02

02-5. 제어문

Basic Syntax

조건문

if 표현식

- 코틀린에서 if는 표현식(expression)

```
fun main(args: Array<String>) {

    val a = 5
    if (a < 10) println("$a < 10")
    //if - else
    if (a > 0  && a <= 10) {
        println("0 < $a <= 10")
    } else if(a > 10 && a <= 20){
        println("10 < $a <=20")
    }else {
        println("$a > 20")
    }
}
```

```
val result=if (a > 10) "hello" else "world"
```

조건문

if 표현식

- if 문이 표현식으로 사용되면 else 문이 꼭 정의되어야 한다.
- 여러 라인이 작성되는 경우 if 표현식에 의한 데이터는 맨 마지막 라인

```
if(a>10) "hello"//ok
val result2=if(a>10) "hello"//error
```

```
val result2 = if (a < 10) {
    print("hello....")
    10+20
} else {
    print("world...")
    20+20
}
```

```
val result3 = if (a > 10) 20
else if(a > 20) 30
else 10
```

조건문

when 표현식

- C 혹은 자바의 switch 구문과 비슷
- when 은 코틀린에서 표현식
- switch 구문에서는 분기 조건을 정수로만 지정할 수 있지만, when에서는 정수 이외에도 다양한 타입의 데이터를 지정할 수 있으며, 아예 타입 자체를 지정할 수도 있습니다.

```
fun main(args: Array<String>) {
    val a2=1
    when (a2) {
        1 -> println("a2 == 1")
        2 -> println("a2 == 2")
        else -> {
            println("a2 is neither 1 nor 2")
        }
    }
}
```

```
val data1="hello"
when(data1){
    "hello"->println("data1 is hello")
    "world"->println("data1 is world")
    else -> println("data1 is not hello or world")
}
```

조건문

when 표현식

- 여러 값을 조건을 표현

```
when(data2){
    10, 20 -> println("data2 is 10 or 20")
    30, 40 -> println("data2 is 30 or 40")
    some() -> println("data2 is 50")
    30 + 30 -> println("data2 is 60")
}
```

- 특정 범위를 조건으로 명시

```
val data3=15
when(data3){
    !in 1..100 -> println("invalid data")
    in 1..10 -> println("1 <= data3 <= 10")
    in 11..20 -> println("11 <= data3 <== 20")
    else -> println("data3 > 20")
}
```

```
val list= listOf<String>("hello","world","kkang")
val array= arrayOf<String>("one","two","three")
val data4="kkang"
when(data4){
    in list -> println("data4 in list")
    in array -> println("data4 in array")
}
```

조건문

when 표현식

- 다양한 타입의 데이터에 대한 조건

```
fun testWhen(data: Any){
    when(data){
        1 -> println("data value is 1")
        "hello" -> println("data value is  hello")
        is Boolean -> println("data type is Boolean")
    }
}
```

- if-else 의 대체용

```
val data5=15
when {
    data5<=10 -> println("data5 < 10")
    data5>10 && data5<=20 -> println("10 < data5 <= 20")
    else -> println("data5 > 20")
}
```

조건문

when 표현식

- 표현식
- when을 표현식으로 이용할 때 주의할 점은 if와 마찬가지로 else 부분을 생략할 수 없습니다.
- 하나의 조건에 여러 문장을 수행하고자 블록을 지정했으면 블록에서 마지막 문장이 when이 반환하는 결괏값입니다.

```
val data6=3
val result2= when(data6){
    1 -> "1...."
    2 -> "2...."
    else -> {
        println("else....")
        "hello"
    }
}
```

반복문

for 반복문

- 반복 횟수 지정

```
fun main(args: Array<String>) {
    var sum: Int=0
    for(i in 1..10) {
        sum += i
    }
    println(sum)
}
```

- for (i in 1..100) { //... } // 100까지 포함
- for (i in 1 until 100) { //... } // 100은 포함되지 않음
- for (x in 2..10 step 2) { //... } //2씩 증가
- for (x in 10 downTo 1) { //... }//숫자 감소

```
for(i in 1 until 11 step 2){
    println(i)
}
```

반복문

for 반복문

- 컬렉션 타입 이용

```
val list = listOf("Hello", "World", "!")
val sb=StringBuffer()
for(str in list) {
    sb.append(str)
}
```

- index 값을 획득하고자 한다면 indices를 이용

```
val list = listOf("Hello", "World", "!")
for (i in list.indices) {
    println(list[i])
}
```

반복문

for 반복문

- withIndex()을 이용하여 index와 value를 획득

```
val list = listOf("Hello", "World", "!")
for ((index, value) in list.withIndex()) {
    println("the element at $index is $value")
}
```

반복문

while 반복문

```
fun main(args: Array<String>) {
    var x=0
    var sum1=0
    while (x < 10) {
        sum1 += ++x
    }
    println(sum1)
}
```

반복문

break와 continue문, 그리고 라벨

- while, for 등의 반복문을 작성할 때 break와 continue를 이용하여 흐름을 제어

```
fun main(args: Array<String>) {
    var x2=0
    var sum2=0
    while(true){
        sum2 += ++x2
        if(x2==10) break
    }
    println(sum2)

}
```

반복문

break와 continue문, 그리고 라벨

- 라벨은 개발자가 특정 위치를 개발자 마음대로 이름을 정하고 continue나 break 문에서 이 이름으로 지정한 반복문을 벗어나게 할 수 있습니다

```
for (i in 1..3) {
    for (j in 1..3) {
        if (j>1) break
        println("i : $i , j : $j")
    }
}
```

```
aaa@ for (i in 1..3) {
    for (j in 1..3) {
        if (j>1) break@aaa
        println("i : $i , j : $j")
    }
}
```

감사합니다

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

윌리엄 셰익스피어
William Shakespeare