Kotlin 基本数据类型 Kotlin 循环控制

Kotlin 条件控制

IF 表达式

一个 if 语句包含一个布尔表达式和一条或多条语句。

```
// 传统用法
var max = a
if (a < b) max = b

// 使用 else
var max: Int
if (a > b) {
    max = a
} else {
    max = b
}

// 作为表达式
val max = if (a > b) a else b
```

我们也可以把 IF 表达式的结果赋值给一个变量。

```
val max = if (a > b) {
    print("Choose a")
    a
} else {
    print("Choose b")
    b
}
```

这也说明我也不需要像Java那种有一个三元操作符，因为我们可以使用它来简单实现：

```
val c = if (condition) a else b
```

实例

```
fun main(args: Array<String>) {
    var x = 0
    if(x>0){
        println("x 大于 0")
    }else if(x==0){
```

```
        println("x 等于 0")
    }else{
        println("x 小于 0")
    }

    var a = 1
    var b = 2
    val c = if (a>=b) a else b
    println("c 的值为 $c")
}
```

输出结果为：

```
x 等于 0
c 的值为 2
```

使用区间

使用 in 运算符来检测某个数字是否在指定区间内，区间格式为 `x..y` ：

实例

```
fun main(args: Array<String>) {
    val x = 5
    val y = 9
    if (x in 1..8) {
        println("x 在区间内")
    }
}
```

输出结果为：

```
x 在区间内
```

When 表达式

when 将它的参数和所有的分支条件顺序比较，直到某个分支满足条件。

when 既可以被当做表达式使用也可以被当做语句使用。如果它被当做表达式，符合条件的分支的值就是整个表达式的值，如果当做语句使用， 则忽略个别分支的值。

when 类似其他语言的 switch 操作符。其最简单的形式如下：

```
when (x) {
    1 -> print("x == 1")
    2 -> print("x == 2")
    else -> { // 注意这个块
```

```
        print("x 不是 1 ，也不是 2")
    }
}
```

在 when 中，else 同 switch 的 default。如果其他分支都不满足条件将会求值 else 分支。

如果很多分支需要用相同的方式处理，则可以把多个分支条件放在一起，用逗号分隔：

```
when (x) {
    0, 1 -> print("x == 0 or x == 1")
    else -> print("otherwise")
}
```

我们也可以检测一个值在（in）或者不在（!in）一个区间或者集合中：

```
when (x) {
    in 1..10 -> print("x is in the range")
    in validNumbers -> print("x is valid")
    !in 10..20 -> print("x is outside the range")
    else -> print("none of the above")
}
```

另一种可能性是检测一个值是（is）或者不是（!is）一个特定类型的值。注意： 由于智能转换，你可以访问该类型的方法和属性而无需 任何额外的检测。

```
fun hasPrefix(x: Any) = when(x) {
    is String -> x.startsWith("prefix")
    else -> false
}
```

when 也可以用来取代 if-else if链。 如果不提供参数，所有的分支条件都是简单的布尔表达式，而当一个分支的条件为真时则执行该分支：

```
when {
    x.isOdd() -> print("x is odd")
    x.isEven() -> print("x is even")
    else -> print("x is funny")
}
```

实例

```
fun main(args: Array<String>) {
    var x = 0
    when (x) {
```

```
        0, 1 -> println("x == 0 or x == 1")
        else -> println("otherwise")
    }

    when (x) {
        1 -> println("x == 1")
        2 -> println("x == 2")
        else -> { // 注意这个块
            println("x 不是 1 ，也不是 2")
        }
    }

    when (x) {
        in 0..10 -> println("x 在该区间范围内")
        else -> println("x 不在该区间范围内")
    }
}
```

输出结果：

```
x == 0 or x == 1
x 不是 1 ，也不是 2
x 在该区间范围内
```

when 中使用 **in** 运算符来判断集合内是否包含某实例：

```
fun main(args: Array<String>) {
    val items = setOf("apple", "banana", "kiwi")
    when {
        "orange" in items -> println("juicy")
        "apple" in items -> println("apple is fine too")
    }
}
```

输出结果：

```
apple is fine too
```

← Kotlin 基本数据类型 Kotlin 循环控制 →

点我分享笔记