1、声明变量的方式：

(1)、 val：定义一个常量: val answer =8\*5+2 大小写和Java的区别。

(2)、var:声明可变的变量： var counter = 0 counter = 1

注意：声明值或变量但不做初始化会报错。 和Java区别。

(3)、指定类型方式：

var greeting: String ="Hello"

在Scala中，变量或函数的类型总是写在变量或函数名称的后面。仅当同一行代码中存在多条语句时才需要用分号隔开。

(4)、多个值一起声明：

val xmax,ymax = 100 //将xman和ymax设为100

Java的方式int a=13,b=12;

var greeting, message: String = null

2、常用类型：(Any问题，超类问题)

和Java一样：7种数值类型以及一个Boolean类型。

Byte、Short 、Int、Lang、Float、Double、Char、Boolean

但这些类型是类。Scala不可以区分基本类型和引用类型。你可以对数字执行方法：1.toString()

所以，不需要包装类型。

比如：Scala通过底层的java.lang.String类来表示字符串。但是，它通过StringOps类给字符串追加了上百种操作。 “Hello”.intersect(”World”) //输出”lo”. 表示输出两个字符串共通的一组字符。这里String对象”Hello”隐式的转成了StringOps类对象，接着就可以调用方法intersect了。

所以，StringOps类为String提供了很多方法。

同样：Int --RichInt

Double—RichDouble

Char--RichChar

BigInt和BigDecimal表示 数组，java.math下。

在Scala中，用方法，而不是强制类型转换，来做数值类型之间的转换。比如，99.44.toInt得到99, 99.toChar得到’c’. 当然，和Java一样，toString将任意的对象转换成字符串。

要将包含了数字的字符串转换成数值，使用toInt或toDouble。比如，”99.44”.toDouble得到99.44

3、算术和操作符重载：

val answer = 8 \* 5 + 2

+\_\*/%等操作符完成的是它们通常的工作，位操作符&|^>><<也一样。只是有一点特别的：这些操作符实际上是方法。例如：

a + b 是如下方法调用的简写： a.+(b)

这里的+是一个方法。

通常来说，你可以用 a 方法 b 作为以下代码的简写： a.方法(b)

和Java相比，Scala并没有提供++和—操作符，我们需要使用+=1或者-=1 如： counter += 1

var a,b=12 //> a : Int = 12

//| b : Int = 12

println(a+b) //> 24

println(a.+(b)) //> 24

对于BigInt和BigDecimal对象，你可以以常规的方式使用那些数学操作符：

val x: BigInt = 1234567890

x \* x \* x //将产出1881676371789154860897069000

这比Java好多了，在Java中同样的操作需要写成x.multiply(x).multiply(x).

4、调用函数和方法：



--第六章介绍。



5、apply方法：



