<https://mp.weixin.qq.com/s/SBAdo2zmOjVXPjfzsR1dLA>

# 用Java8的Optional类替代判空结构

# ****1.前言****

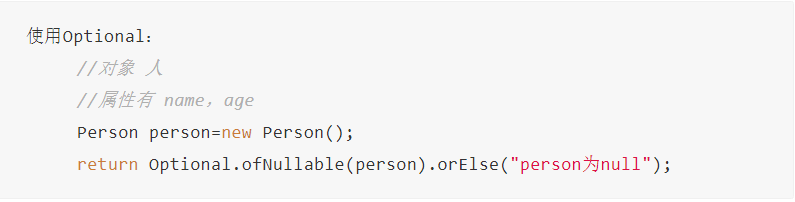
相信不少小伙伴已经被java的NPE(Null Pointer Exception)所谓的空指针异常搞的头昏脑涨,

有大佬说过“防止 NPE，是程序员的基本修养。”但是修养归修养，也是我们程序员最头疼的问题之一，那么我们今天就要尽可能的利用Java8的新特性 Optional来尽量简化代码同时高效处理NPE（Null Pointer Exception 空指针异常）

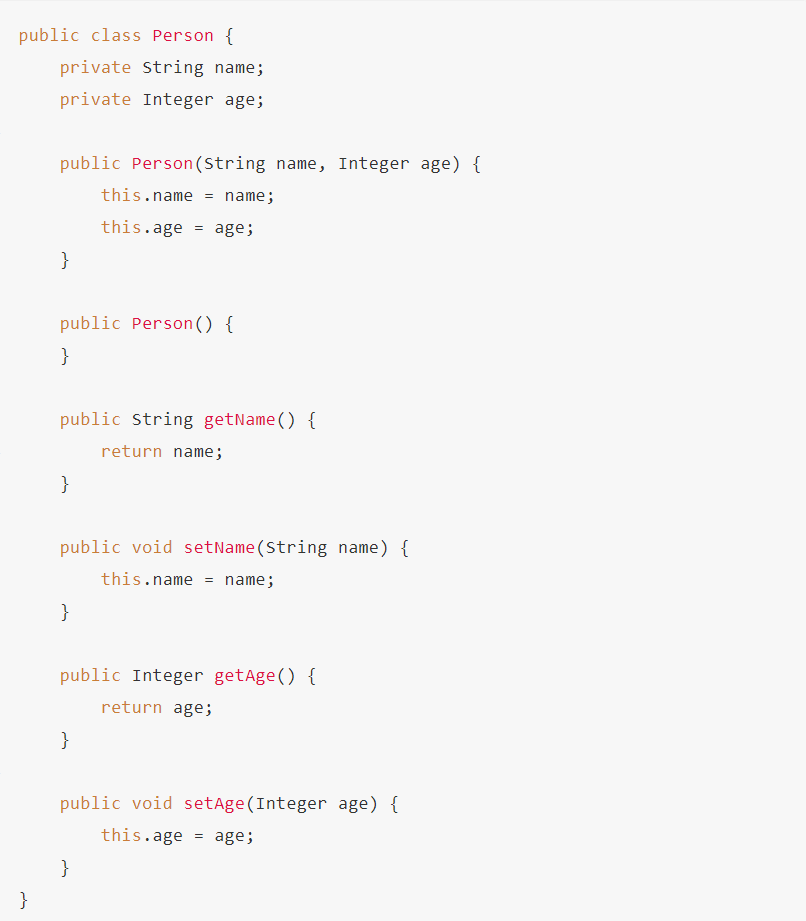
# ****2.认识Optional并使用****

简单来说，Opitonal类就是Java提供的为了解决大家平时判断对象是否为空用 会用 null!=obj 这样的方式存在的判断，从而令人头疼导致NPE（Null Pointer Exception 空指针异常），同时Optional的存在可以让代码更加简单，可读性跟高，代码写起来更高效.





测试展示类Person代码(如果有朋友不明白可以看一下这个)：



下面，我们就高效的学习一下神奇的Optional类！

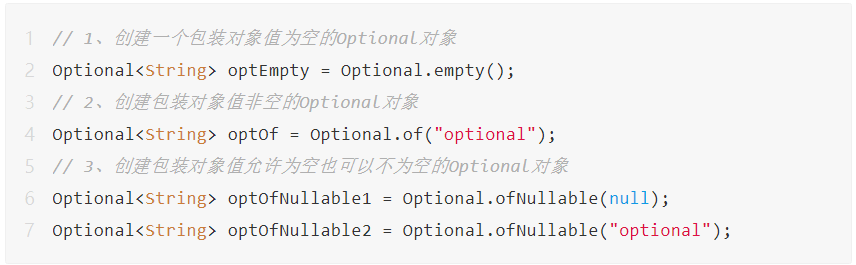
## 2.1 Optional对象创建

首先我们先打开Optional的内部,去一探究竟

先把几个创建Optional对象的方法提取出来



再做一个简单的实例展示  
与上面对应

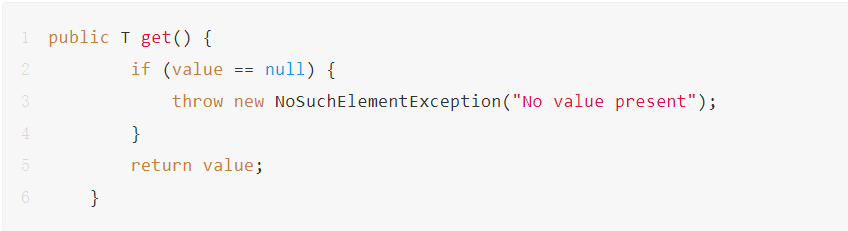


我们关于创建Optional对象的内部方法大致分析完毕  
接下来也正式的进入Optional的学习与使用中

## ****2.2 Optional.get()方法(返回对象的值)****

get()方法是返回一个option的实例值

源码：



也就是如果value不为空则做返回，如果为空则抛出异常 "No value present"

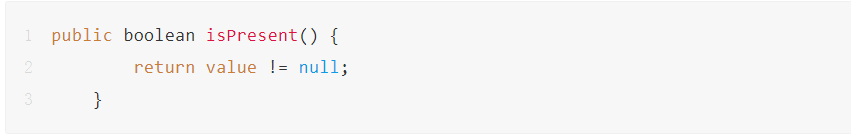
简单实例展示



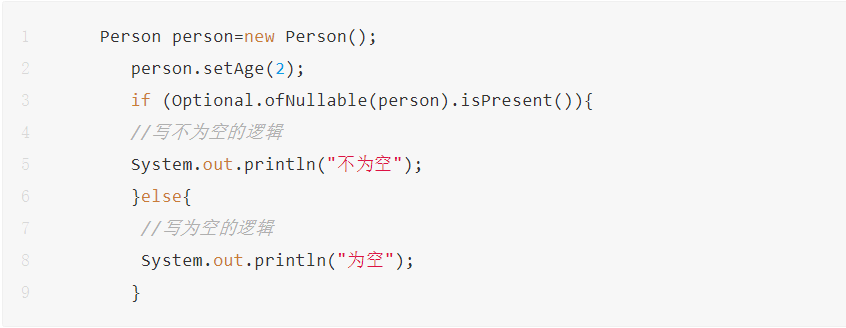
## ****2.3 Optional.isPresent()方法(判读是否为空)****

isPresent()方法就是会返回一个boolean类型值，如果对象不为空则为真，如果为空则false

源码:



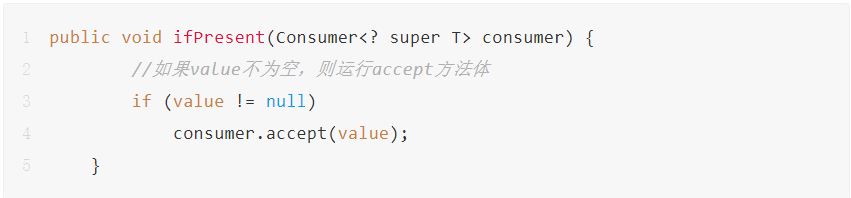
简单的实例展示:



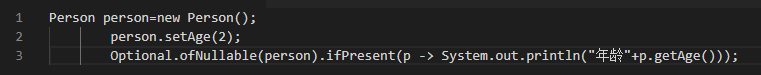
## ****2.4 Optional.ifPresent()方法(判读是否为空并返回函数)****

这个意思是如果对象非空，则运行函数体

源码:



看实例：

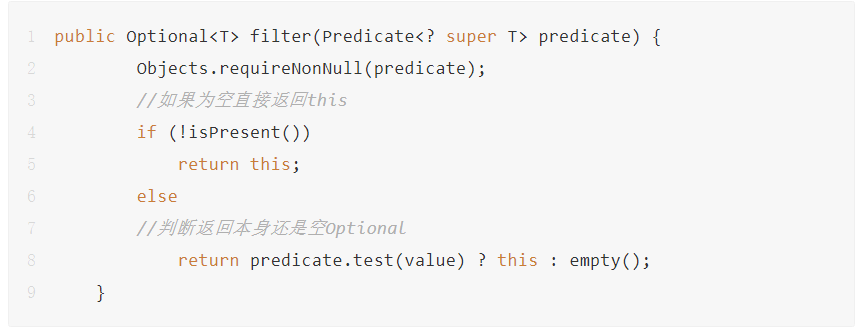


如果对象不为空，则会打印这个年龄，因为内部已经做了NPE（非空判断），所以就不用担心空指针异常了。

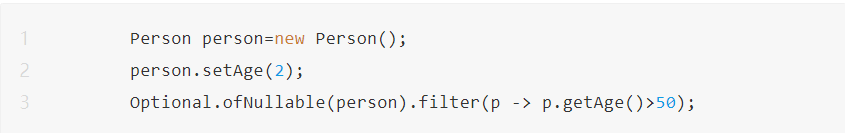
**2.5 Optional.filter()方法(过滤对象)**

filter()方法大致意思是，接受一个对象，然后对他进行条件过滤，如果条件符合则返回Optional对象本身，如果不符合则返回空Optional

源码：



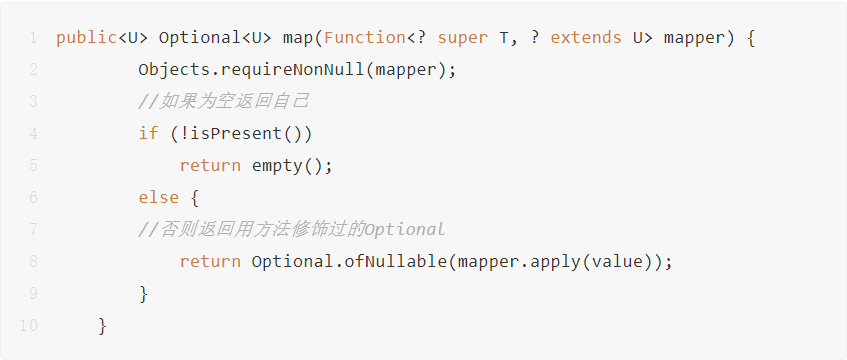
简单实例：



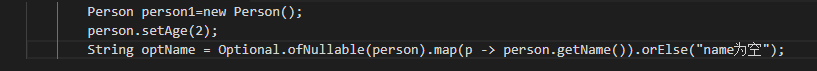
## ****2.6 Optional.map()方法(对象进行二次包装)****

map()方法将对应Funcation函数式接口中的对象，进行二次运算，封装成新的对象然后返回在Optional中

源码：



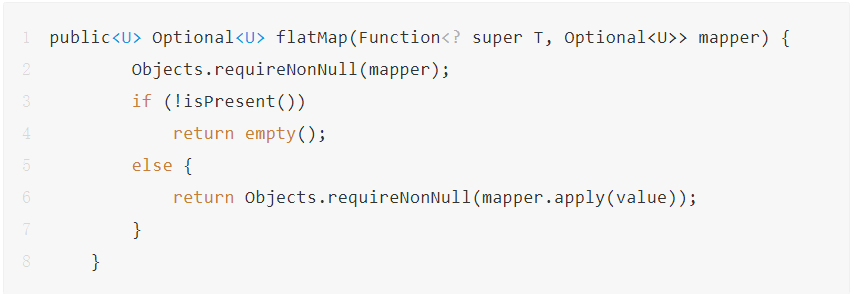
实例展示：



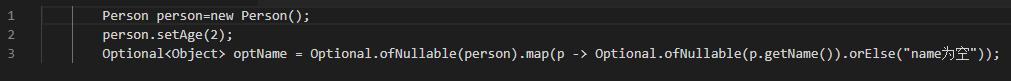
## 2.7 Optional.flatMap()方法(Optional对象进行二次包装)

map()方法将对应Optional< Funcation >函数式接口中的对象，进行二次运算，封装成新的对象然后返回在Optional中

源码:

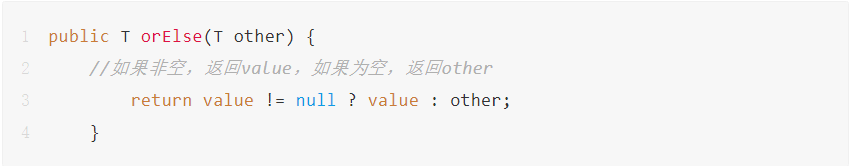


实例：

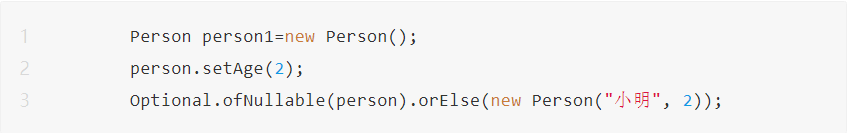


## ****2.8 Optional.orElse()方法(为空返回对象)****

常用方法之一，这个方法意思是如果包装对象为空的话，就执行orElse方法里的value，如果非空，则返回写入对象  
源码:

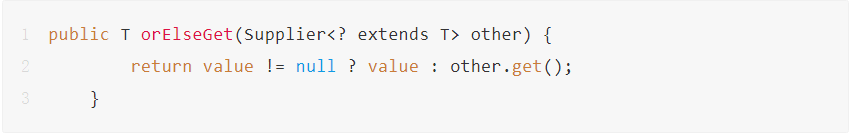


实例：

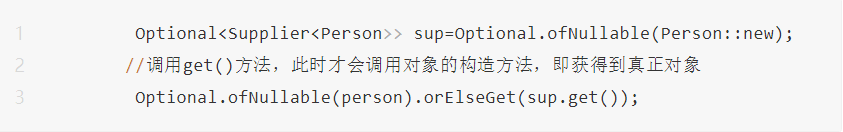


## 2.9 Optional.orElseGet()方法(为空返回Supplier对象)

这个与orElse很相似，入参不一样，入参为Supplier对象，为空返回传入对象的.get()方法，如果非空则返回当前对象  
源码:

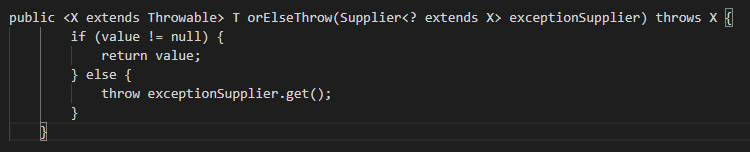


实例：



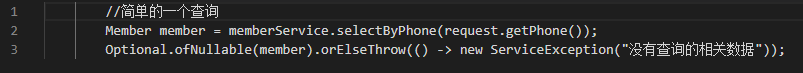
## 2.10 Optional.orElseThrow()方法(为空返回异常)

这个我个人在实战中也经常用到这个方法，方法作用的话就是如果为空，就抛出你定义的异常，如果不为空返回当前对象，在实战中所有异常肯定是要处理好的，为了代码的可读性  
源码：



实例：

这个就贴实战源码了



## 2.11 相似方法进行对比分析

可能小伙伴看到这，没用用过的话会觉得orElse()和orElseGet()还有orElseThrow()很相似，map()和flatMap()好相似

哈哈哈不用着急，都是从这一步过来的，我再给大家总结一下不同方法的异同点

**orElse()和orElseGet()和orElseThrow()的异同点**

方法效果类似，如果对象不为空，则返回对象，如果为空，则返回方法体中的对应参数，所以可以看出这三个方法体中参数是不一样的  
orElse（T 对象）  
orElseGet（Supplier < T >对象）  
orElseThrow（异常）

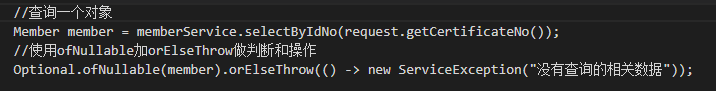
**map()和orElseGet的异同点**

方法效果类似，对方法参数进行二次包装，并返回,入参不同  
map（function函数）  
flatmap（Optional< function >函数）

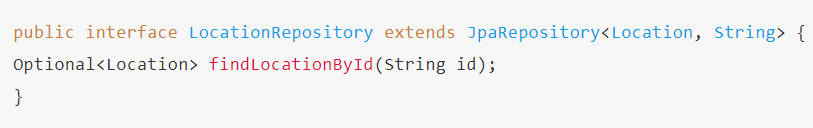
具体要怎么用，要根据业务场景以及代码规范来定义，下面可以简单看一下我在实战中怎用使用神奇的Optional。

# ****3.实战场景再现****

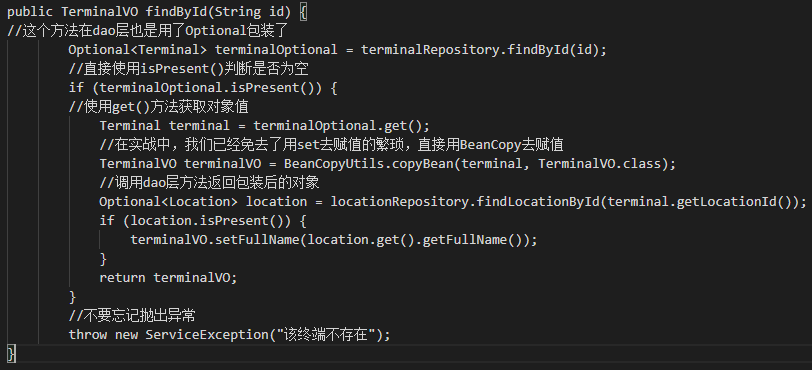
场景1：  
在service层中  
查询一个对象，返回之后判断是否为空并做处理



场景2：  
我们可以在dao接口层中定义返回值时就加上Optional  
例如：  
我使用的是jpa，其他也同理



service中：



实战场景还有很多，包括return时可以判断是否返回当前值还是跳转到另一个方法体中，什么的还有很多，如果大家没有经验的小伙伴还想进行学习，可以评论一下我会回复大家。

# ****4.Optional使用注意事项****

Optional真么好用，真的可以完全替代if判断吗？  
我想这肯定是大家使用完之后Optional之后可能会产生的想法，答案是否定的  
举一个最简单的栗子：

例子1：

如果我只想判断对象的某一个变量是否为空并且做出判断呢？



我觉得这个例子就能很好的说明这个问题，只是一个很简单判断，如果用了Optional我们还需要考虑包装值，考虑代码书写，考虑方法调用，虽然只有一行，但是可读性并不好，如果别的程序员去读，我觉得肯定没有if看的明显。

# ****5.jdk1.9对Optional优化****

首先增加了三个方法:  
or()、ifPresentOrElse() 和 stream()。

or() 与orElse等方法相似，如果对象不为空返回对象，如果为空则返回or()方法中预设的值。  
ifPresentOrElse() 方法有两个参数：一个 Consumer 和一个 Runnable。如果对象不为空，会执行 Consumer 的动作，否则运行 Runnable。相比ifPresent（）多了OrElse判断。  
stream()将Optional转换成stream，如果有值就返回包含值的stream，如果没值，就返回空的stream。

因为这个jdk1.9的Optional具体我没有测试，同时也发现有蛮好的文章已经也能让大家明白jdk1.9的option的优化,我就不深入去说了。