TEB

iOS Programlama Eğitimi
4. Gün

Initialization

- Bir class, struct yada enum nesnesi hazırlama işlemi
- Objective-C’de init metodları değeri return eder
- Swift’te return yok

Initializers

```
init() {
    // perform some initialization here
}
```

Initializers

```
struct Celsius {
    var temperature: Double
    init() {
        temperature = 20.0
    }
}
var f = Celsius()
println("Sıcaklık \(f.temperature)° ")
// "Sıcaklık 20.0° "
```

Initializers

```
struct Celsius {
    var temperature: Double
    init() {
        temperature = 20.0
    }
}
var f = Celsius()
println("Sıcaklık \(f.temperature)° ")
// "Sıcaklık 20.0° "
```

Initialization Parameters

```swift
struct Celsius {
    var temperatureInCelsius: Double
    init(fromFahrenheit fahrenheit: Double) {
        temperatureInCelsius = (fahrenheit - 32.0) / 1.8
    }
    init(fromKelvin kelvin: Double) {
        temperatureInCelsius = kelvin - 273.15
    }
}
let boilingPointOfWater = Celsius(fromFahrenheit: 212.0)
// boilingPointOfWater.temperatureInCelsius is 100.0

let freezingPointOfWater = Celsius(fromKelvin: 273.15)
// freezingPointOfWater.temperatureInCelsius is 0.0
```

Local and External Parameter Names

```
struct Color {
    let red, green, blue: Double
    init(red: Double, green: Double, blue: Double) {
        self.red   = red
        self.green = green
        self.blue  = blue
    }
    init(white: Double) {
        red   = white
        green = white
        blue  = white
    }
}
```

Local and External Parameter Names

```
let magenta = Color(red: 1.0, green: 0.0, blue: 1.0)
let halfGray = Color(white: 0.5)
```

Initializer Parameters Without External Names

```
struct Celsius {
    var temperatureInCelsius: Double
    init(fromFahrenheit fahrenheit: Double) {
        temperatureInCelsius = (fahrenheit - 32.0) / 1.8
    }
    init(fromKelvin kelvin: Double) {
        temperatureInCelsius = kelvin - 273.15
    }
    init(_ celsius: Double) {
        temperatureInCelsius = celsius
    }
}
let bodyTemperature = Celsius(37.0)
// bodyTemperature.temperatureInCelsius is 37.0
```

Memberwise Initializers

```
struct Size {
    var width = 0.0, height = 0.0
}
let twoByTwo = Size(width: 2.0, height: 2.0)
```

Initializer Delegation

```
struct Size {
    var width = 0.0, height = 0.0
}

struct Point {
    var x = 0.0, y = 0.0
}
```

Initializer Delegation

```swift
struct Rect {
    var origin = Point()
    var size = Size()
    init() {}
    init(origin: Point, size: Size) {
        self.origin = origin
        self.size = size
    }
    init(center: Point, size: Size) {
        let originX = center.x - (size.width / 2)
        let originY = center.y - (size.height / 2)
        self.init(origin: Point(x: originX, y: originY), size: size)
    }
}
```

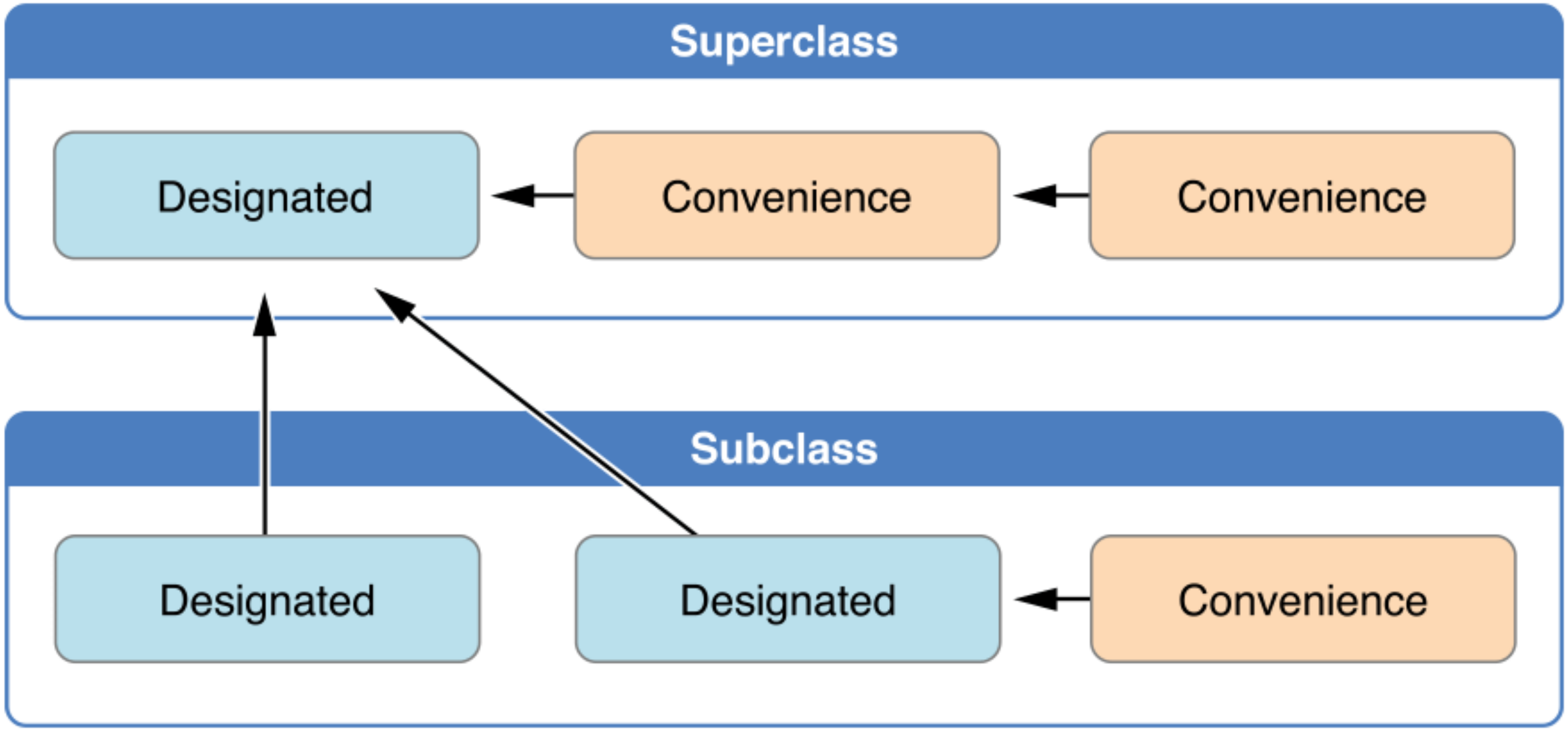
Designated and Convenience Initializers

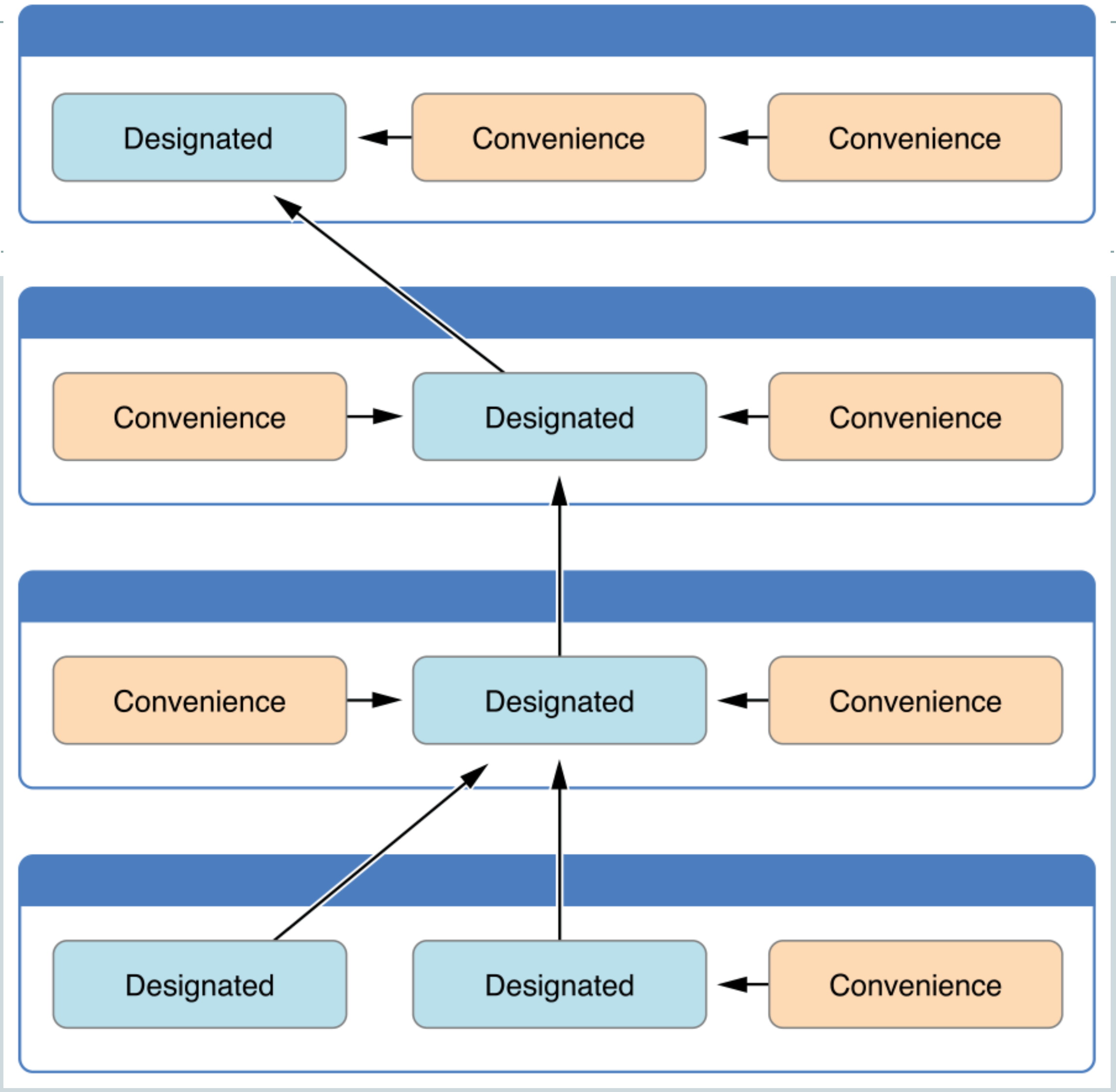
```
init(parameters) {
    statements
}

convenience init(parameters) {
    statements
}
```

Initializer Chaining

- Kural 1
  - ‘Designated initializer’ , super class’ının ‘designated initializer’ methodunu çağırır
- Kural 2
  - ‘Convenience initializer’ aynı class’taki farklı bir initializer’ı çağırır
- Kural 3
  - ‘Convenience initializer’ en nihayetinde bir ‘designated initializer’ çağırmalıdır

Initializer Chaining

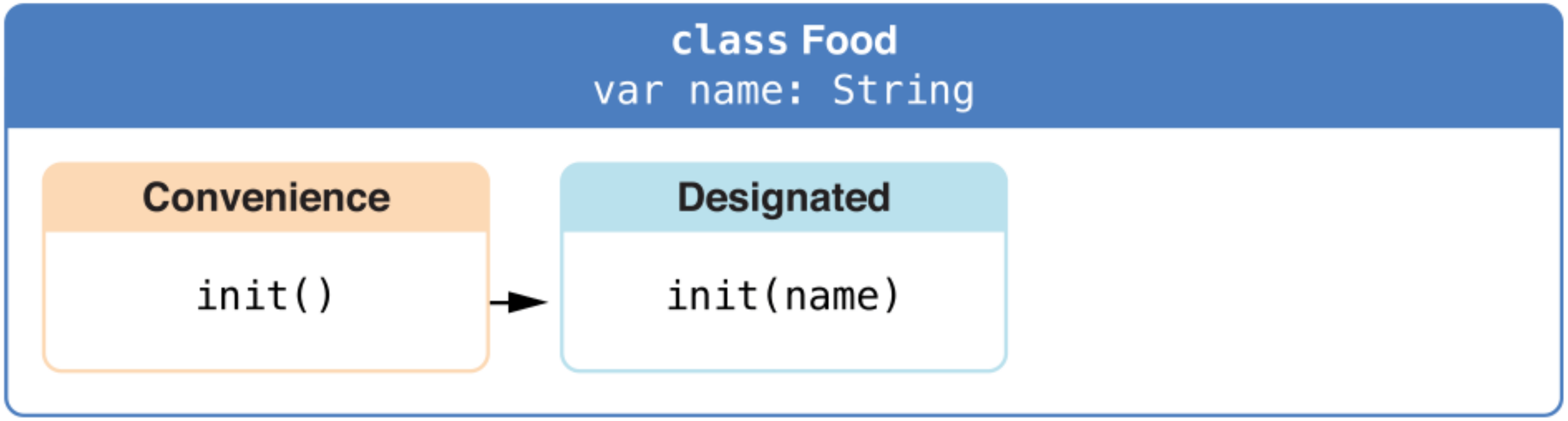



Designated
Convenience
Convenience
Convenience
Designated
Convenience
Convenience
Designated
Convenience
Designated
Designated
Convenience

Initializers In Action

```
class Food {
    var name: String
    init(name: String) {
        self.name = name
    }
    convenience init() {
        self.init(name: "[Unnamed]")
    }
}
```

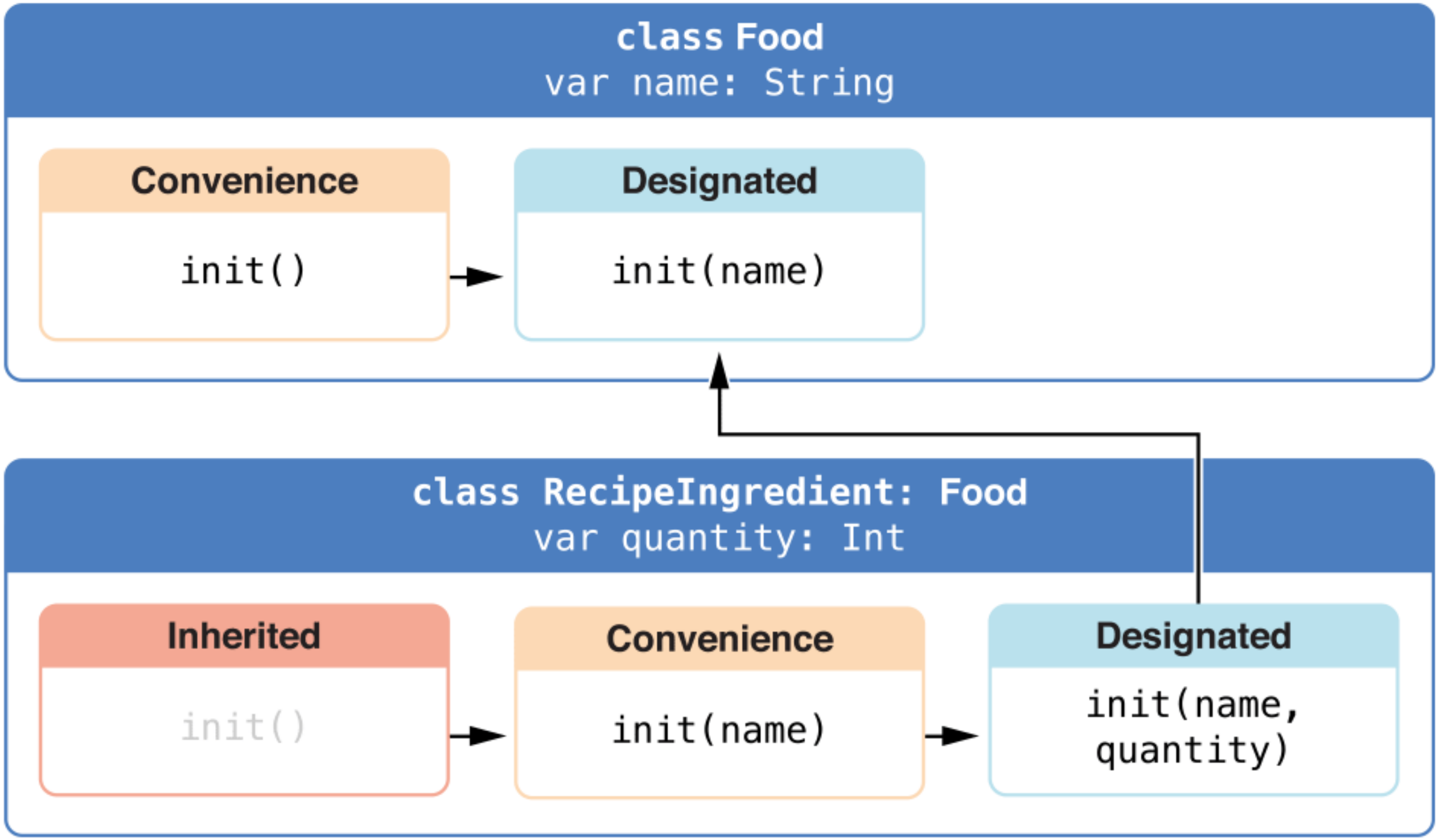
Initializers In Action



Initializers In Action

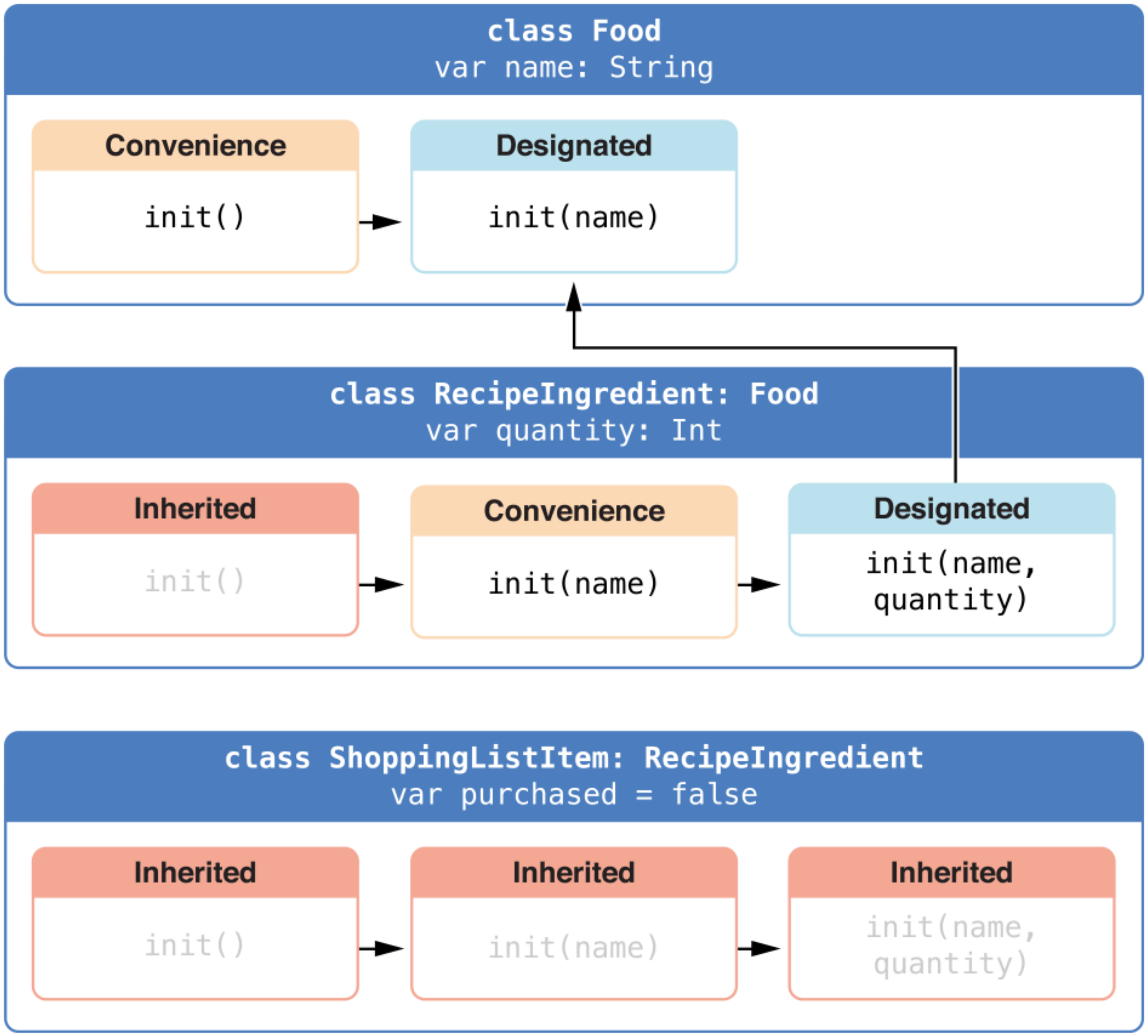
```
class RecipeIngredient: Food {
    var quantity: Int
    init(name: String, quantity: Int) {
        self.quantity = quantity
        super.init(name: name)
    }
    override convenience init(name: String) {
        self.init(name: name, quantity: 1)
    }
}
```

Initializers In Action



Initializers In Action

```
class ShoppingListItem: RecipeIngredient {
    var purchased = false
    var description: String {
        var output = "\(quantity) x \(name)"
            output += purchased ? " ✔" : " ✘"
            return output
    }
}
```


class Food
var name: String
Convenience
init()
Designated
init(name)
class RecipeIngredient: Food
var quantity: Int
Inherited
init()
Convenience
init(name)
Designated
init(name, quantity)
class ShoppingListItem: RecipeIngredient
var purchased = false
Inherited
init()
Inherited
init(name)
Inherited
init(name, quantity)

Initializers In Action

```
var breakfastList = [
    ShoppingListItem(),
    ShoppingListItem(name: "Bacon"),
    ShoppingListItem(name: "Eggs", quantity: 6),
]
breakfastList[0].name = "Orange juice"
breakfastList[0].purchased = true
for item in breakfastList {
    println(item.description)
}
// 1 x Orange juice ✔
// 1 x Bacon ✘
// 6 x Eggs ✘
```

Required Initializers

```
class SomeClass {
    required init() {
        // initializer implementation goes here
    }
}
```

Required Initializers

```
class SomeSubclass: SomeClass {
    required init() {
        // subclass implementation of required initializer
    }
}
```

Deinitialization

- ‘Deinitializer’ method, class hafızadan silinmeden hemen önce çalışır. (Yani deallocation öncesi)
- deinit keyword’ü kullanılır
- Sadece class’larda vardır.

Deinitializer

```
deinit {
    // perform the deinitialization
}
```

Deinitializer

- Parametre almaz
- Parantez kullanılmaz
- Otomatik olarak çağrılır, biz çağıramayız
- Çağrılma sırasında henüz class yok olmadığı için bütün property’lerine erişebilir

Deinitializer In Action

```
struct Bank {
    static var coinsInBank = 10_000
    static func vendCoins(var numberOfCoinsToVend: Int) -> Int {
        numberOfCoinsToVend = min(numberOfCoinsToVend, coinsInBank)
        coinsInBank -= numberOfCoinsToVend
        return numberOfCoinsToVend
    }
    static func receiveCoins(coins: Int) {
        coinsInBank += coins
    }
}
```

Deinitializer In Action

```swift
class Player {
    var coinsInPurse: Int
    init(coins: Int) {
        coinsInPurse = Bank.vendCoins(coins)
    }
    func winCoins(coins: Int) {
        coinsInPurse += Bank.vendCoins(coins)
    }
    deinit {
        Bank.receiveCoins(coinsInPurse)
    }
}
```

Deinitializer In Action

```
var playerOne: Player? = Player(coins: 100)
println("Player joined with \(playerOne!.coinsInPurse) coins")
// prints "A new player has joined the game with 100 coins"
println("There are now \(Bank.coinsInBank) coins left in the bank")
// prints "There are now 9900 coins left in the bank"
```

Deinitializer In Action

```
playerOne!.winCoins(2_000)
println("Player won 2000. Now has \(playerOne!.coinsInPurse) ")
// prints "PlayerOne won 2000 coins & now has 2100 coins"
println("The bank now only has \(Bank.coinsInBank) coins left")
// prints "The bank now only has 7900 coins left"
```

Deinitializer In Action

```
playerOne = nil
println("PlayerOne has left the game")
// prints "PlayerOne has left the game"
println("The bank now has \(Bank.coinsInBank) coins")
// prints "The bank now has 10000 coins"
```