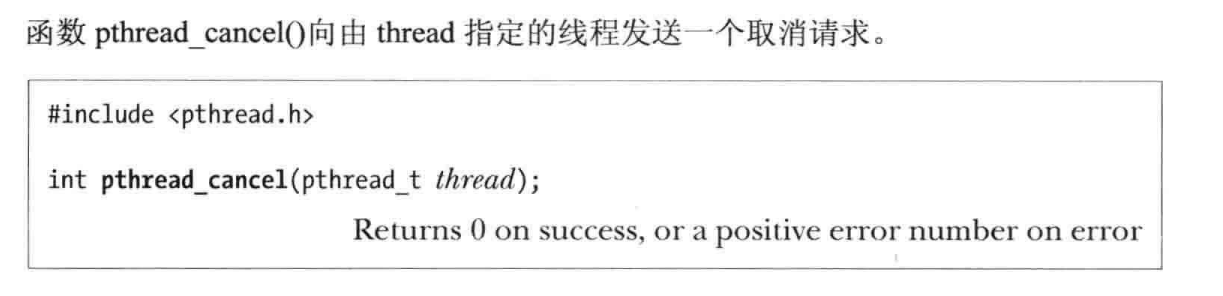
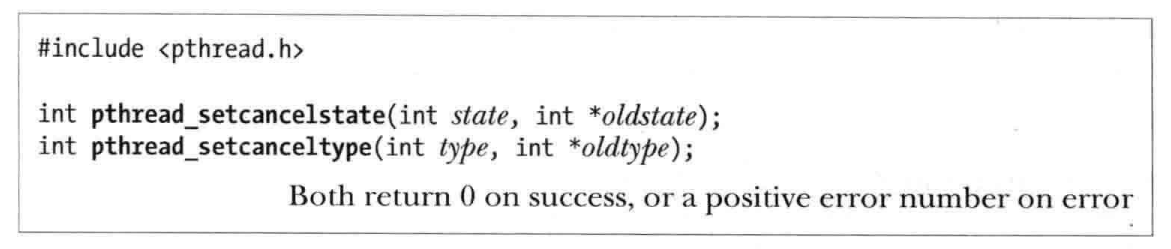
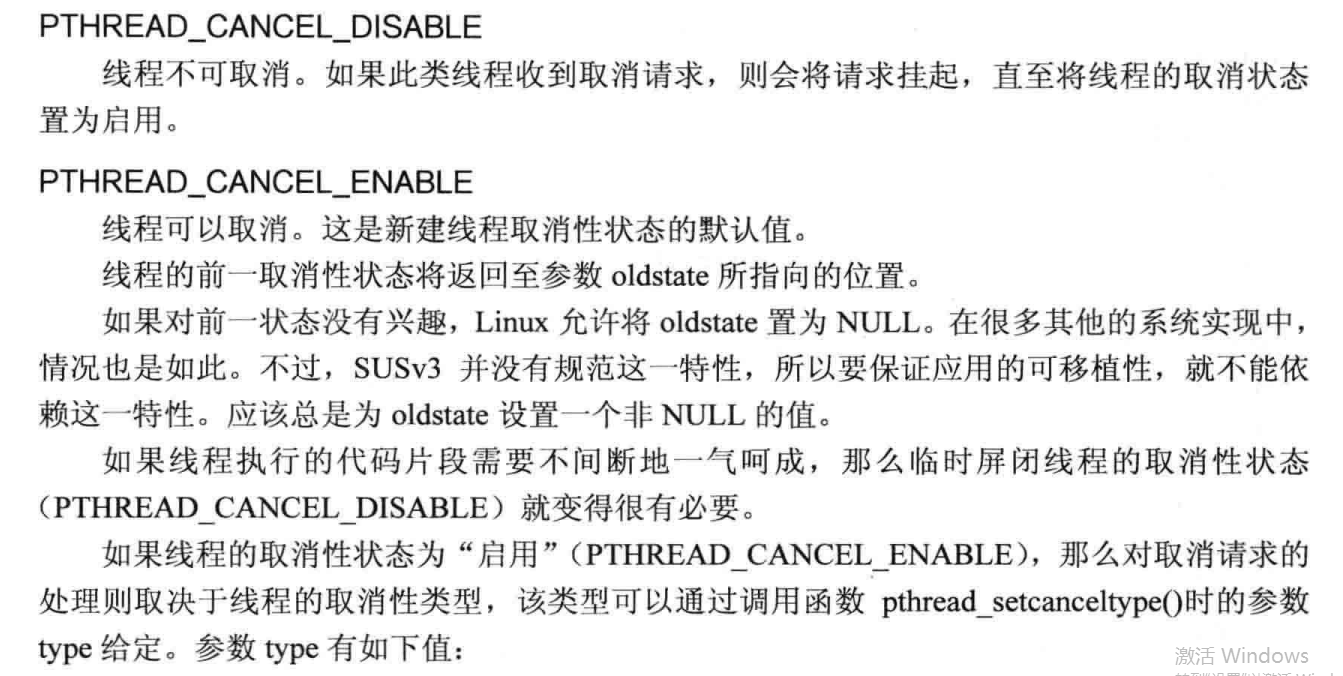
第三十二章

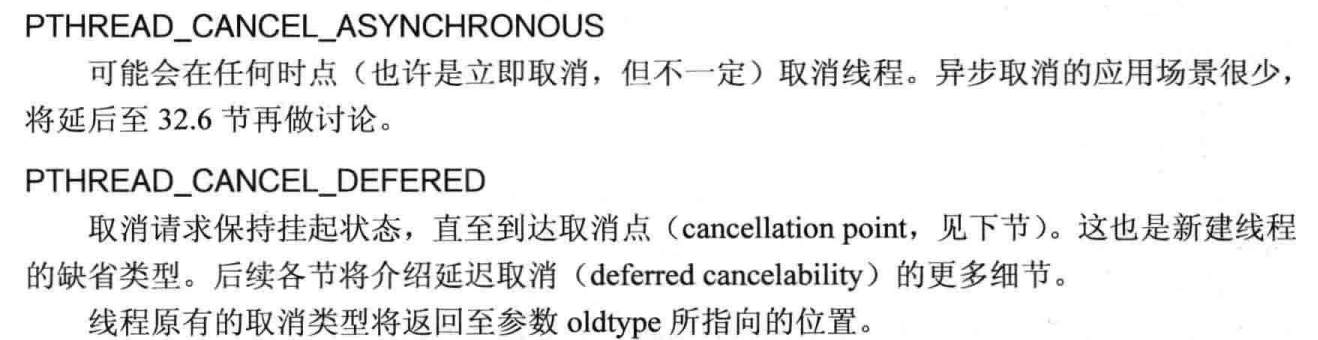
1. 线程取消：
2. 取消一个线程：



1. 取消状态和类型：





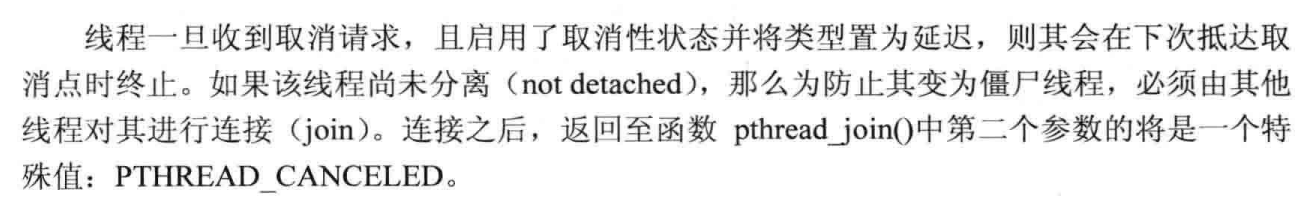


1. 取消点：

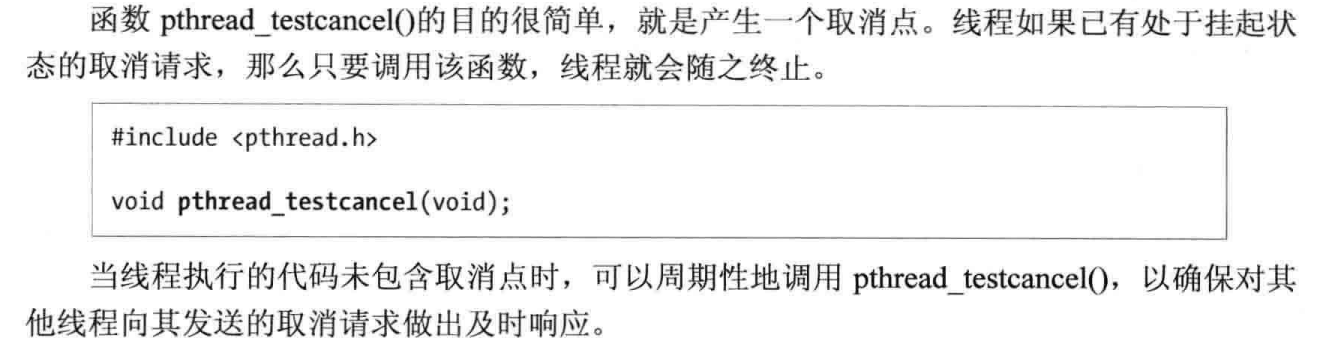
如果线程类型和状态分别置为启用和延迟，仅当线程到达某个取消点，取消才会起作用。所以取消点是由实现定义的某些函数的调用处。SUSv3规定了某些函数必须是取消点。

还有部分函数是可能取消点，如stdio函数等等。除了这两个外，别的函数不能实现为取消点。

通过线程取消退出的函数也有连接防止出现僵尸线程。

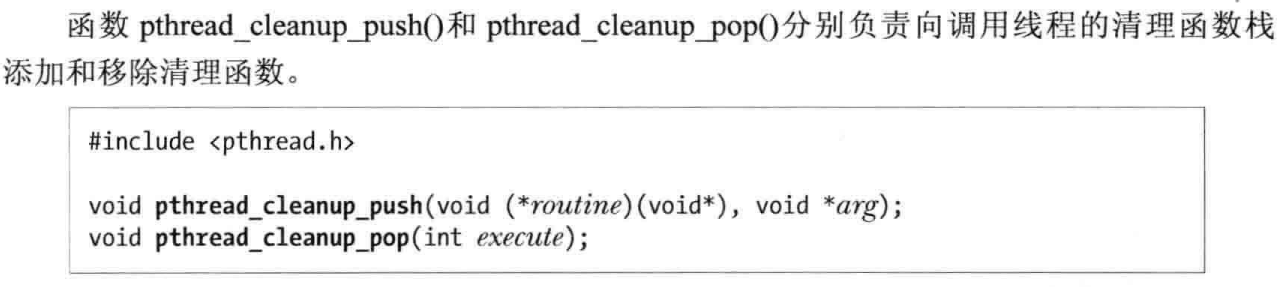


1. 线程的可取消性检测：



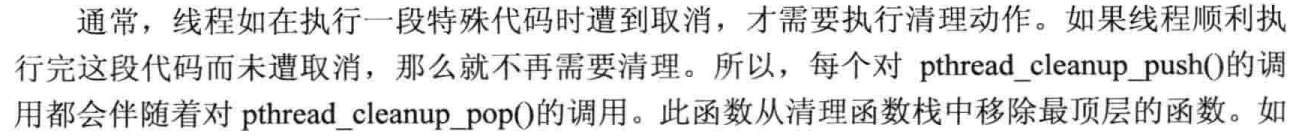
1. 清理函数：

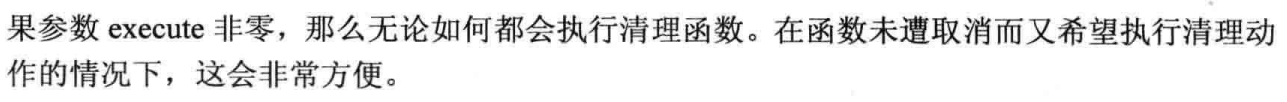
为了让线程取消可以做一些收尾工作，每个线程都拥有一个清理函数栈，当线程取消的时候，会一次从栈顶到栈底执行清理函数。



什么情况会执行清理函数？

1. 线程取消。
2. pthread\_exit
3. Pthread\_clean\_pop(nozero)





因为这两个函数要成对调用，所以某些系统实现为宏，即包含在两个{}之间的代码，这时候需要注意，在两个函数间声明的变量仅在这里面可见。

1. 异步取消：

取消类型设置为PTHREAD\_CANCEL\_ASYNCHRONOUS。这种情况很少使用，仅在取消执行计算密集型线程，因为清理函数无法得知线程的具体状态。仅仅三个函数实现为异步取消安全。

