Struct & Class (1)

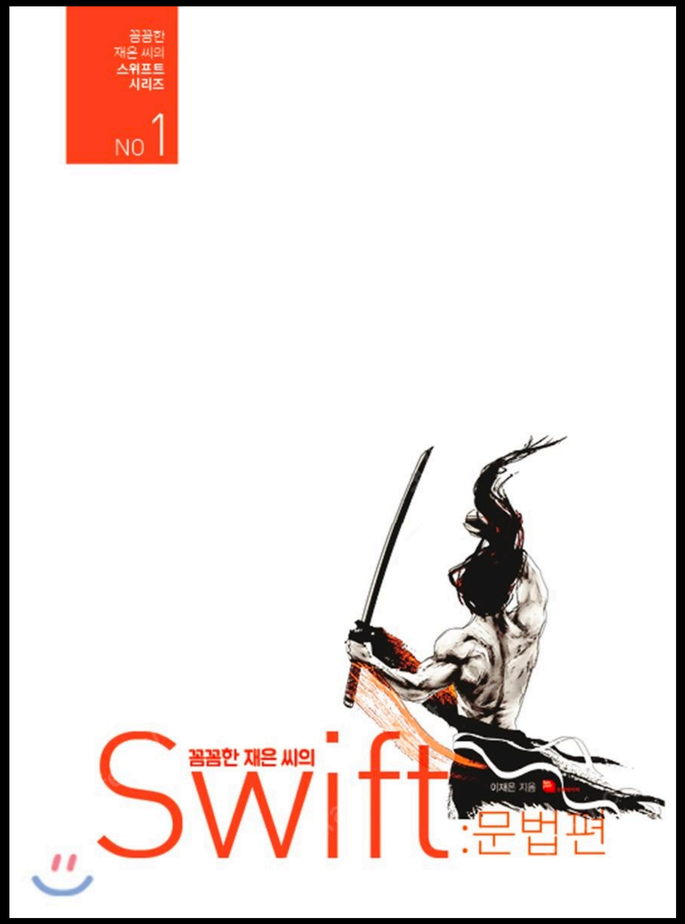
G4ENG

8 JUL 19

Agenda

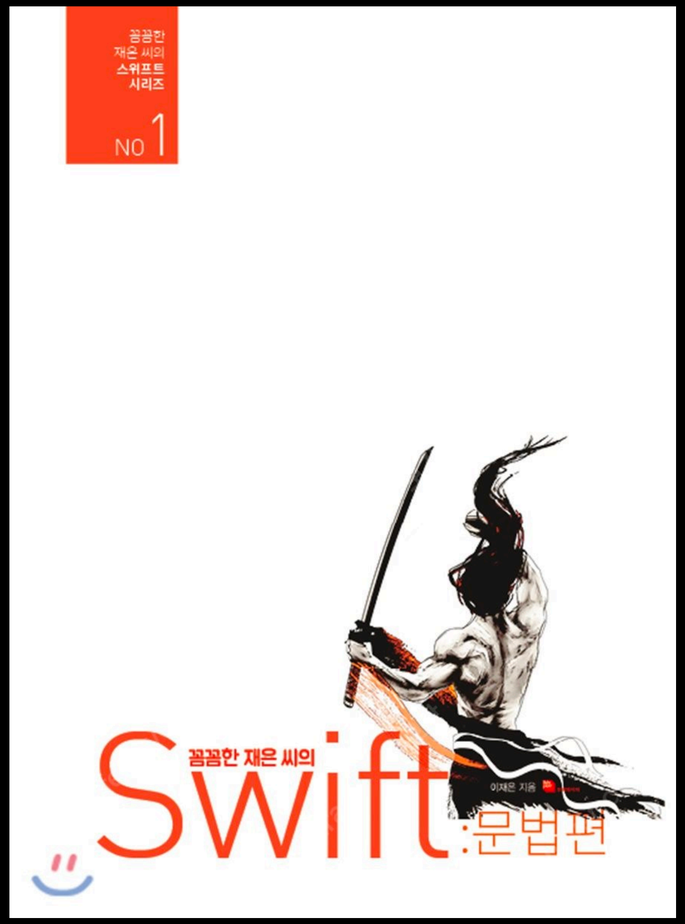
- What do I do in holiday
- Common
- Difference
- Property
- Method

What do?

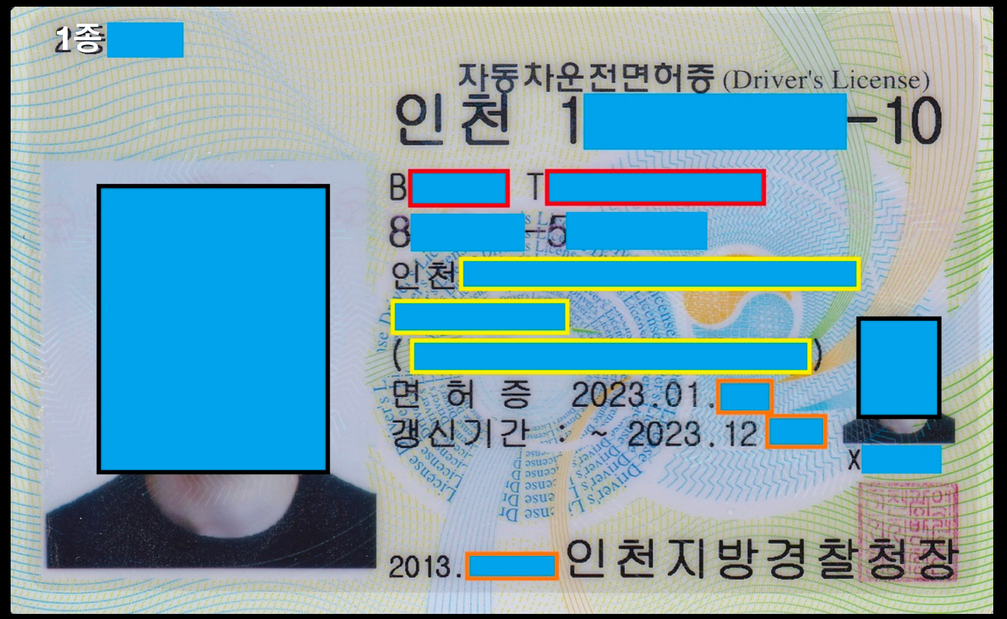


한번 다 읽음

What do?

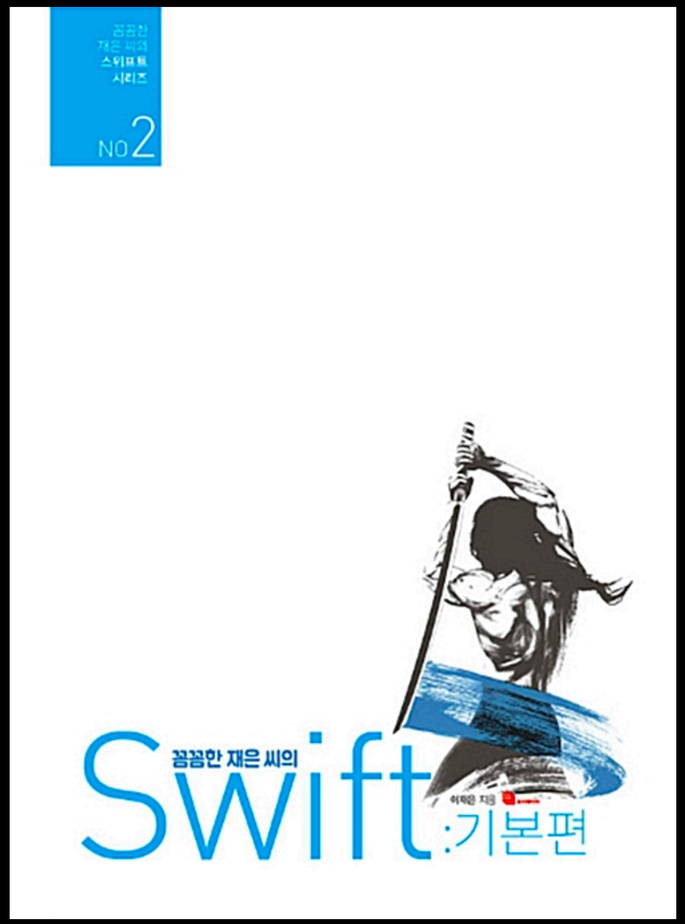


한번 다 읽음



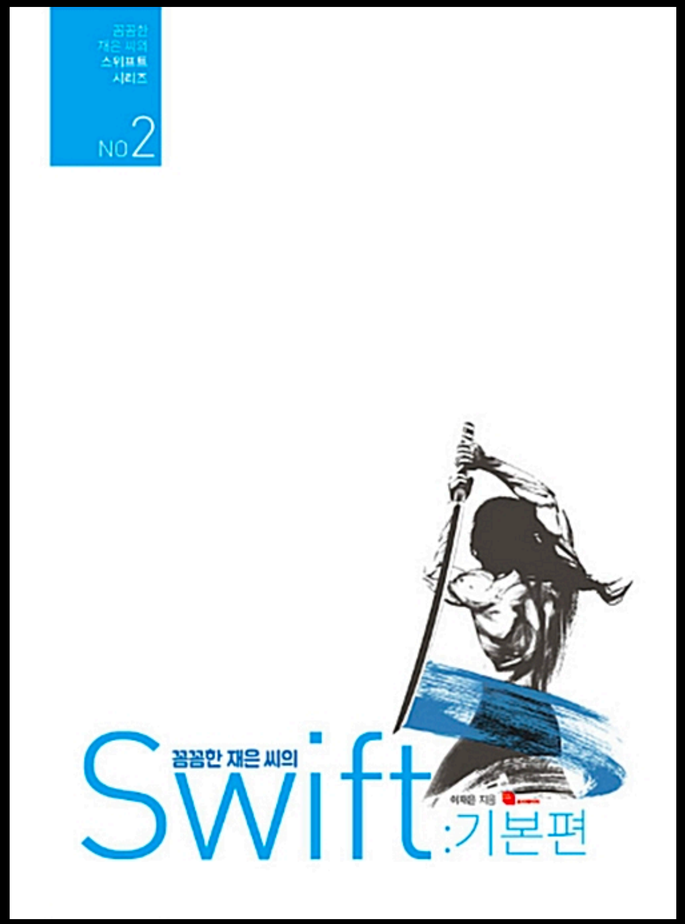
면허따는 중... 이번주 안에 땀...

What do?



이거 보는 중...

What do?



이거 보는 중...

이거 신청함...

Common

- 하나의 큰 코드 블록
- 변수, 상수 -> 프로퍼티
- 함수 -> 메소드
- 서브스크립트, 초기화 구문, 익스텐션, 프로토콜

Difference

- 상속 -> 서브클래싱
- 타입 캐스팅 -> 업 & 다운
- 소멸화 구문
- Call by ~

Property

- 저장 프로퍼티
  - 입력된 값을 저장하거나 저장된 값을 제공하는 역할
  - 상수, 변수에 저장 가능
- 연산 프로퍼티
  - 특정 연산을 통해 값을 만들어 제공 (메소드와 형식이 비슷)
  - 변수에만 정의 가능

Property

```
struct Rect1 {
    // 사각형이 위치한 기준 좌표(좌측 상단 기준)
    var origin = Position()
    // 가로 세로 길이
    var size = Size()
    // 사각형의 X 좌표 중심
    var center: Position {
        get {
            let centerX = self.origin.x + (self.size.width / 2)          (2 times)
            let centerY = self.origin.y + (self.size.height / 2)         (2 times)
            return Position(x: centerX, y: centerY)                      (2 times)
        }
        set(newCenter) {
            self.origin.x = newCenter.x - (size.width / 2)               Position
            self.origin.y = newCenter.y - (size.height / 2)              Position
        }
    }
}

let p = Position(x: 0.0, y: 0.0)                                         Position
let si = Size(width: 10.0, height: 10.0)                                 Size

var square1 = Rect1(origin: p, size: si)                                 Rect1
print("\(square1.center.x), \(square1.center.y)")                        "5.0, 5.0\n"
```

Property

- 인스턴스 프로퍼티
  - 인스턴스에 소속되는 프로퍼티 -> 인스턴스를 만들어야 호출할 수 있음
- 타입 프로퍼티 (static or class)
  - 클래스와 구조체 자체에 소속되어 값을 가지는 프로퍼티 -> 자체적으로 호출

Property

```
struct Foo {
    // 타입 저장 프로퍼티
    static var sFoo = "구조체 타입 프로퍼티값"

    // 타입 연산 프로퍼티
    static var cFoo: Int{
        return 1
    }
}

class Boo {
    // 타입 저장 프로퍼티
    static var sFoo = "클래스 타입 프로퍼티값"
    // 타입 연산 프로퍼티
    static var cFoo: Int {
        return 10
    }

    // 오버라이드 가능한 타입 연산 프로퍼티
    class var oFoo: Int {
        return 100
    }
}
```

10

Property Observer

- willset{ }
  - 값이 변경되기 전에 실행할 내용
- didset{ }
  - 값이 변경된 직후에 실행할 내용

Property Observer

```
struct Job {
    var income: Int = 0 {
        willSet(newIncome) {
            print("이번 달 월급은 \(newIncome)")
        }
        didSet {
            if income > oldValue {
                print("\(income - oldValue)원 증가함")
            }
            else {
                print("월급 삭감됨")
            }
        }
    }
}
```

Method

- 인스턴스 메소드
  - 프로퍼티와 마찬가지로 인스턴스 생성해서 호출
- 타입 메소드
  - static or class -> 객체 자체에서 호출하는 메소드

Method

```swift
struct Resol {
    var width = 0, height = 0

    // 구조체의 요약된 설명을 리턴해주는 인스턴스 메소드
    func desc() -> String {
        let desc = "이 해상도는 가로 \(self.width) X \(self.height)"
        return desc
    }
}

class ViMode {
    var resolution = Resol()
    var interlaced = false
    var frameRate = 0.0
    var name: String?

    // 클래스의 요약된 설명을 리턴해주는 인스턴스 메소드
    func desc() -> String {
        if self.name != nil {
            let desc = "\(self.name!) is \(self.frameRate)"
            return desc
        }
        else {
            let desc = "\(self.frameRate)"
            return desc
        }
    }
}
```

Method

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

    mutating func moveByX(x deltaX: Double, y deltaY: Double) {
        self.x += deltaX
        self.y += deltaY
    }
}

var point = Point(x: 10.5, y: 12.0)                                  Point
point.moveByX(x: 3.0, y: 4.5)                                        Point
print("point.x = \(point.x), point.y = \(point.y)")                  "point.x = 13.5, p...

// 이와 달리 클래스의 인스턴스 메소드에서는 프로퍼티를 수정할 때 별도의 키워드를 필요로 하지 않음.
class Location {
    var x = 0.0, y = 0.0

    func moveByX(x deltaX: Double, y deltaY: Double) {
        self.x += deltaX
        self.y += deltaY
    }
}
var loc = Location()                                                 Location
loc.x = 10.5; loc.y = 12.0; loc.moveByX(x: 3.0, y: 4.5)              (3 times)
print("(\(loc.x),\(loc.y))")                                        "(13.5,16.5)\n"
```

Method

```
class Foo1 {
    // 타입 메소드 선언
    class func fooTypeMethod() {
        // 타입 메소드의 구현 내용이 여기에 들어감
    }
}

let f1 = Foo1()
// f1.fooTypeMethod() // 에러
Foo1.fooTypeMethod()
```