

Intermediate Swift

Part 5: Closures

Closures

- ⚙️ Closures captures any variables referenced in the code.

```
func incrementer() -> () -> Int {  
    var value: Int = 0  
    return { () -> Int in  
        value += 1  
        return value  
    }  
}  
  
let increment = incrementer()  
increment() // 1  
increment() // 2
```

- ⚙️ Swift determines when a value is referenced or copied.

Closure Syntax

{ (**param 1 name** **:** **param 1 type** **,** **...** **) ->** **retval type** **in**
statements **}**

```
publicDatabase.performQuery(query, inZoneWithID: nil, completionHandler:
{ (CKRecord record, NSError error)
    in
    if error != nil {
        println(error.localizedDescription)
    }
})
```

Closures (con't)

⚙ Inferred syntax

```
publicDatabase.performQuery(query, inZoneWithID: nil, completionHandler:
{ record, error in
    if error != nil {
        println(error.localizedDescription)
    }
})
```

⚙ Trailing closures

```
publicDatabase.performQuery(query, inZoneWithID: nil) { record, error in
    if error != nil {
        println(error.localizedDescription)
    }
}
```


Closures (cont'd)

```
var numbers = [100, 20, 12, 34, 1]
```

⚙ Implicit returns

```
names.sort() { number1, number2 in (number1 > number2) }
```

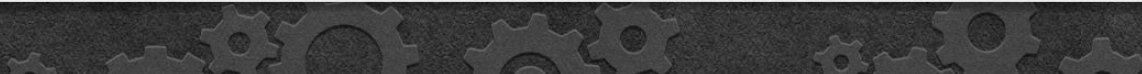
⚙ Shorthand argument names

```
names.sort { return $0 > $1 }
```

⚙ Operator functions

```
names.sort(>)
```

Demo



Challenge Time!

```
var calc = InterestCalculator()  
calc.principal = 56504  
calc.rate = 7.5  
calc.time = 10  
  
calc.simpleInterest // 98,882  
calc.compoundInterest // 116,456.58
```

