不是所有的线程都有消息队列，只有GUI线程才有消息队列。

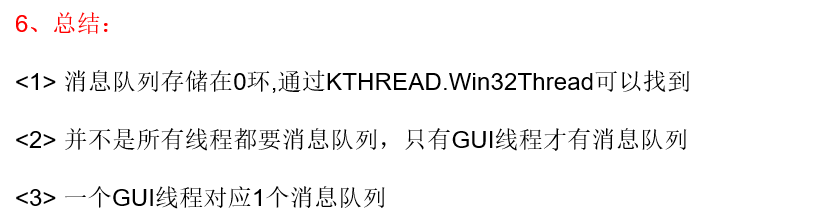


消息队列结构放在KTHREAD结构的Win32Thread项上。

一个普通的线程调用Win32K.sys中的代码时，就会把一个普通线程转换为GUI线程，加上

了消息队列。

一个线程刚创建的时候是一个普通线程，当调用任何一个Win32k.sys中的函数的时候，就会转换为GUI线程，此时就有了消息队列。



在加载启动初始化Win32K.sys模块的时候，会启动两个线程，一个是键盘监控线程，一个是鼠标监控线程，这两个线程是0环的线程，这两个线程一个用来收集鼠标消息，一个用来收集键盘消息，收到消息之后，会将消息存到对应线程的消息队列中。

真正的窗口是0环画的，windows是把画窗口，消息处理这些实现放在0环中，交由Win32K.sys模块来处理。系统调用号大于1000-时调用Win32K.sys模块，小于1000调用ntokrl.exe模块。

User32.dll中只要与窗口相关的函数都进0环，GDI.dll所有函数都要进0环。都是进入到Win32K.dll这个模块。

任何一个窗口，在0环都拥有一个结构体：WINDOW\_OBJECT结构。其中重要的成员PTHREADINFO pti 当前窗口指向所属线程。一个线程中存在很多窗口，注意窗口的定义，一个主窗口中存在好多窗口，例如一个按钮也是一个窗口。一个线程可以包含多个床，但是一个窗口只能属于一个线程。并且窗口都是在0环画的，每一个窗口不管窗口多大，在0环都有一个独一无二的0环结构体，该结构体中又指向了所属线程。

该WINDOW\_OBJECT结构中还有重要成员：

Wnd，这个成员指出了窗口的风格，父窗口，子窗口等等。

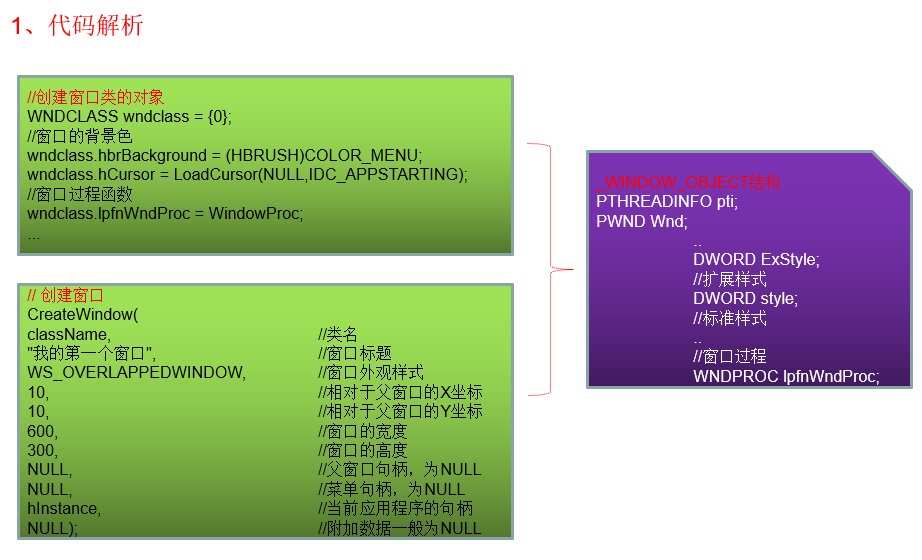
IpfnWndproc，对应的窗口过程。

鼠标监控线程能够检测到你鼠标点在那个窗口，根据0环的窗口结构体找到所对应的线程，把点击生成的消息放到**对应线程**的消息队列中，也就是内核对象KTHREAD中的Win32Thread中。



这个CreateWindow函数返回一个句柄，这个句柄其实就是内核0环的窗口结构体在句柄表的索引。窗口的所有重要数据根本不在3环，在0环的WINDOW\_OBJECT结构中。那么3环如果想用的话，不能把0环的WINDOW\_OBJECT结构体的地址指针返回，这样就会出错，因此返回一个句柄。加深对句柄的理解，每个句柄在3环都对应一个结构体。

在0环创建一张大表，无论哪个进程创建的窗口都在这张表中，这张表不是进程私有的，是所有进程共有的。一般的句柄表是进程私有的，也就是一个进程在内核存在一张只属于他自己的一张句柄表。因此可以不同的进程来获取其他进程窗口的句柄，由于句柄表在0环只有一个，是共享的。



创建窗口类，创建窗口的目的都是为了为0环的窗口结构体WINDOW\_OBJECT赋值。



一共存在7个消息队列。

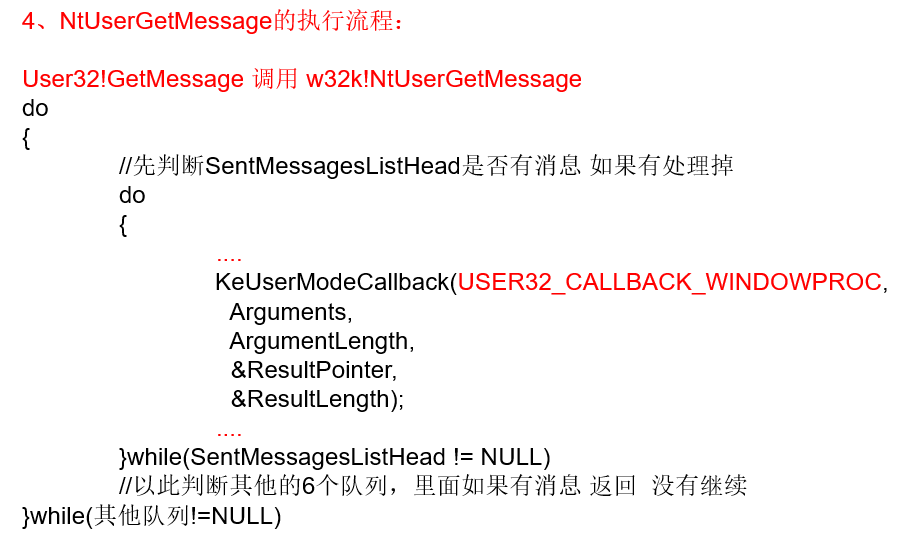
KTHREAD结构体中的成员Win32Thread指向ThreadInfo结构体，ThreadInfo结构体中的\_USER\_MESSAGE\_QUEUE结构体中存在这几个消息队列。

GetMessage函数会先判断

<1> SentMessagesListHead //接到SendMessage发来的消息

这个消息队列中是否存在消息，如果存在就直接处理执行掉，调用相应的消息处理函数也就是回调函数。

如果没有上述队列中的消息，就从指定的窗口的消息队列中取出消息。而取消息的时候，是从0环的内核WINDOW\_OBJECT所属的内核线程对象中的Win32Thread取出消息。



GetMessage函数的底层在0环的实现中，会先遍历SentMessageListHead消息队列中是否存在消息（消息是由SendMessage函数来发送的），如果存在就会调用3环的回调函数来处理，而回调函数是我们自己写在3环的函数。

**一个线程的所有窗口共享一个消息队列**，GetMessage的参数hWnd：取得该消息的窗口句柄，值为NULL时，GetMessage会检索任何属于调用线程的窗口消息。因此GetMessage会将属于这个线程的所有消息从消息队列中取出来，取出来的消息中会属于不同的窗口。

DispatchMessage函数消息分发，根据窗口句柄调用相关的窗口过程。该函数的具体实现就是，进入到0环，根据消息MSG结构体中的句柄，来找到对应0环的窗口结构体对象，再通过窗口对象找到相对应的窗口过程，之后再调用。（首先说一下，WINDOW\_OBJECT结构中存在成员IpfnWndproc，对应的窗口过程。）

因此GetMessage函数和DispatchMessage函数的底层均会出现函数KeUserModeCallback由0环返回到3环，来执行3环的回调函数。只不过处理的消息的类型不同。

TranslateMessage函数是用来加工消息的，加工键盘消息，接收到具体的键盘字符。仅此而已。

CreateWindow在创建的过程，通过0环的API，调用KeUserModeCallback函数，之后通过KeUserModeCallback函数调用窗口过程。因此如果你希望在窗口创建的过程中，自己想要做一些事情，就在窗口过程中的vm\_create消息中填写自己的代码。

从内核调用用户态的代码函数：

https://bbs.pediy.com/thread-104918.htm%E3%80%82

