2.3 编程之本：变量和函数

编程语言之多，让人眼花缭乱。你可能不知道，世界上一共诞生过 600 多门有记录的编程语言，没有记录的那就更多了。这些编程语言基本上共有的特性就是变量和函数。可以说，变量和函数就是编程语言之本。那么本节我们就来学习一下 Kotlin 中变量和函数的用法。

2.3.1 变量

先来学习变量。在 Kotlin 中定义变量的方式和 Java 区别很大，在 Java 中如果想要定义一个变量，需要在变量前面声明这个变量的类型，比如说 `int a` 表示 a 是一个整型变量，`String b` 表示 b 是一个字符串变量。而 Kotlin 中定义一个变量，只允许在变量前声明两种关键字：`val` 和 `var`。

`val`（value 的简写）用来声明一个不可变的变量，这种变量在初始赋值之后就再也不能重新赋值，对应 Java 中的 `final` 变量。

`var`（variable 的简写）用来声明一个可变的变量，这种变量在初始赋值之后仍然可以再被重新赋值，对应 Java 中的非 `final` 变量。

如果你有 Java 编程经验的话，可能会在这里产生疑惑，仅仅使用 `val` 或者 `var` 来声明一个变量，那么编译器怎么能知道这个变量是什么类型呢？这也是 Kotlin 比较有特色的一点，它拥有出色的类型推导机制。

举个例子，我们打开上一节创建的 LearnKotlin 文件，在 `main()` 函数中编写如下代码：

```
fun main() {
    val a = 10
    println("a = " + a)
}
```

注意，Kotlin 每一行代码的结尾是不用加分号的，如果你写惯了 Java 的话，在这里得先熟悉一下。

在上述代码中，我们使用 `val` 关键字定义了一个变量 a，并将它赋值为 10，这里 a 就会被自动推导成整型变量。因为既然你要把一个整数赋值给 a，那么 a 就只能是整型变量，而如果你要把一个字符串赋值给 a 的话，那么 a 就会被自动推导成字符串变量，这就是 Kotlin 的类型推导机制。

现在我们运行一下 `main()` 函数，执行结果如图 2.6 所示，正是我们所预期的。

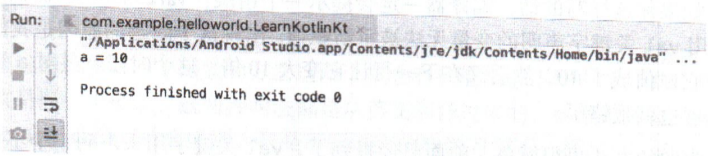

图 2.6 打印变量 a 的值