

来自于著名的IDE IntelliJ IDEA(Android Studio基于此开发) 软件开发公司 JetBrains(位于东欧捷克)

起源来自JetBrains的圣彼得堡团队，Kotlin以靠近俄罗斯圣彼得堡的一座岛屿(Kotlin Island)命名，Kotlin的大部分开发团队就在那里。

一种基于JVM的静态类型编程语言

Kotlin 是 JetBrains 在 2010 年推出的基于 JVM 的新编程语言。开发者称，设计它的目的是避免 Java 语言编程中的一些难题。比如：在 Kotlin 中类型系统控制了空指针引用，可以有效避免 Java 中常见的NullPointException。   
作为一个跨平台的语言，Kotlin 可以工作于任何 Java 的工作环境：服务器端的应用，移动应用（Android版），桌面应用程序。

Kotlin 中没有基础类型，数组是定长的，泛型是安全的，即便运行时也是安全的。





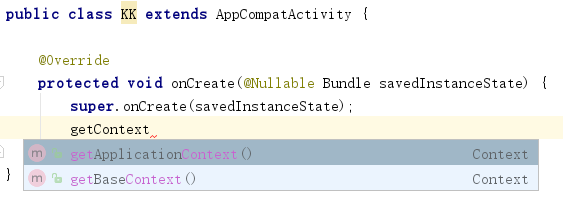
**//** 超级简单的非空判断：

**fun** printStringSize(str: String?) {  
 *println*(str?.*toCharArray*()?.**size**)  
}

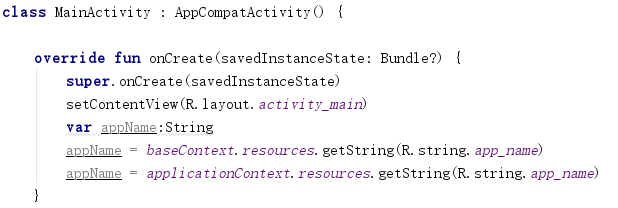
printStringSize(**"aaa"**) // 输出3  
printStringSize(**null**) // 输出null

**var age** = 18 *// 类型Int根据值推导出来***var score** : Int = 98 *// 显式声明Int类型***val TAG** = **"MainActivity"** *// 常量TAG，String类型根据值推导出来  
  
// Java中的main函数***fun** main(args: Array<String>) {  
 *println*(args)  
}  
  
*// data class类型会自动生成getter/setter、hashCode、equals、toString等方法***data class** Student(**var name**: String, **var age**: Int)

不需要参数的get方法变成了类型成员变量的调用，如在Activity中有如下两个方法：



在Kotlin中直接用变量的方式使用，如下：

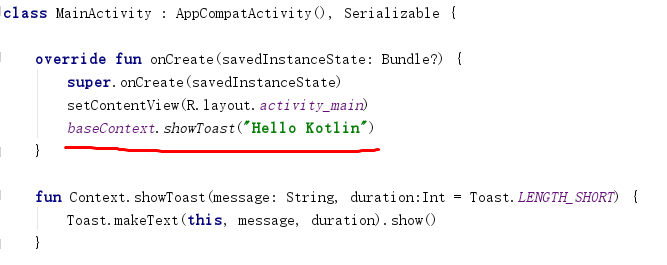


这里resource等于getResource()

实现和继承使用：代替，继承的话需要在类的后面加()，实现的话就不需要，如下面的MainActiivty继承了AppCompatActivity，实现了Serialazible，如下：







如上图，我们声明了一个showToast方法，在这个类中只要是Context就都可以调用这个方法，但是如果在其它的类上的Context则无法调用这个方法。

字符串拼接：

**val** stu= Student(**"张三"**, 38)  
**val** grade = **"三年级"**Log.d(**"MainActivity"**, **"name = ${**stu.**name}; age=${**stu.**age}; grade=$**grade**"**)

类型判断与强转：

if(view is TextView) {

TextView textView = view as TextView

textView.setText("text")// 这里可写成：textView.text = “text”

}

再简单一点：

if(view is TextView) {

(view as TextView).setText("text")

}

再简单一点：

if(view is TextView) {

view.setText("text")

}

when替换switch，如下：

fun testWhen(obj: Any) {

when(obj) {

is Int -> {

println("obj is a int")

println(obj + 1)

}

is String -> {

println("obj is a string")

println(obj.length) // 这里都不需要强转了，直接使用String的length方法

}

else -> {

println("obj is something i don't care")

}

}

}



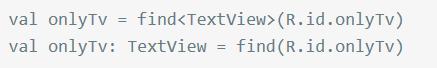
容器的操作符：







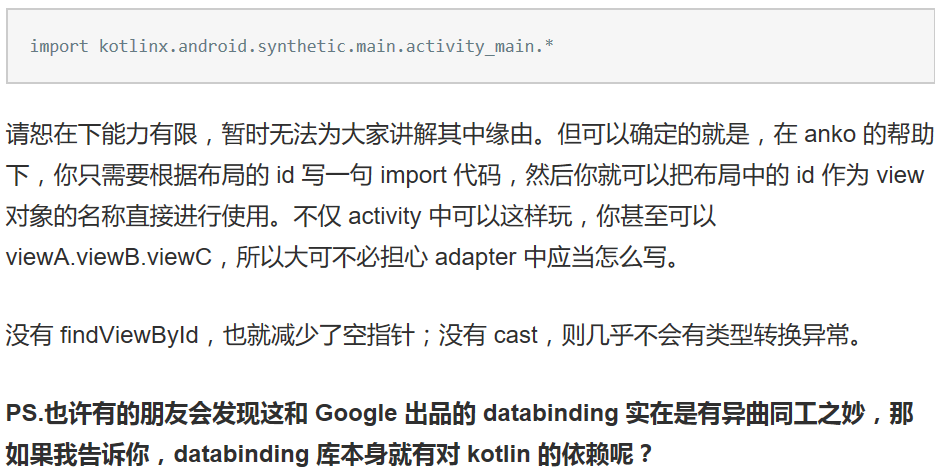
FindViewById，需要anko 库，如下：





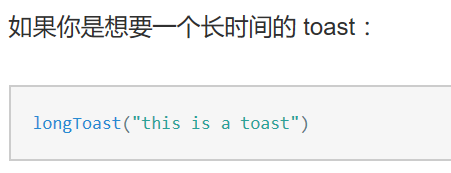












# 与Java交互性好

Kotlin和Java几乎是无缝连接的。

Kotlin可以引用Java的代码，反之亦然。

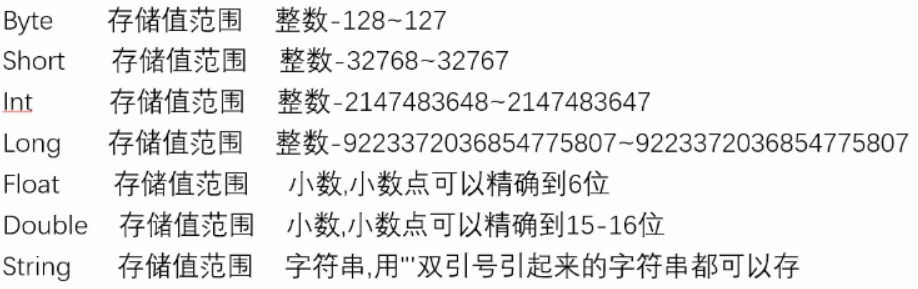
Java文件可以插件转成kotlin

# Kotlin优势





# 数据类型





通过Int.MAX\_VALUE或者Int.MIN\_VALUE即可查看最大最小值，其它类型的取值范围也可用类似的方法查看。

## 类型推断

在声明变量的时候，使用var声明，没有指明变量的类型，这时看赋值的是什么，如var i = 18，则是Int类型，如果给一个很大的数值，则是Long类型。

也可以在声明变量的时候指定类型，如：var i:Int = 18

var a 报错，编译器无法知道a是什么类型，如果在声明时不赋值，则需要显示声明：var a:Int

常量声明：val a = 15

## 二进制表示

var a:Int = 0b11 // a的值为3