<https://mp.weixin.qq.com/s/796x_BU595R0vESbnl2KBA>

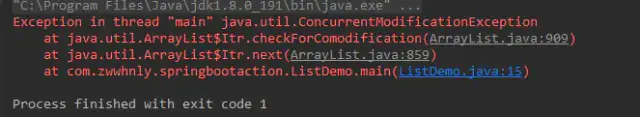
# List如何一边遍历，一边删除？

## 1. 新手常犯的错误

可能很多新手（包括当年的我，哈哈）第一时间想到的写法是下面这样的：

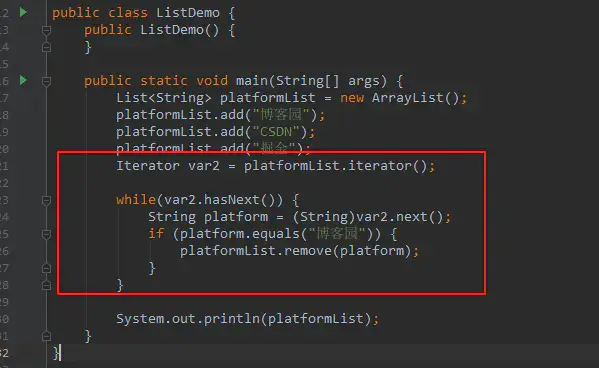


然后满怀信心的去运行，结果竟然抛java.util.ConcurrentModificationException异常了，翻译成中文就是：并发修改异常。



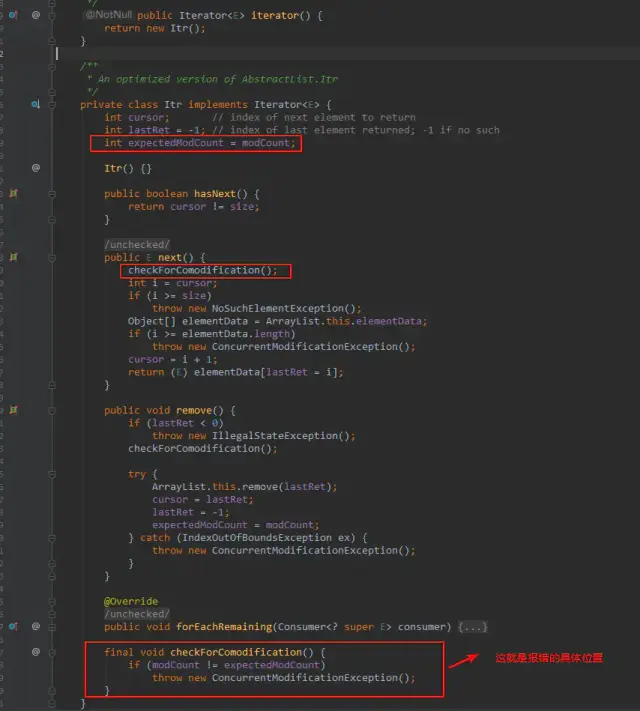
是不是很懵，心想这是为什么呢？

让我们首先看下上面这段代码生成的字节码，如下所示：



由此可以看出，foreach循环在实际执行时，其实使用的是Iterator，使用的核心方法是hasnext()和next()。

然后再来看下ArrayList类的Iterator是如何实现的呢？

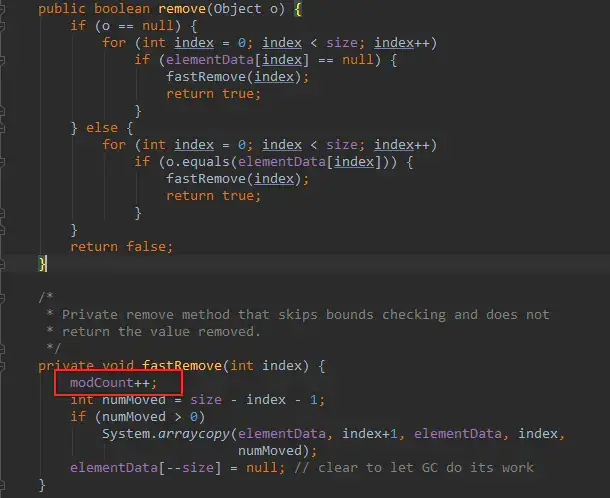


可以看出，调用next()方法获取下一个元素时，第一行代码就是调用了checkForComodification();，而该方法的核心逻辑就是比较modCount和expectedModCount这2个变量的值。

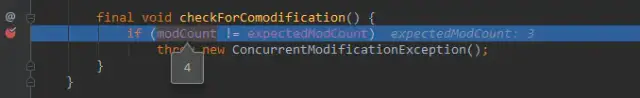
在上面的例子中，刚开始modCount和expectedModCount的值都为3，所以第1次获取元素"博客园"是没问题的，但是当执行完下面这行代码时：

platformList.remove(platform);

modCount的值就被修改成了4。



所以在第2次获取元素时，modCount和expectedModCount的值就不相等了，所以抛出了java.util.ConcurrentModificationException异常。



既然不能使用foreach来实现，那么我们该如何实现呢？

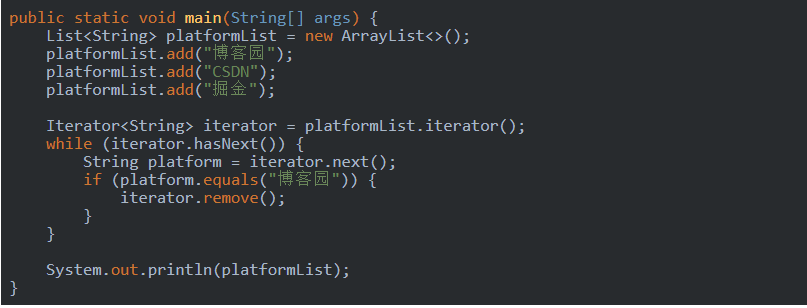
主要有以下3种方法：

* 使用Iterator的remove()方法
* 使用for循环正序遍历
* 使用for循环倒序遍历

接下来一一讲解。

## 2. 使用Iterator的remove()方法

使用Iterator的remove()方法的实现方式如下所示：

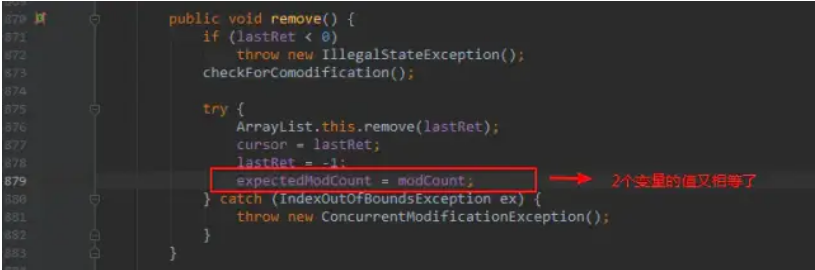


输出结果为：

[CSDN, 掘金]

为什么使用iterator.remove();就可以呢？

让我们看下它的源码：



可以看出，每次删除一个元素，都会将modCount的值重新赋值给expectedModCount，这样2个变量就相等了，不会触发java.util.ConcurrentModificationException异常。更多面试题，欢迎关注公众号 Java面试题精选

## 3. 使用for循环正序遍历

使用for循环正序遍历的实现方式如下所示：



这种实现方式比较好理解，就是通过数组的下标来删除，不过有个注意事项就是删除元素后，要修正下下标的值：

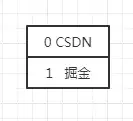
i = i - 1;

为什么要修正下标的值呢？

因为刚开始元素的下标是这样的：



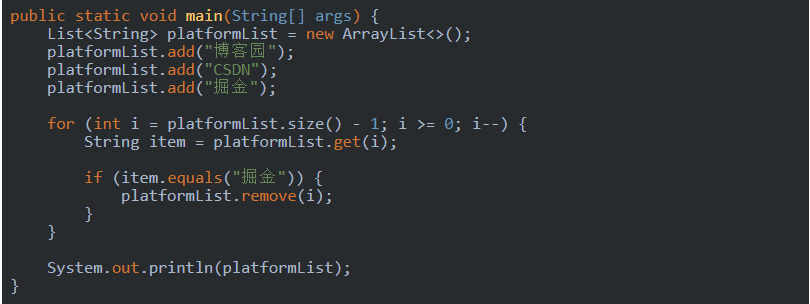
第1次循环将元素"博客园"删除后，元素的下标变成了下面这样：



第2次循环时i的值为1，也就是取到了元素”掘金“，这样就导致元素"CSDN"被跳过检查了，所以删除完元素后，我们要修正下下标，这也是上面代码中i = i - 1;的用途。

## 4. 使用for循环倒序遍历

使用for循环倒序遍历的实现方式如下所示：



这种实现方式和使用for循环正序遍历类似，不过不用再修正下标，因为刚开始元素的下标是这样的：



第1次循环将元素"掘金"删除后，元素的下标变成了下面这样：



第2次循环时i的值为1，也就是取到了元素”CSDN“，不会导致跳过元素，所以不需要修正下标。