
옵셔널

nil이란?

```
class Person{  
    var name:String  
  
    init(name:String) {  
        self.name = name  
    }  
}
```

```
let person1:Person = Person(name:"joo")  
print(person1.name)
```

결과 : "joo"

nil이란?

```
class Person{  
    var name:String  
  
    init(name:String) {  
        self.name = name  
    }  
}
```

```
let person2:Person  
print(person2.name)
```

결과 : ????

nil이란?

```
class Person{  
    var name:String  
  
    init(name:String) {  
        self.name = name  
    }  
}
```

```
let person2:Person  
print(person2.name)
```

변수에 선언후 초기화를 하지 않은 상태는 nil입니다.

Type Safety

- nil인 상태에서 속성을 참조하거나, 함수를 실행시 발생하는 error로 인한 코드의 불안정성 내포
- Swift의 중요한 특징 중 하나는 Safety!!
- Type Safety를 위해 컴파일러 수준의 nil 체크
- 만약 nil인 변수 선언을 해야할 경우 optional을 사용한다.
- optional은 두가지 가능성을 가질수 있는데
한개는 값이 있음을 나타내고
또다른 한가지는 nil일 가능성을 내포하고 있다.(?기호 사용)

옵셔널 타입

!

```
var person: Person!  
var name: String!  
var age: Int!  
var subjects: [Int]!
```

해당 변수에는 값이 존재 한다.

?

```
var person: Person?  
var name: String?  
var age: Int?  
var subjects: [Int]?
```

해당 변수가 값이 있을수도 있고, nil일수도 있다!

옵셔널 타입

!

```
var person: Person!  
person = Person()
```

```
person.run()
```

만약 person가 초기화가 안되어 있다면! 프로그램이 멈춘다.

?

```
var person: Person?  
person = Person()
```

```
person?.run()
```

만약 person가 초기화가 안되어 있다면! run()이 실행이 안된다.

```
person!.run()
```

만약 person가 초기화가 안되어 있다면! 프로그램이 멈춘다.

옵셔널 타입 테스트

- 일반 변수 VS 옵셔널 변수

- ?와 !의 차이

Unwrapping

Optional 변수에 값이 있음을 확인하여 일반 변수로 전환해준다.

- Forced Unwrapping

- Optional Binding

- Early Exit

강제 해제(Forced Unwrapping)

```
func testFuc(optionalStr:String?)  
{  
    if optionalStr != nil  
    {  
        let unwrapStr:String = optionalStr!  
        print(unwrapStr)  
    }  
}
```

선택적 해제(Optional Binding)

```
func testFuc(optionalStr:String?)  
{  
    if let unwrapStr:String = optionalStr  
    {  
        print(unwrapStr)  
    }  
}
```

선택적 해제 - 예제

```
func isNumber(inputNum:String) -> Bool
{
    if let firstNumber = Int(inputNum)
    {
        print("\(firstNumber)")
        return true
    }else
    {
        return false
    }
}
```

선택적 해제 - 예제

*문제 : inputNum이 한개가 아닌 두개라면?

```
func isNumber(inputNum1:String, inputNum2:String) -> Bool
{

}
}
```

선택적 해제 - 예제

```
func isNumber(inputNum1:String, inputNum2:String) -> Bool
{
    if let firstNumber = Int(inputNum1), let secondNumber =
Int(inputNum1)
    {
        return true
    }else
    {
        return false
    }
}
```

* (,) 콤마를 통해 옵셔널 바인딩을 추가하고, 또 조건도 추가 할수 있다.

Early Exit - guard문

```
guard 조건값 else
{
    //조건값이 거짓일때 실행
    //종료 조건이 항상 필요
}
```

Early Exit

```
func testFuc(optionalStr:String)
{
    guard let unwrapStr:String = optionalStr else
    {
        return
    }

    print(unwrapStr)
}
```


실습 : 친구 리스트 만들기

1. friendList(배열) 옵셔널 변수 만들기
2. func addFriend(name:String) 만들기
3. printFriendList() 함수 만들기