Kotlin

# Run,apply,let,also,with的用法和区别

## Run:

* run函数块。返回值为函数块最后一行，或者指定return表达式
* 调用某对象的run函数，在函数块内可以通过 *this* 指代该对象。返回值为函数块的最后一行或指定return表达式

val a = run {

println("run")

return@run 3

}

println(a)

运行结果：

run

3

## Apply

调用某对象的apply函数，在函数块内可以通过 *this* 指代该对象。返回值为该对象自己

## Let

调用某对象的let函数，则该对象为函数的参数。在函数块内可以通过 *it* 指代该对象。返回值为函数块的最后一行或指定return表达式。

## Also

调用某对象的also函数，则该对象为函数的参数。在函数块内可以通过 *it* 指代该对象。返回值为该对象自己

## With

with函数和前面的几个函数使用方式略有不同，因为它不是以扩展的形式存在的。它是将某对象作为函数的参数，在函数块内可以通过 *this* 指代该对象。返回值为函数块的最后一行或指定return表达式。

实例：

val a = with("string") {

println(this) 3

}

println(a)

运行结果：

string

# 双冒号::

Kotlin中将函数作为参数传递，使用双冒号::引用其他方法

# xml中的使用语法

## 每一个变量variable都是由名称name和类型type组成。

name可以在布局文件中使用，也可以通过Setter和Getter在代码中使用。type可以是基本数据类型、集合、适配器、自定义类等，除了基本类型，其他都必须把包名写全

## 双引号中可以套“`”（Tab键上面，非单引号），单引号内可以套双引号；

## 不能直接用boolean和int类型的值。

而且int值将被当做资源文件。因此要使用其数值，必须转换成字符串，如（boolean同理）：   
"@{user.age + ``}"，或 "@{String.valueOf(user.age)}"

## 颜色必须使用ARGB制，指定其所有的透明度和单色值，以前的#F00和#FF0000无效，且前面不能加“#”

## "@{user.nickname ?? user.name}"

代表user.nickname为null时显示user.name，否则显示自己。等同于"@{user.nickname == null ? user.name : user.nickname}"

## 比较运算符，必须使用转义字符

大于号“>”——&gt;   
小于号“<”——&lt;

## 不能直接使用中文（MD），如：： android:text='@{user.male ? "男" : "女"}'  将报错，得用string引用，改成：  android:text='@{user.male ? @string/male : @string/female}'

## 注意事项

### 在type中的泛型同样不能使用”<”与”>”，必须使用转义字符"&lt;"和"&gt;"

### 转义字符在type中会包红色错误，可忽略，编译时能通过

### 不能直接使用android:text="@{userList[0]}"，即使你想打印对象的地址，也不行。这样使用的后果是：每个databing都报“程序包不存在”的错误，连之前正常的也错了。会懵逼的

### 子布局中的(user.male ? `男` : `女`)，必须用小括号括起来

### android:text中的数据组合很长，但也只能写一行，目前还不知如何分行

## 注意