June 14th, 2013

1. SDK: Software development kit, it is a set of software development tools that allows for the creation of applications for a certain software package, software framework, hardware platform, computer system, video game console, operating system, or similar development platform.
2. Create Thread function:

Creates a thread to execute within the virtual address space of another process.

HANDLE WINAPI CreateThread(

\_In\_opt\_   LPSECURITY\_ATTRIBUTES lpThreadAttributes,

\_In\_       SIZE\_T dwStackSize,

\_In\_       LPTHREAD\_START\_ROUTINE lpStartAddress,

\_In\_opt\_   LPVOID lpParameter,

\_In\_       DWORD dwCreationFlags,

\_Out\_opt\_  LPDWORD lpThreadId

);

lpThreadAttributes:

A pointer to a security\_attributes structure that determines whether the returned handle can be inherited by child processes. If it is NULL, the handle cannot be inherited.

dwStackSize:

The initial size of the stack, in bytes. If this parameter is zero, the new thread uses the default size for the executable.

lpStartAddress:

A pointer to the application-defined function to be executed by the thread. This pointer represents the starting address of the thread.

lpParameter:

A pointer to a variable to be passed to the thread.

dwCreationFlags:

The flags that control the creation of the thread. Value 0 means the thread runs immediately after creation.

lpThreadId: A pointer to a variable that receives the thread identifier. If this parameter is NULL, the thread identifier is not returned.

1. SetEvent function:

Sets the specified event object to the signaled state.

BOOL WINAPI SetEvent(

\_In\_  HANDLE hEvent

);

hEvent:

A handle to the event object, the handle must have the EVENT\_MODIFY\_STATE access right.

1. DeviceIoControl function:

BOOL WINAPI DeviceIoControl(

\_In\_         HANDLE hDevice,

\_In\_         DWORD dwIoControlCode,

\_In\_opt\_     LPVOID lpInBuffer,

\_In\_         DWORD nInBufferSize,

\_Out\_opt\_    LPVOID lpOutBuffer,

\_In\_         DWORD nOutBufferSize,

\_Out\_opt\_    LPDWORD lpBytesReturned,

\_Inout\_opt\_  LPOVERLAPPED lpOverlapped

);

hDevice:

A handle to the device on which the operation is to be performed. The device is typically a volume, directory, file, or stream.

deloControlCode:

The control code for the operation. This value identifies the specific operation to be performed and the type of device on which to perform it.

lpInBuffer:

A pointer to the input buffer that contains the data required to perform the operation. The format of this data depends on the value of the dwIoControlCode parameter.

nInBufferSize:

The size of the input buffer in bytes.

lpOutBuffer:

A pointer to the output buffer that is to receive the data returned by the operation. The format of this data depends on the value of the dwIoControlCode parameter.

nOutBufferSize:

The size of the output buffer in bytes.

lpBytesReturned:

A pointer to a variable that receives the size of the data stored in the output buffer in bytes.

lpOverlapped:

A pointer to an overlapped structure.

If the operation completes successfully, the return value is non-zero. If the operation fails or is pending, the return value is zero.

1. WaitForSingleObject function:

DWORD WINAPI WaitForSingleObject(

\_In\_  HANDLE hHandle,

\_In\_  DWORD dwMilliseconds

);

hHandle:

A handle to the object. The handle must have the synchronize access right.

dwMillisecond:

The time-out interval, in milliseconds. If a nonzero value is specified, the function waits until the object is signaled or the interval elapses. If dwMilliseconds is zero, the function does not enter a wait state if the object is not signaled; it always returns immediately. If dwMilliseconds is INFINITE, the function will return only when the object is signaled.

If the function succeeds, the return value indicates the event that caused the function to return.

1. ReleaseMutex function:

BOOL WINAPI ReleaseMutex(

\_In\_  HANDLE hMutex

);

hMutex:

A handle to the mutex object.

If the function succeeds, the return value is nonzero. If the function fails, the return value is zero.

1. WaitForSingleObject and ReleaseMutes 是互相照应的一组函数，只要参数一样，那么waitforsingleobject在这个mutex空闲的时候会抢走这个mutex，然后直到releasemutex函数将这个mutex release掉以供其他人用。