1、WH\_CALLWNDPROC和WH\_CALLWNDPROCRET Hooks

WH\_CALLWNDPROC和WH\_CALLWNDPROCRET Hooks使你可以监视发送到窗口过程的消息。系统在消息发送到接收窗口过程之前调用WH\_CALLWNDPROC Hook子程，并且在窗口过程处理完消息之后调用WH\_CALLWNDPROCRET Hook子程。

WH\_CALLWNDPROCRET Hook传递指针到CWPRETSTRUCT结构，再传递到Hook子程。

CWPRETSTRUCT结构包含了来自处理消息的窗口过程的返回值，同样也包括了与这个消息关联的消息参数。

2、WH\_CBT Hook

在以下事件之前，系统都会调用WH\_CBT Hook子程，这些事件包括：

1）激活，建立，销毁，最小化，最大化，移动，改变尺寸等窗口事件；

2）完成系统指令；

3）来自系统消息队列中的移动鼠标，键盘事件；

4）设置输入焦点事件；

5）同步系统消息队列事件。

Hook子程的返回值确定系统是否允许或者防止这些操作中的一个。

3、WH\_DEBUG Hook

在系统调用系统中与其他Hook关联的Hook子程之前，系统会调用WH\_DEBUG Hook子程。你可以使用这个Hook来决定是否允许系统调用与其他Hook关联的Hook子程。

4、WH\_FOREGROUNDIDLE Hook

当应用程序的前台线程处于空闲状态时，可以使用WH\_FOREGROUNDIDLE Hook执行低优先级的任务。当应用程序的前台线程大概要变成空闲状态时，系统就会调用WH\_FOREGROUNDIDLE Hook子程。

5、WH\_GETMESSAGE Hook

应用程序使用WH\_GETMESSAGE Hook来监视从GetMessage or PeekMessage函数返回的消息。你可以使用WH\_GETMESSAGE Hook去监视鼠标和键盘输入，以及其他发送到消息队列中的消息。

6、WH\_JOURNALPLAYBACK Hook

WH\_JOURNALPLAYBACK Hook使应用程序可以插入消息到系统消息队列。可以使用这个Hook回放通过使用WH\_JOURNALRECORD Hook记录下来的连续的鼠标和键盘事件。只要WH\_JOURNALPLAYBACK Hook已经安装，正常的鼠标和键盘事件就是无效的。

WH\_JOURNALPLAYBACK Hook是全局Hook，它不能象线程特定Hook一样使用。

WH\_JOURNALPLAYBACK Hook返回超时值，这个值告诉系统在处理来自回放Hook当前消息之前需要等待多长时间（毫秒）。这就使Hook可以控制实时事件的回放。

WH\_JOURNALPLAYBACK是system-wide local hooks，它们不会被注射到任何行程位址空间。

7、WH\_JOURNALRECORD Hook

WH\_JOURNALRECORD Hook用来监视和记录输入事件。典型的，可以使用这个Hook记录连续的鼠标和键盘事件，然后通过使用WH\_JOURNALPLAYBACK Hook来回放。

WH\_JOURNALRECORD Hook是全局Hook，它不能象线程特定Hook一样使用。

WH\_JOURNALRECORD是system-wide local hooks，它们不会被注射到任何行程位址空间。

8、WH\_KEYBOARD Hook

在应用程序中，WH\_KEYBOARD Hook用来监视WM\_KEYDOWN and WM\_KEYUP消息，这些消息通过GetMessage or PeekMessage function返回。可以使用这个Hook来监视输入到消息队列中的键盘消息。

9、WH\_KEYBOARD\_LL Hook

WH\_KEYBOARD\_LL Hook监视输入到线程消息队列中的键盘消息。

1. WH\_MOUSE Hook，

只能监控钩子所在模块的鼠标事件。

WH\_MOUSE Hook监视从GetMessage 或者 PeekMessage 函数返回的鼠标消息。使用这个Hook监视输入到消息队列中的鼠标消息。

1. WH\_MOUSE\_LL Hook

可以截获整个系统所有模块的鼠标事件

WH\_MOUSE\_LL Hook监视输入到线程消息队列中的鼠标消息。

12、WH\_MSGFILTER 和 WH\_SYSMSGFILTER Hooks

WH\_MSGFILTER 和 WH\_SYSMSGFILTER Hooks使我们可以监视菜单，[滚动条](http://baike.so.com/doc/6940271.html" \t "http://baike.so.com/doc/_blank)，消息框，[对话框](http://baike.so.com/doc/6158761.html" \t "http://baike.so.com/doc/_blank)消息并且发现用户使用ALT+TAB or ALT+ESC 组合键切换窗口。WH\_MSGFILTER Hook只能监视传递到菜单，滚动条，消息框的消息，以及传递到通过安装了Hook子程的应用程序建立的对话框的消息。WH\_SYSMSGFILTER Hook监视所有应用程序消息。

WH\_MSGFILTER 和 WH\_SYSMSGFILTER Hooks使我们可以在模式循环期间过滤消息，这等价于在主消息循环中过滤消息。

通过调用CallMsgFilter function可以直接的调用WH\_MSGFILTER Hook。通过使用这个函数，应用程序能够在模式循环期间使用相同的代码去过滤消息，如同在主消息循环里一样。

13、WH\_SHELL Hook

外壳应用程序可以使用WH\_SHELL Hook去接收重要的通知。当外壳应用程序是激活的并且当顶层窗口建立或者销毁时，系统调用WH\_SHELL Hook子程。

WH\_SHELL 共有5钟情况：

1）只要有个top-level、unowned 窗口被产生、起作用、或是被摧毁；

2）当Taskbar需要重画某个按钮；

3）当系统需要显示关于Taskbar的一个程序的最小化形式；

4）当如今的键盘布局状态改变；

5）当使用者按Ctrl+Esc去执行Task Manager（或相同级别的程序）。

按照惯例，外壳应用程序都不接收WH\_SHELL消息。所以，在应用程序能够接收WH\_SHELL消息之前，应用程序必须调用SystemParametersInfo function注册它自己。

也就是说，如果你的App没有调用如下代码注册自己之前，那么它永远也不会响应你的ShellProc：

MINIMIZEDMETRICS mm = {sizeof(MINIMIZEDMETRICS), 0,0, ARW\_HIDE};

SystemParametersInfo(SPI\_SETMINIMIZEDMETRICS, sizeof(MINIMIZEDMETRICS) , &mm, 0);