코틀린을 활용한 안드로이드 앱 개발

코틀린 기초문법 이해 – Function, Condition, Loop

한국기술교육대학교
온라인평생교육원

학습내용

1. 함수 선언법의 이해
2. Conditional Expression 이해
3. Loop 조건 설정법 이해

학습목표

1. 함수 선언법을 이해할 수 있다.
2. Conditional Expression에 대해 이해할 수 있다.
3. Loop 조건 설정법을 이해할 수 있다.

1. 함수 선언법의 이해

1 함수 선언

- fun 함수명(매개변수명 : 타입) : 리턴타입 { }

```
fun sum(a: Int, b: Int): Int {
    return a + b
}
```

- 매개변수에는 var, val을 선언할 수 없고 매개변수는 기본으로 val이 적용
- 반환 값이 없을 때는 Unit으로 명시
- Unit은 생략할 수 있으며 함수의 반환 타입이 선언되지 않았다면 기본으로 Unit이 적용

```
fun sum(a: Int, b: Int): Unit {
    //.....
}
```

1. 함수 선언법의 이해

1 함수 선언

- 함수 내 함수 선언 가능

```
fun sum(a: Int, b: Int): Int {
    var sum=0
    fun calSum(){
        for(i in a..b){
            sum += i
        }
    }
    calSum()
    return sum
}
```

- Single expression function

```
fun some(a: Int, b: Int): Int {
    return a + b
}
```

```
fun some(a: Int, b: Int): Int = a  + b
```

```
fun some(a: Int, b: Int) = a  + b
```

- 함수 오버로딩

```
fun some(a: String){
    println("some(a: String) call....")
}
fun some(a: Int){
    println("some(a: Int) call....")
}
fun some(a: Int, b: String){
    println("some(a: Int, b: String) call....")
}
```

1. 함수 선언법의 이해

2 기본 인수와 명명된 인수

- default argument

```
fun sayHello(name: String = "kkang"){
    println("Hello!!"+name)
}
```

- Named argument

```
fun sayHello(name: String = "kkang", no: Int){
    println("Hello!!"+name)
}
fun main(args: Array<String>) {
//    sayHello(10)//error
    sayHello("lee", 20)
    sayHello(no=10)
    sayHello(name="kim", no=10)
}
```

3 가변인수

```
fun <T> varargsFun(a1: Int, vararg array: T){
    for( a in array){
        println(a)
    }
}

fun main(args: Array<String>) {
    varargsFun(10, "hello", "world")
    varargsFun(10, 20, false)
}
```

2. Conditional Expression 이해

1 if 표현식

```
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")
    }
}
```

- 코틀린에서 if는 표현식(expression) 사용 가능
- else 문이 꼭 정의되어야 함
- 여러 라인이 작성되는 경우 if 표현식에 의한 데이터는 맨 마지막 라인

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

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

2. Conditional Expression 이해

2 when 표현식

- C 혹은 자바의 switch 구문과 비슷
- when은 코틀린에서의 표현식

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

- 여러 값의 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..10 -> println("1 <= data3 <= 10")
    in 11..20 -> println("11 <= data3 <== 20")
    else -> println("data3 > 20")
}
```

- 표현식으로 이용 가능

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

3. Loop 조건 설정법 이해

1 For 반복문

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

```
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])
}
```

- withIndex()을 이용하여 index와 value를 획득

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

3. Loop 조건 설정법 이해

1 For 반복문

For문의 조건

- 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

2 While

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

▪ 정리하기

1. 함수 선언법의 이해

Function

- 클래스를 사용하지 않고 변수와 함수로만 구성 가능
- fun 함수명(매개변수명 : 타입) : 리턴타입 { }
- 반환 값이 없을 때는 Unit으로 명시
- Single expression function 지원
- default argument 와 named parameter 지원

2. Conditional Expression 이해

- if는 표현식(expression) 사용 가능
- when 은 Java 의 switch 구문과 비슷하며 코틀린에서 표현식 사용 가능

3. Loop 조건 설정법 이해

- for, while 문으로 반복문을 작성
- Index 값을 획득하고자 한다면 indices를 이용