Overriding Review, APIs, Access Control, Enums

by Ash Dreyer & Donald Pinckney

Table of Contents

- Review of Overriding Methods
- APIs: Examples, Encapsulation
- Access Modifiers
- Enumerations

Review of Overriding Methods

```
class A {
   func f() {
      print(“A”)
   }
}
class B: A {
   override func f() {
      super.f()
      print(“B”)
   }
}
class C: B {
   override func f() {
      super.f()
      print(“C”)
   }
}
let cat = A()
let dog = B()
let mouse = C()
let rat: B = C()
let lizard: A = C()
let snake: A = B()
```

```
cat.f()
dog.f()
mouse.f()
rat.f()
lizard.f()
snake.f()
```

APIs - Kinda the Point of Classes

Apple iOS Exercise

- Pretend you are working at Apple, working on first iPhone
- 3rd Party Developers need to make apps
- Creating user interface should be:
  - Easy
  - Consistent
- Design a system of classes / subclasses for this

Access Control

Access Control

```
class UILabel {

    var text: String = “”
    var isSelected: Bool = false

    …

}
```

Access Control

```
class UILabel {

    var text: String = “”
    private var isSelected: Bool = false

    …

}
let label = UILabel()
print(label.isSelected) // Compiler Error!
```

Access Control

```
class UILabel {

   private var text: String = “”
   private var isSelected: Bool = false

   func setText(_ text: String) {
      self.text = text
   }
   func getText() -> String {
      return text
   }

   …

}
```

Access Control

- Make as many things private as possible!
- Only make thing un-private as you need to

Enumerations

Enumerations (`enum`)

- If something can be in 2 states, then you can use a Bool
  - Example:
  - isBlackTurn: Bool
- Sometimes we want something in more than 2 states:
  - Cardinal Direction (East, South, West, North)
  - We **could** use an Int (0, 1, 2, 3)
  - But this becomes hard to keep track of, easy to mess up
  - We want to **label** these integers with better names

Enumerations

```
enum Direction {
    case east, south, west, north
}

var dir = Direction.south
if dir == Direction.south {
    dir = Direction.west
}
print(dir)
```

Enumerations

```
enum CheckersBoardColor {
    case red, black, empty
}
```