Swift 프로그래밍

2. 데이터타입과 연산자

Tacademy

CONTENTS

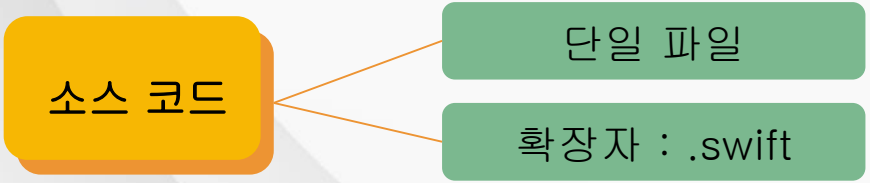
학습 목표

- 변수와 상수를 이용해서 데이터를 다루고 그 차이를 이해할 수 있다.
- 기본 연산자, Swift에서 금지된 연산자, Swift의 신상 연산자를 이해할 수 있다.
- 기본 데이터 타입을 이해할 수 있다.

1. 기본 코드 작성 방법

코드 작성

❖ 소스코드



❖ 문장 작성

- 세미콜론 생략 가능
- 여러 문장을 한 줄로 작성 때 사용

코드 작성

❖ 프레임워크 사용

- **import** [프레임워크]

❖ 다른 소스 코드

- 모듈 단위로 로딩
- 소스 코드 **include/import** 안 함

❖ **main**

- **main** 함수 없음
- **main.swift**의 **top**영역에 작성

콘솔에 정보 출력

❖ 콘솔에 정보 출력 함수

```
print()
```

- 콤마(,)를 이용해서 다수의 값 출력 가능
- 평가 결과 출력 : \()

```
var str = “Hello, playground”
print("Hello, \(str)”)
```

2. 변수와 상수

변수와 상수

- 데이터 다루기 - 변수, 상수
- 변수
  - 처음 값이 대입된 이후에 값이 변할 수 있다.
- 상수
  - 처음 값이 대입된 이후에 값을 변경할 수 없다.
- 변경 가능한 데이터인지 먼저 정할 것

변수

❖ 변수 : 변경 가능

- 변경 가능
- **var**로 선언

❖ 타입 정보

- 선언 생략 가능
- 대입되는 값에서 추론

```
var i = 1
var f : Float = 1.1
i = 3
```

선언과 초기값

❖ 변수 선언과 초기값 분리

- 변수 선언과 초기값 대입 분리 가능
- 분리 시 타입 선언 생략 불가

```
var intVal : Int
intVal = 10
```

타입 선언과 자주 접하는 에러

- 타입 정보가 있으면 타입 선언 생략 가능
- 타입 정보가 없으면 타입 선언 생략 불가
- 변수의 타입과 대입 값의 타입이 다르면 에러
- 서로 다른 타입 간 연산 에러

타입 에러

- 타입 ~~혼합~~ 대입 에러

```
var var1 = 10
var1 = 3.14
```

Cannot assign a value of type 'Double' to a value of type 'Int'

- 타입 ~~대입~~ 혼합 에러

```
var intVal = 1
var floatVal = 1.2
var ret = intVal + floatVal
```

doubleVal

doubleVal

Binary operator '+' cannot be applied to operands of type 'Int' and 'Double'

상수

- 변경 불가능
- **let**으로 선언

```
let constant = 123
constant = 456
```

Cannot assign to 'let' value 'constant'

변수/상수 사용 전 초기화

- 자동 초기화 안됨
- 초기화 전에 사용하면 에러

```
var i : Int
i + 10
```

Variable 'i' used before being initialized 2

3. 연산자

산술 연산자

❖ 산술 이항 연산자(Binary Operator)

+, -, *, /, %

정수 % 정수 : 나눈 나머지 (정수)
정수 / 정수 : 나눈 몫(정수)

❖ 산술 단항 연산자(Unary Operator)

+, -

❖ 증감 단항 연산자는 사용 불가

++, --

```
var i = 0
i++                    '++' is unavailable: it has been removed in Swift 3
```

+=, -= 연산자 사용

공백 문자와 연산자

❖ 산술 이항 연산자(Binary Operator)

1 + 2, 1+2

~~1 +2~~

❖ 연산자와 피연산자 사이의 공백

-i : 음수로 바꾸는 단항 연산자

- i : 뺄셈을 위한 이항 연산자

복합 대입 연산자

❖ 대입(=) 연산자와 다른 연산자 결합

- 곱하기 대입 연산자 : ***=**
- 더하기 대입 연산자 : **+=**
- 빼기 대입 연산자 : **-=**
- 나누기 대입 연산자 : **/=**

나머지 대입 연산자 : %=

```
var i = 1
i += 2  // 3
```

비교, 3항 연산자

❖ 비교 연산자

- ==, !=
- >, <, >=, <=
- === reference type에만 사용. 객체가 같은 주소인지 비교

❖ 3항 연산자

- ? (true expression) : (false expression)

```
value = isTrue ? 10 : 20
```

범위 연산자

❖ 닫힌(Closed) 범위

- 1...10 : 1에서 10까지, 10 포함

❖ 반 닫힌(Half Closed) 범위

- 1..<10 : 9까지

논리 연산자

- &&
- ||
- !

❖ 복잡하면 괄호 사용

```
(condition1 && condition2) || condition3 || condition4
```

nil 연산자

❖ nil-coalescing 연산자 : ??

- nil 관련된 연산자
- nil 이란? - 옵셔널을 다루는 장에서 더 보자.
- nil을 다루는 방법 - unwrapping
- ?? 연산자는 nil이 아니면 unwrapping, nil이면 defaultValue

```
value = optionalValue ?? defaultValue
```

4. 기본 데이터 타입(정수, 실수형)

4. 기본 데이터 타입(정수, 실수형)

데이터 타입

- 부울 : **Bool**
- 정수 : **Int, UInt**
- 실수 : **Float, Double**
- 문자, 문자열 : **Character, String**

4. 기본 데이터 타입(정수, 실수 형)

부울형

- ❖ 타입 선언 : **Bool**
  - ◉ 값 : **true, false**

```
var boolVal : Bool = true
```

4. 기본 데이터 타입(정수, 실수형)

정수 타입

❖ 정수형 타입

- **Int**
- **UInt(Unsigned)**

❖ 크기에 따른 정수형 타입

- **Int8, Int16, Int32, Int64(& Unsigned)**

UInt8, UInt16, UInt32, UInt64

4. 기본 데이터 타입(정수, 실수형)

정수 타입

❖ 정수형 타입 크기

- **Int8 : -128 ~ 127**
- **UInt8 : 0 ~ 255**

❖ **Int** 타입

- **Int** 타입 : **32**비트 **- Int32**, **64**비트 **- Int64**
  32비트 CPU환경 64비트 CPU환경

❖ 값의 범위 : **max, min**

- **Int.max**
- **Int.min**

4. 기본 데이터 타입(정수, 실수 형)

타입 크기 에러

❖ 타입 범위 벗어나면 에러

- **Int8 : -128~127**

```
var intVal1 : Int8 = 127
var intVal2 : Int8 = 128
```

Integer literal overflows when stored into 'Int8'

4. 기본 데이터 타입(정수, 실수 형)

타입 호환

- 타입 혼합 사용 에러

```
int8Val + int16Val
```

- 타입 선언과 다른 타입의 값 대입 에러

```
let int32Val : Int32 = int8Val
```

4. 기본 데이터 타입(정수, 실수형)

타입 변환

❖ 정수형 타입 변환

```
Int16(int8Val) + int16Val
```

❖ 다른 타입의 값에서 변환

```
var i1 = Int(3.14)
var i2 = Int("1234")
var i3 = Int(true)     // 1
```

❖ 변환 실패

```
let str = "hello"
Int(str) // nil
```

4. 기본 데이터 타입(정수, 실수형)

정수형 타입

❖ 랜덤 함수

- **func arc4random() → UInt32**
- **func arc4random_uniform(_: UInt32) → UInt32**

```
~~var r1 : Int = arc4random_uniform(10)~~
~~var r2 : Int = arc4random_uniform(intVal)~~
var r3 : Int = Int(arc4random_uniform(10))
var r4 : UInt32 = arc4random_uniform(10)
var r5: UInt32 = Int(arc4random())
```

4. 기본 데이터 타입(정수, 실수형)

실수형

❖ 실수 타입

- **Float, Double**
- 기본 타입 : **Double**

```
var doubleVar = 3.14
var floatVar : Float = 36.5
```

- 타입 변환

```
~~doubleVal + floatVal~~
doubleVal + Double(floatVal)
```

4. 기본 데이터 타입(정수, 실수 형)

타입 정보

❖ 타입 정보 얻기 : type(of : Val)

- 타입 정보

```
let i = 100
type(of: i) // Int

let d = 4.19
type(of: d) // Double
```

- 타입 비교

```
type(of: intVal) == type(of: intVal2)
type(of: intVal) == Int.self
```

학습정리

지금까지 [데이터타입과 연산자]에 대해서 살펴보았습니다.

변수/상수

- 변수: 동작 중에 값이 변경될 때 사용
- 상수: 값이 변경되면 안 될 때 사용
- 변수/상수 선언 시 데이터 타입 선언은 정보가 충분하면 생략 가능

연산자

두 종류의 범위 연산자(..., ..<), 단항 증감 연산자(++/--)는 삭제

기본 데이터 타입

- 정수형과 실수형 데이터 타입
- 데이터 타입 간 호환과 변환