02

02-2. 변수와 타입

Basic Syntax

변수 선언 및 초기화

변수 선언법

- val(혹은 var) 변수명 : 타입 = 값
- val (value)는 Assign-once 변수 (Immutable), var (variable)은 Mutable 변수
- 타입 추론 지원

```
//변수 선언
val data1: Int = 10
val data2 = 20
var data3 = 30

fun main(args: Array<String>) {
    data2 = 40 <-------------------------- 에러
    data3 = 40 <-------------------------- 성공
}
```

변수 선언 및 초기화

변수 초기화

- 변수 선언은 최상위, 클래스 내부, 함수 내부에 선언.
- 최상위 레벨이나 클래스의 멤버 변수는 선언과 동시에 초기화해주어야 한다.
- 함수 내부의 지역 변수는 선언과 동시에 초기화하지 않더라도된다. 초기화한후 사용할 수 있다

```
val topData1: Int//error
var topData2: Int//error

class User {
    val objData1: String//error
    var objData2: String//error

    fun some(){
        val localData1: Int//ok...
        var localData2: String//ok...

        println(localData1)//error

        localData2="hello"//ok...
        println(localData2)//ok...
    }
}
```

변수 선언 및 초기화

상수변수 선언

- 코틀린에서 변수는 프로퍼티(property)이다.
- val로 선언한 변수의 초깃값을 변경할 수는 없지만, 일반적인 상수변수와는 차이가 있다.
- const라는 예약어를 이용해 상수 변수를 만든다.

- 컴파일 타임에 초기화됨
- 기본 타입과 String만 가능
- 최상위 레벨이나 object 선언 내부에서만 사용 가능
- 리터럴 값만 할당 가능

```
const val myConst: Int = 10

//const var myConst2: Int = 10//error

class MyClass {
//    const val myConst3 = 30//error
}
fun some(){
//    const val myConst4= 40//error
}
```

기초 데이터 타입

숫자타입

- 코틀린에서 모든 것은 객체
- Int, Double 등은 클래스이며 이 클래스로 타입을 명시하여 선언한 변수는 그 자체로 객체
- 타입 클래스에 정의된 기초 함수와 프로퍼티를 사용할 수 있습니다.

```
val intData : Int = 10
val result=intData.minus(5)
```

- Int, Double, Float, Long, Short, Byte, Char, Boolean, String, Any, Unit, Nothing 타입을 제공

기초 데이터 타입

숫자타입

Type	Bit width
Double	64
Float	32
Long	64
Int	32
Short	16
Byte	8

- 코틀린의 숫자 타입의 클래스들은 모두 Number 타입의 서브 클래스
- 문자(Characters)는 Number Type이 아니다.
- Number Type에 대한 자동 형 변형(implicit conversions for number)을 제공하지 않는다.

```
val a3: Byte=0b00001011
val a4: Int=123
val a5: Int=0x0F
val a6: Long = 10L
val a7: Double=10.0
val a8: Double=123.5e10
val a9: Float=10.0f
```

기초 데이터 타입

숫자타입

- 숫자 타입에 대입되는 데이터에 언더바(Underscore)를 추가 할수 있다.

```
val oneMillion: Int = 1_000_000
```

기초 데이터 타입

논리, 문자와 문자열 타입

- Boolean 타입은 true, false 값을 표현하기 위한 타입

```
val isTrue1: Boolean = true && false
val isTrue2: Boolean = true || false
val isTrue3: Boolean = !true
```

- Char는 문자를 표현하는 타입
- Char 타입의 데이터는 작은따옴표(' ', single quotes)로 묶어서 표현
- 코틀린의 Char 타입은 Number 타입으로 표현되지 않습니다.

```
val charData='C'

fun check(c: Char) {
    if (c == 1) {//error

    }
}
```

기초 데이터 타입

논리, 문자와 문자열 타입

- String은 문자열을 표현하는 타입
- 문자열은 str[i]로 접근할 수 있는 Char의 집합

```
var str: String = "Hello"
println("str[1] : ${str[1]}")
```

- String 문자열은 큰따옴표(" ", double quote)로 묶거나 큰따옴표 세 개로(" " ", triple quote)로 묶어서 표현
- 큰따옴표로 묶이는 문자열을 escaped string이라 표현하며, 큰따옴표 세 개로 묶이는 문자열을 raw string이라 표현합니다.

```
val str2="Hello ₩n World"
val str3="""Hello
World"""
```

기초 데이터 타입

논리, 문자와 문자열 타입

- 문자열 템플릿(string template) 개념으로 문자열 내에 변수의 데이터나 특정 연산식 결과에 의한 데이터를 $ 기호로 쉽게 포함할 수 있습니다

```
println("result : $name .. ${sum(10)}")
```

- 문자열 내에서 변수는 "$변수명"으로 작성하고, 표현식은 "${표현식}"으로 작성

기초 데이터 타입

Any 타입

- 코틀린 클래스의 최상위 클래스가 Any

```
fun getLength(obj : Any) : Int {
    if(obj is String) {
        return obj.length
    }
    return 0
}

fun main(args: Array<String>) {
    println(getLength("Hello"))
    println(getLength(10))
}
```

기초 데이터 타입

Unit 과 Nothing

- 코틀린에서 제공하는 타입 중 데이터와 관계 없이 특수 상황을 표현하기 위한 Unit과 Nothing 타입
- 제네릭과 관련 있음
- Unit은 흔히 함수의 반환 구문이 없다는 것을 표현하기 위해 사용
- 자바의 void에 해당한다 (정확히 이야기하면 자바의 void와는 차이가 있다)

```
fun myFun1(){ }
fun myFun2(): Unit { }
```

- Nothing은 의미 있는 데이터가 없다는 것을 명시적으로 선언하기 위해 사용하는 타입

```
fun myFun(arg: Nothing?): Nothing {
    throw Exception()
}
```

기초 데이터 타입

타입확인과 캐스팅

- 타입 체크를 위해 is 연산자 이용
- is 연산자를 이용해 타입으로 체크되면 smart cast

```
fun getStringLength(obj: Any): Int? {
    val strData: String = obj//error
    if (obj is String) {
        return obj.length
    }
    return null
}
```

```
fun getStringLength2(obj: Any): Int? {
    if (obj !is String) return null
    return obj.length
}
```

기초 데이터 타입

타입확인과 캐스팅

- 기초 데이터 타입에 대한 자동 형변형(implicit conversions for number)을 제공하지 않는다.
- 기초 타입의 캐스팅은 toXXX() 함수를 이용해 명시적으로 진행

```
var a1: Int =10
var a2 : Double = a1//error
```

```
var a1: Int =10
var a2 : Double = a1.toDouble()//ok~~
```

- toByte(): Byte
- toShort(): Short
- toInt(): Int
- toLong(): Long
- toFloat(): Float
- toDouble(): Double
- toChar(): Char

감사합니다

단단히 마음먹고 떠난 사람은
산꼭대기에 도착할 수 있다.
산은 올라가는 사람에게만 정복된다.

윌리엄 셰익스피어
William Shakespeare