#### JVM原理手记

##### 之再说空指针

问题描述

再说空指针

解决方案

**怎样处理空指针异常？**

1. 显式判断空指针异常

if (foo != null) {

return foo.value;

} else {

throw new NullPointerException();

}

1. 隐式捕捉空指针异常

try {

return foo.value;

} catch (segment\_fault) {

uncommon\_trap();

}

**两种方式的优劣？**

虚拟机会注册一个Segment Fault信号的异常处理器（伪代码中的uncommon\_trap()），这样当foo不为空的时候，对value的访问是不会额外消耗一次对foo判空的开销的。代价就是当foo真的为空时，必须转入到异常处理器中恢复并抛出NullPointerException异常，这个过程必须从用户态转到内核态中处理，处理结束后再回到用户态，速度远比一次判空检查慢。当foo极少为空的时候，隐式异常优化是值得的，但假如foo经常为空的话，这样的优化反而会让程序更慢

**一句话小结**？

空值出现频率较低时，使用try-catch块注册异常处理器；

反之，显式判断空指针。

□