## Lambda 表达式

val list = listOf("Apple", "Banana", "Orange", "Pear", "Grape")for (fruit in list) {

println(fruit)

}复制代码

**注意**：在这里使用 listOf()函数创建的是一个不可变的集合。在 Java 中没有不可变的集合，但是在 Kotlin 中不可变的集合指的是，该集合中的元素只能用于读取，不能进行添加、修改或者删除。

这么设计的理由和 val、类默认不可继承是一样的，可见 Kotlin 在不可变性方面的控制及其严格。那么如果我们确实需要创建一个可变的集合，可以使用mutableListOf()函数即可。

val list = mutableListOf("Apple", "Banana", "Orange", "Pear", "Grape")

list.add("Watermelon")for (fruit in list) {

println(fruit)

}

前面介绍的 List 集合的用法其实和 Set 一模一样，只需要将创建集合的方法换成 setOf()和 mutableSetOf() 即可。

val set = setOf("Apple", "Banana", "Orange", "Pear", "Grape")for (fruit in set) {

println(fruit)

}

println("==========================")

val mutableSet = mutableSetOf("Apple", "Banana", "Orange", "Pear", "Grape")

mutableSet.add("Watermelon")for (fruit in mutableSet) {

println(fruit)

}

接下来讲解的 Map 和前面的 List 和 Set 有很大的不同，传统的 Map 用法是先创建一个 HashMap 的实例，然后将一个个的键值对添加到 Map 中，比如给每个水果一个对应的编号。

val map = HashMap<String, Int>()

map.put("Apple", 1)

map.put("Banana", 2)

map.put("Orange", 3)

map.put("Pear", 4)

map.put("Grape", 5)复制代码

这种写法与 Java 中的写法相似，但是在 Kotlin 中并不建议使用 put() 和 get() 方法对 Map 进行添加和读取操作，而是更加建议使用一种类似于数组下标的语法结构，比如向 Map 中添加一条数据可以这么写：

map["Apple"] = 1复制代码

从 Map 中读取一条数据可以这么写

val number = map["Apple"]复制代码

因此可以将代码优化为一下形式

val map = HashMap<String, Int>()

map["Apple"] = 1

map["Banana"] = 2

map["Orange"] = 3

map["Pear"] = 4

map["Grape"] = 5复制代码

这样的写法也不是最简便的，在 Kotlin 中提供了一个 mapOf() 和 mutableMapOf() 函数来继续简化 Map 的用法。在 mapOf() 函数中，我们可以直接传入初始化的键值对组合来完成对 Map 集合的创建：

val map = mapOf("Apple" to 1, "Banana" to 2, "Orange" to 3, "Pear" to 4, "Grape" to 5)

// for (entry in map) {

// println(entry.key + "\t" + entry.value)

// }for ((fruit, number) in map) {

println("fruit is " + fruit + ", number is " + number)

}

### 5.2 集合的函数式 API

**需求**：如何在一个水果集合中找到单词最长的哪个水果？

* 传统实现方式

val list = mutableListOf("Apple", "Banana", "Orange", "Pear", "Grape")

var maxLengthFruit = "";for (fruit in list) {

if (fruit.length > maxLengthFruit.length) {

maxLengthFruit = fruit

}

}

println("max length fruit is " + maxLengthFruit)复制代码

* 使用集合 API 实现

val list = mutableListOf("Apple", "Banana", "Orange", "Pear", "Grape")

val maxLengthFruit = list.maxBy { it.length }

println("max length fruit is " + maxLengthFruit)

集合内数据类型必须一致