函数简述:

　　 中止Windows Sockets DLL的使用.

　　        #include <winsock.h>

　　        int PASCAL FAR WSACleanup ( void );

注释:

       应用程序或DLL在使用Windows Sockets服务之前必须要进行一次成功的WSAStartup()调用.当它完成了Windows Sockets的使用后,应用程序或DLL必须调用WSACleanup()将其从Windows Sockets的实现中注销,并且该实现释放为应用程序或DLL分配的任何资源.任何打开的并已建立连接的SOCK\_STREAM类型套接口在调用WSACleanup()时会重置; 而已经由closesocket()关闭却仍有要发送的悬而未决数据的套接口则不会受影响－ 该数据仍要发送.

        对应于一个任务进行的每一次WSAStartup()调用,必须有一个WSACleanup()调用.只有最后的WSACleanup()做实际的清除工作;前面的调用仅仅将Windows Sockets DLL中的内置引用计数递减.一个简单的应用程序为确保WSACleanup()调用了足够的次数,可以在一个循环中不断调用WSACleanup()直至返回WSANOTINITIALISED.

返回值:   0 操作成功.     SOCKET\_ERROR 否则.同时可以调用WSAGetLastError()获得错误代码.

评价:

　　 一个常见的Windows Sockets编程错误是:试图在一个阻塞钩子函数中调用WSACleanup()并且检测返回值失败.如果在一次阻塞调用正在进行时应用程序需要退出,应用程序必须首先通过调用WSACancelBlockingCall()使该阻塞操作作废, 然后一旦控制返回给应用程序时就启动WSACleanup().

关于Windows Sockets提供者的说明：

　 良好的Windows Sockets应用程序会通过调用WSACleanup()指出它从Windows Sockets实现中注销.本函数因此可以用来释放分配给指定应用程序的资源.

　Windows Sockets的实现必须能处理应用程序在调用WSACleanup()函数之前就中止的情况.－例如,返回一个错误.

　在一个多线程的环境下,WSACleanup()中止了Windows Sockets在所有线程上的操作.

　Windows Sockets的实现必须确认WSACleanup()调用后,应用程序能调用WSAStartup()函数来重新建立Windows Sockets的应用.

慎用原因：

由于WSACleanup()的调用会导致将socket的初始化资源从Windows Sockets的实现中注销,并且该实现释放为应用程序或DLL分配的任何资源，因此，在编写ie空间的时候没有调用WSAStartUP()，而又调用了WSACleanup()，或导致页面不正常请求中断，致使ie浏览出现不可预见的意外......