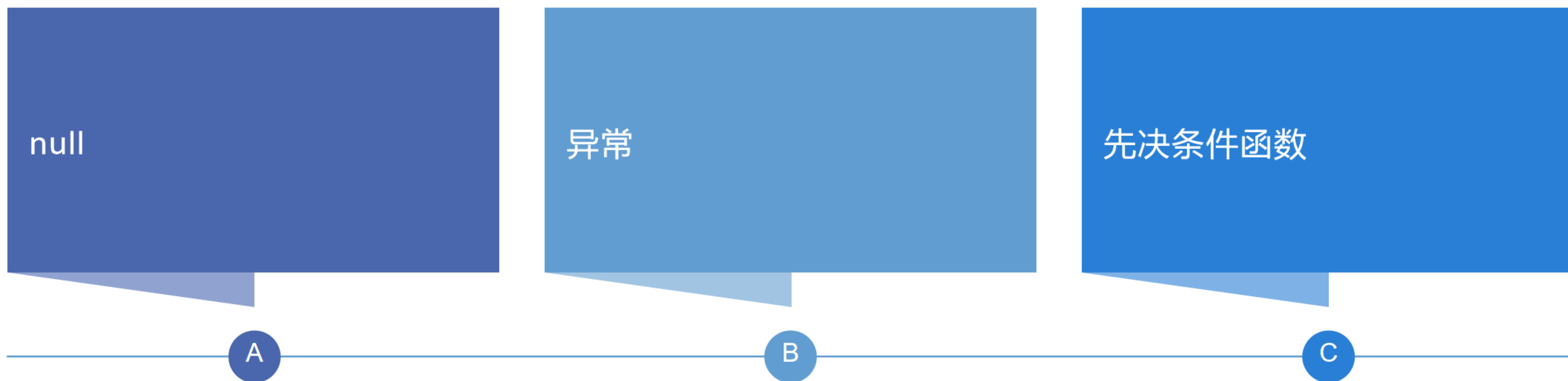


《Android高级课程》

VIP试听课

让人人都能享受到高品质的教育服务

彻底掌握Kotlin



讲师简介



宁传奇

动脑学院讲师合伙人

曾任华为高级工程师、架构师，全栈开发爱好者。

技术这条路，不和你一比高下，只伴你一同成长。

➤ 在Java中我们司空见惯的空指针异常NullPointerException，带给了我们很多麻烦。

Kotlin作为更强大的语言，势必会基于以往的语言设计经验对其进行改良。Kotlin更多地把运行时可能会出现>null问题，以编译时错误的方式，**提前在编译期强迫我们重视起来**，而不是等到运行时报错，防范于未然，提高了我们程序的健壮性。



可空性

- 对于null值问题，Kotlin反其道而行之，除非另有规定，变量不可为null值，这样一来，运行时崩溃从根源上得到解决。

```
1  ▶ fun main() {  
2      val str = "butterfly"  
3      str = null  
4      println(str)  
5  }
```

Kotlin的null类型

- 为了避免NullPointerException，Kotlin的做法是不让我们给非空类型变量赋null值，但null在Kotlin中依然存在。

```
1  ▶ fun main() {  
2      var str:String? = readLine()  
3      str = null  
4      println("input:$str")  
5  }
```

null安全

- Kotlin区分可空类型和非可空类型，所以，你要一个可空类型变量运行，而它又可能不存在，对于这种潜在危险，编译器时刻警惕着。为了应对这种风险，Kotlin不允许你在可空类型值上调用函数，除非你**主动接手安全管理**。

```
1 ▶ fun main() {  
2     val str:String = readLine().capitalize()  
3     println(str)  
4 }
```



选项一：安全调用操作符

➤ 这次Kotlin不报错了，编译器看到有安全调用操作符，所以它知道如何检查null值。

如果遇到null值，它就**跳过函数调用**，而不是返回null。

```
1  ▶ fun main() {  
2      val str:String? = readLine()?.capitalize()  
3      println(str)  
4  }
```



使用带let的安全调用

- 安全调用允许在可空类型上调用函数，但是如果还想做点额外的事，比如创建新值，或判断不为null就调用其他函数，怎么办？可以使用带let函数的安全调用操作符。你可以在任何类型上调用let函数，它的主要作用是让你在指定的作用域内定义一个或多个变量。

```
10 ▶ fun main() {  
11     val str:String? = readLine()?.let { it:String  
12         if(it.isNotBlank()) {  
13             it.capitalize() ^let  
14         } else {  
15             "butterfly" ^let  
16         }  
17     }  
18     println(str)  
19 }
```

选项二：使用非空断言操作符

➤ `!!` 又称感叹号操作符，当变量值为 `null` 是，会抛出 `KotlinNullPointerException`。

```
1  ▶ fun main() {  
2      val str:String = readLine()!!.capitalize()  
3      println(str)  
4  }
```



选项三：使用if判断null值情况

- 我们也可以使用if判断，但是相比之下安全调用操作符用起来更灵活，代码也更简洁，我们可以用安全操作符进行多个函数的链式调用。

```
1  fun main() {  
2      var str:String? = readLine()  
3      if(str != null){  
4          str = str.capitalize()  
5      }else{  
6          println("为null.")  
7      }  
8  
9      str = str?.capitalize()?.plus(" is great.");  
10     println(str)  
11 }
```

使用空合并操作符

- ?:操作符的意思是，如果左边的求值结果为null，就使用右边的结果值。

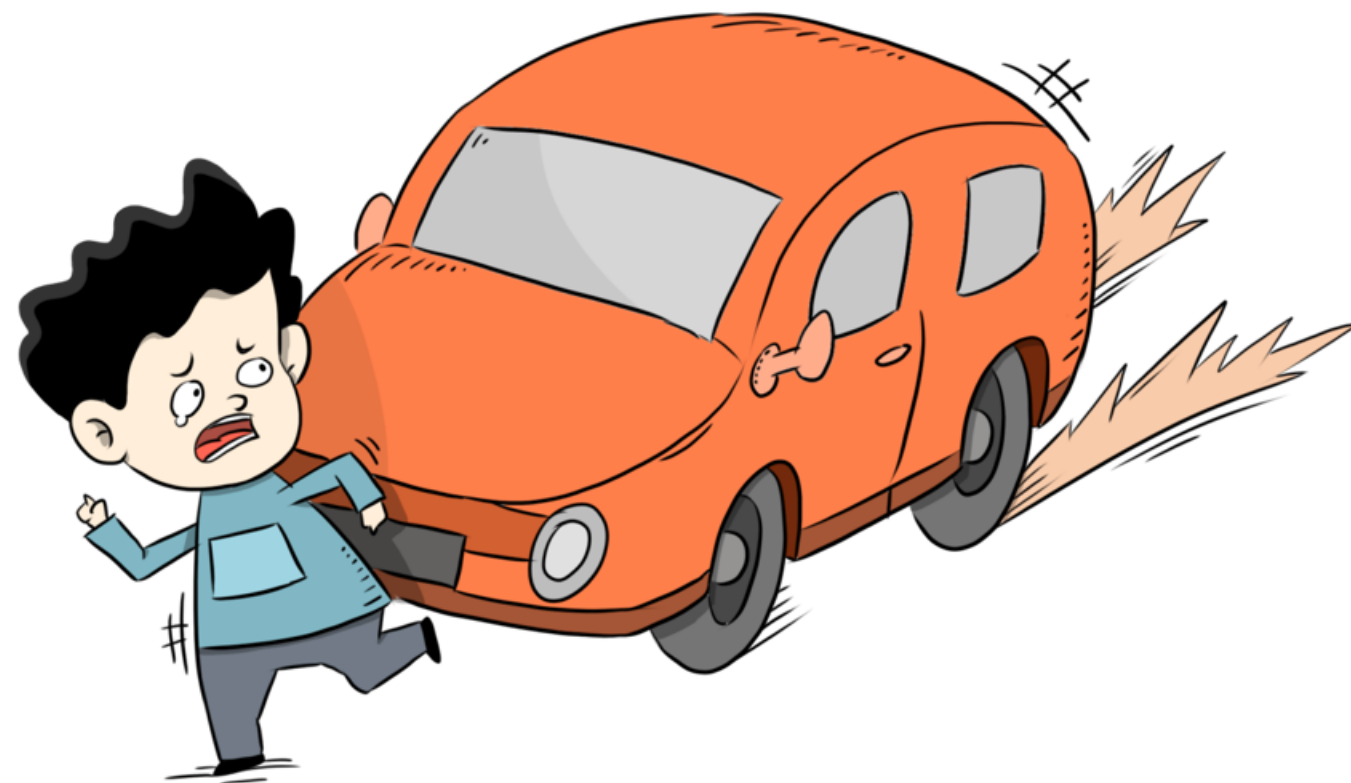
```
val strWithSafe : String = str ?: "butterfly"
```

- 空合并操作符也可以和let函数一起使用来代替if/else语句。

```
17 ▶ fun main() {  
18     var str : String? = readLine()  
19     str = str?.let { it.capitalize() } ?: "butterfly"  
20     println(str)  
21 }
```

异常

- 抛出异常
- 自定义异常
- 异常处理



先决条件函数

- Kotlin标准库提供了一些便利函数，使用这些内置函数，你可以抛出带自定义信息的异常，这些便利函数叫做先决条件函数，你可以用它定义先决条件，条件必须满足，目标代码才能执行。

函数	描述
checkNotNull	如果参数为null，则抛出IllegalStateException异常，否则返回非null值
require	如果参数为false，则抛出IllegalArgumentException异常
requireNotNull	如果参数为null，则抛出IllegalStateException异常，否则返回非null值
error	如果参数为null，则抛出IllegalStateException异常并输出错误信息，否则返回非null值
assert	如果参数为false，则抛出AssertError异常，并打上断言编译器标记