**Kotlin**是强类型语言,要求所有的变量必须先声明后使用. 声明变量时必须**显式**或**隐式**指定变量的类型（隐式的是指，声明的时候同时初始化，这样编译的时候就可以推断出该变量的类型了，Java 新版本也加了 var 关键字）。

声明变量关键字:**var**和**va l**

声明方式:

|  |
| --- |
| var变量名 : [类型] [= 初始值]  val变量名 : [类型] [= 初始值] |

可以看出var和val并不是变量的类型 而是一个声明 其中:

var 是用来声明一个可变的量 --- 变量

val 是用于声明一个不变的量(只能被赋值一次) --- 常量

变量的声明可以有两种声明方式**显式**和**隐式**

显式 : 通过 :[类型]的形式显示指定该变量的类型

隐式 : 根据变量的初始值kotlin会根据该初始值确定变量的类型

例子:

|  |
| --- |
| // 声明变量时同时指定类型和初始值  var a :Int = 5  val a1 :Int = 5    // 声明变量指定类型，不指定初始值  var b :Int  b = 6  val b1: Int  b1 = 6  // b1 = 8 // 错误，val 变量不能重复赋值    // 声明类型不指定类型，类型由类型推断得出  var c = 7  val c1 = 7 |

val 声明的是常量 Kotlin的常量分为两种

1、**局部范围的常量：**这种常量允许在声明时不指定初始值，只要在第一次使用之前指定初始值即可。

2、**类的常量属性：**这种常量属性既可以在声明时指定初始值，也可以在类或结构体的构造器中指定初始值。

-----------------------------------------------------------------------------------

需要指出的是，由于 Kotlin 程序编译的字节码必须遵守 JVM 规范，因此，如果直接在 Kotlin 程序中定义变量、函数，kotlinc 将会自动生成一个名为 “文件名首字母大写+Kt” 的类，并将变量转换为该类的静态的 getter、setter 方法（其中 val 声明的只有 getter 方法），函数则转换为该类的静态方法。

还有一点需要说明的是，由于 kotlinc 会为包含函数、变量的 Kotlin 程序生成额外的类，这就要求不能在该包下重复定义同名的类。例如，我们定义了一个名为 liang.kt 的Kotlin 程序，且该程序中包含了函数或变量，那么 kotlinc 会自动生成 LiangKt 类，因此就不能在该包下重复定义 LiangKt 类。