantiated.
- Objects may implement more than one interface.
- If class A implements interface B, the A "is-a" B, and so are its subclasses.

# INTERFACES

```
public interface Singable {  
  
    String getName();  
    String getSound();  
  
}
```



No access modifier specified (defaults to **public**).

Ends with semicolon because there is no code. Interface is only the contract.