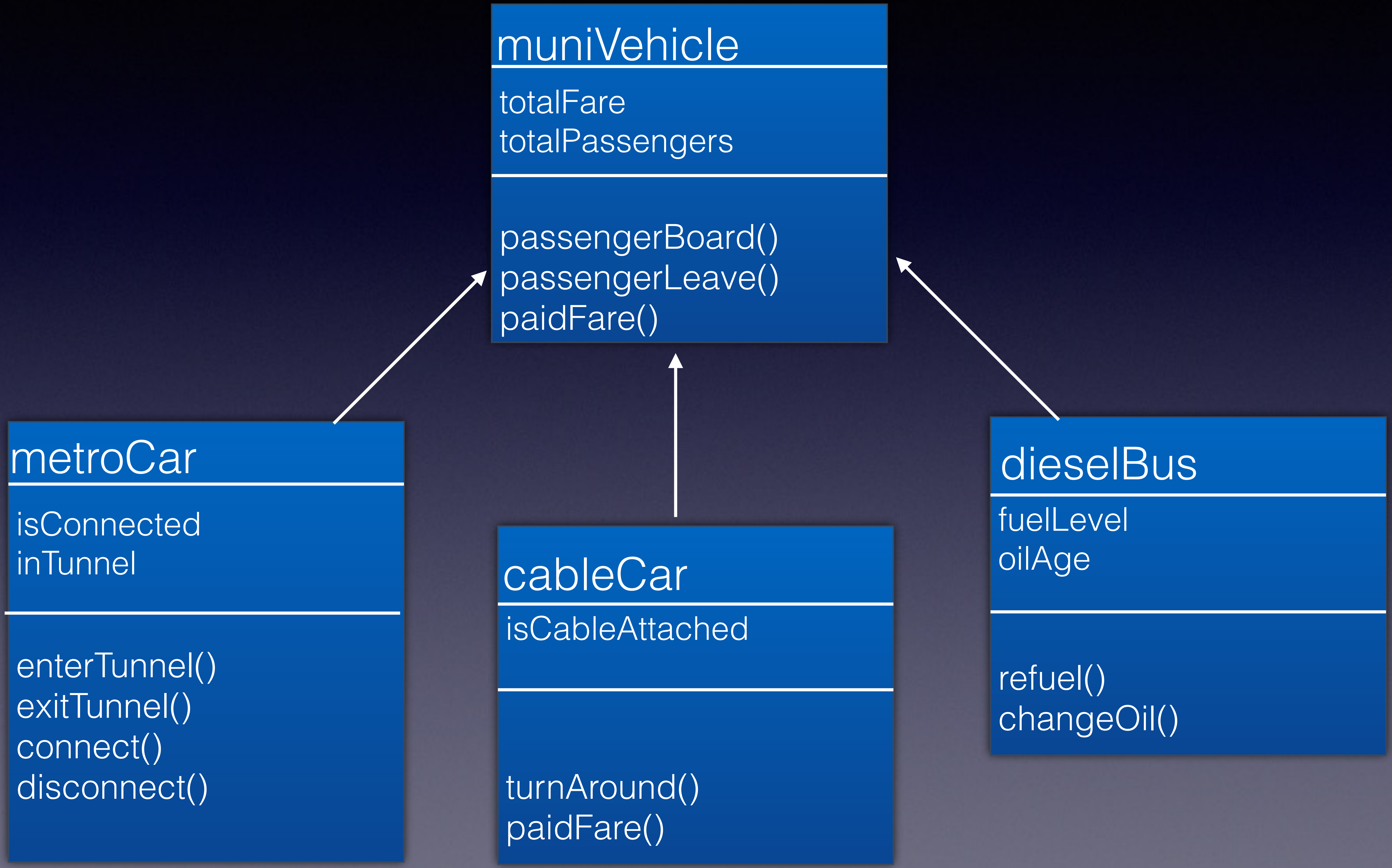
Swift Fundamentals V
Inheritance

CS112 Unit 6
Max Luttrell, Fall 2016

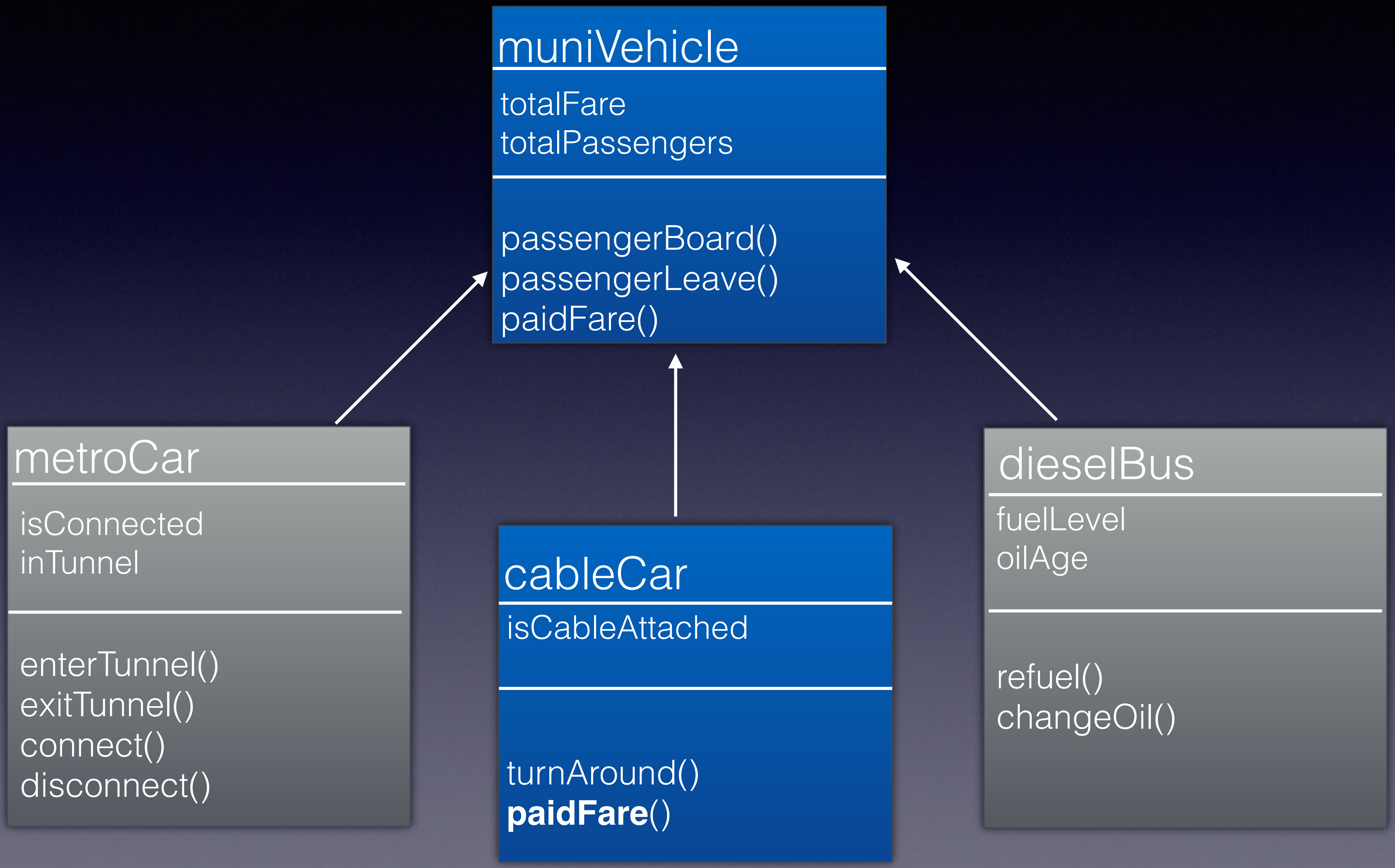
inheritance

- Inheritance allows a new class to be based on an existing class. The new class inherits the properties and methods of the class it is based on.

inheritance



inheritance



Player class

```
class Player {
    var name = ""
    var weight = 0.0
    var height = 0.0
    var age = 0
    func printInfo() {
        print("\(name)")
        print("\(weight) kg, \(height) m, \(age) yrs")
    }
    func incrementAge() {
        age += 1
    }
}
```

```
var qb = Player()
qb.name = "Joe Montana"
qb.weight = 93
qb.height = 1.88
qb.age = 60
qb.printInfo()
```

sample debug output

```
Joe Montana
93.0 kg, 1.88 m, 60 yrs
```

subclass

- we will make the BasketballPlayer class a **subclass** of class Player, and add some basketball-specific properties:

```
class Player {
    var name = ""
    var weight = 0.0
    var height = 0.0
    var age = 0
    func printInfo() {…}
    func incrementAge() {…}
}
```

```
class BasketballPlayer : Player {
    var fieldgoals = 0
    var attempts = 0
}
```

- because BasketballPlayer is a subclass of Player, it **inherits** all of Player’s properties and methods

inheritance - example

```
class Player {
    var name = ""
    var weight = 0.0
    var height = 0.0
    var age = 0
    func printInfo() {…}
    func incrementAge() {…}
}
```

```
class BasketballPlayer : Player {
    var fieldgoals = 0
    var attempts = 0
}
```

```
var pointguard = BasketballPlayer()
pointguard.name = "Stephen Curry"
pointguard.weight = 86.2
pointguard.height = 1.91
pointguard.age = 28
pointguard.fieldgoals = 402
pointguard.attempts = 886

pointguard.printInfo()
```

sample debug output

```
Stephen Curry
86.2 kg, 1.91 m, 28 yrs
```

overriding a function

```
class Player {
    var name = ""
    var weight = 0.0
    var height = 0.0
    var age = 0
    func printInfo() {…}
    func incrementAge() {…}
}
```

```
class BasketballPlayer : Player {
    var fieldgoals = 0
    var attempts = 0
    override func printInfo() {
        let percentage = Double(fieldgoals) / Double(attempts)
        print("Fieldgoal percentage: \(percentage)")
    }
}
```

```
pointguard.printInfo()
```

sample debug output

```
Fieldgoal percentage: 0.45372460496614
```

calling super class' function

```
class Player {
    var name = ""
    var weight = 0.0
    var height = 0.0
    var age = 0
    func printInfo() {…}
    func incrementAge() {…}
}
```

```
class BasketballPlayer : Player {
    var fieldgoals = 0
    var attempts = 0
    override func printInfo() {
        let percentage = Double(fieldgoals) / Double(attempts)
        super.printInfo()
        print("Fieldgoal percentage: \(percentage)")
    }
}
```

```
pointguard.printInfo()
```

sample debug output

```
Stephen Curry
86.2 kg, 1.91 m, 28 yrs
Fieldgoal percentage: 0.45372460496614
```

Exercise 6A

- In a playground, define the Player and BasketballPlayer classes as discussed today (you can just copy from the slide)
- Create a new class BaseballPlayer which is derived from Player. It should have two properties: atBats, and hits, initialized to zero
- Override the printInfo() function to display the batting average, for which the formula is: hits / atBats. Note: it should also display name, weight, height, age.
- Add some Swift code which creates two objects: one BaseballPlayer and one BasketballPlayer. Initialize the objects with some values of your choosing, and call printInfo() on both.
- Call the incrementAge() function on your BaseballPlayer object, and call printInfo() one more time. Does it work? Why?