TranslateMessage(&msg);

TranslateMessage是用来把快捷键消息转换为字符消息,并将转换后的新消息投递到调用线程的消息队列中。

由于Windows对所有键盘编码都是采用虚拟键的定义，这样当按键按下时，并不得字符消息，需要键盘映射转换为字符的消息。字符消息被投递到调用线程的消息队列中，当下一次调用GetMessage函数时被取出。

LONG DispatchMessage（CONST MSG\*lpmsg）；

函数功能：该函数分发一个消息给窗口程序，即把消息推送到afxwndproc,最后流向pwnd->windowproc,而DispatchMessage是在cwinapp::run 调用pumpmessage,而pumpmessage又调用DispatchMessage。通常消息从GetMessage函数获得或者TranslateMessage函数传递的。消息被分发到回调函数（过程函数)，作用是消息传递给操作系统，然后操作系统去调用我们的回调函数，也就是说我们在窗体的过程函数中处理消息。

PeekMessage和GetMessage函数的主要区别有：

GetMessage的主要功能是从消息队列中“取出”消息，消息被取出以后，就从消息队列中将其删除；而PeekMessage的主要功能是“窥视”消息，如果有消息，就返回true，否则返回false。也可以使用PeekMessage从消息队列中取出消息，这要用到它的一个参数（UINT wRemoveMsg），如果设置为PM\_REMOVE，消息则被取出并从消息队列中删除；如果设置为PM\_NOREMOVE，消息就不会从消息队列中取出。

如果GetMessage从消息队列中取不到消息，则线程就会被操作系统挂起，等到OS重新调度该线程时，两者的性质不同：使用GetMessage线程仍会被挂起，使用PeekMessage线程会得到CPU的控制权，运行一段时间。

GetMessage每次都会等待消息，直到取到消息才返回；而PeekMessage只是查询消息队列，没有消息就立即返回，从返回值判断是否取到了消息

PeekMessage是一个具有线程异步行为的函数，不管消息队列中是否有消息，函数都会立即返回。而GetMessage则是一个具有线程同步行为的函数，如果消息队列中没有消息的话，函数就会一直等待，直到消息队列中至少有一条消息时才返回。

另外：PumpMessage起到消息泵的作用，其原理就是使用GetMessage或PeekMessage函数从消息队列中逐个读取消息，然后进行必要处理之后再发送出去。我们可以使用PumpMessage默认处理过程，当然也可以自己修改消息处理的方式。通常我们在我们自己设计窗口时也会自己设计消息循环方式。即PumpMessage函数会调用GetMessage、TranslateMessage、DispatchMessage。

HOOK（钩子，挂钩）是一种实现Windows平台下类似于中断的机制。HOOK机制允许应用程序拦截并处理Windows消息或指定事件，当指定的消息发出后，HOOK程序就可以在消息到达目标窗口之前将其捕获，从而得到对消息的控制权，进而可以对该消息进行处理或修改，加入我们所需的功能。

定义钩子函数

钩子函数是一种特殊的回调函数。钩子监视的特定事件发生后，系统会调用钩子函数进行处理。不同事件的钩子函数的形式是各不相同的。下面以鼠标钩子函数举例说明钩子函数的原型：

LRESULT CALLBACK HookProc(int nCode ,WPARAM wParam,LPARAM lParam)

参数wParam和 lParam包含所钩消息的信息，比如鼠标位置、状态，键盘按键等。

安装钩子

在程序初始化的时候，调用函数SetWindowsHookEx安装钩子。其函数原型为：

HHOOK SetWindowsHookEx( int idHook,HOOKPROC lpfn, INSTANCE hMod,DWORD dwThreadId )