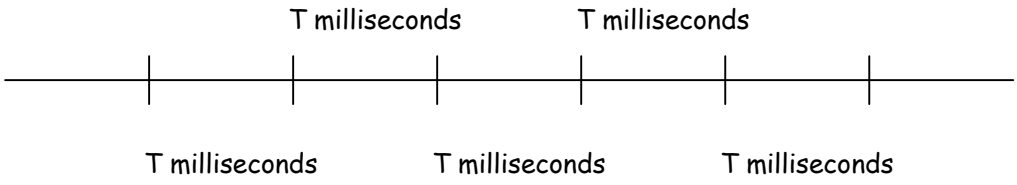
此文章使用的範例《每 T milliseconds 內顯示一次系統時間 [時：分：秒：毫秒]》。



假設顯示一次系統時間需要 S milliseconds，若 S 小於 T，則系統需延遲 T-S milliseconds 後，才能繼續顯示下一次系統時間。先將 Source code 列出來：

Note：記得加入 Winmm.lib

```
#include <stdio.h>
#include <conio.h>    // for kbhit()
#include <windows.h>

#define TARGET_RESOLUTION  1   // 1 millisecond target resolution
#define RATE    10                    //  每秒顯示幾次系統時間

HANDLE hEvent;    // Global variable
void CALLBACK OneShotTimer(UINT wTimerID, UINT msg, DWORD dwUser,
                           DWORD dw1, DWORD dw2);
int main()
{
    TIMECAPS tc;
    UINT wTimerRes,wTimerID;
    int RateInt,diffTime;
    DWORD targetTime;
    SYSTEMTIME systime;

    // Obtaining and Setting Timer Resolution
```

```
if ( timeGetDevCaps( &tc,sizeof(TIMECAPS) ) != TIMERR_NOERROR) {
    printf("Cannot Get Timer!\n");
    return -1;
}
wTimerRes = min(max(tc.wPeriodMin,TARGET_RESOLUTION),tc.wPeriodMax);

// Create Event Object
hEvent = CreateEvent( NULL , FALSE , FALSE , NULL );

while(!kbhit()) {
    RateInt = 1000 / RATE;
    targetTime = timeGetTime() + RateInt;

    // Get Local Time
    GetLocalTime(&systime);
    printf("目前時間：%02d:%02d:%02d:%03d\r", systime.wHour,
                                              systime.wMinute,
                                              systime.wSecond,
                                              systime.wMilliseconds);

    diffTime = targetTime - timeGetTime();
    if ( diffTime > 0 ) {
        // Set timer's resolution
        timeBeginPeriod(wTimerRes);
        // Start timer event
        wTimerID = timeSetEvent(diffTime,wTimerRes,OneShotTimer,0,TIME_ONESHOT);
```

```
        if ( !wTimerID ){
            printf("Cannot Set Timer Event!\n");
            return -1;
        }

        WaitForSingleObject(hEvent,RateInt+wTimerRes);
        // Cancel timer event
        timeKillEvent(wTimerID);
        // Clear timer's resolution
        timeEndPeriod(wTimerRes);
      }
   }

   return 0;
}
void CALLBACK OneShotTimer(UINT wTimerID, UINT msg,
    DWORD dwUser, DWORD dw1, DWORD dw2)
{
    wTimerID = 0;
    SetEvent(hEvent);
}
```

Note:以下整理至 MSDN

□ Multimedia Timers

<1>Introduction

- Multimedia timer services allow applications to schedule timer with the greatest

resolution(or accuracy) possible for the hardware platform.

- These multimedia timer services allow you to schedule timer events at a higher resolution than other timer services.
- These timer services are useful for applications that demand high-resolution timing.

<2>Timer Resolution

- Use the **timeGetDevCaps** function to determine the minimum and maximum timer resolutions supported by the timer services.
- This function fills the **wPeriodMin** and **wPeriodMax** members of the **TIMECAPS** structure with the minimum and maximum resolutions.
- After you determine the minimum and maximum available timer resolutions, you must establish the minimum resolution you want your application to use.
- Use the **timeBeginPeriod** to set this resolution.
- Use the **timeEndPeriod** to clear this resolution.
- You must match each call to **timeBeginPeriod** with a call to **timeEndPeriod**, specifying the same minimum resolution in both calls.

<3> Timer Event Operations

- After you have established your application's timer resolution, you can start timer events by using the **timeSetEvent** function.
- One of the **timeSetEvent** function's parameters is the address of a **TimeProc** callback function that is called when the timer event takes space.
- Two types of timer events:
  - Single : A single timer event occurs once, after a specified number of milliseconds.
  - Periodic : A periodic timer event occurs every time a specified number of milliseconds elapses.
- The interval between periodic events is called *event delay*.

- The relationship between the resolution of a timer event and the length of the event delay is important in timer events.
  - For example, if you specify a resolution of 5 and an event delay of 100, the timer services notify the callback function after an interval ranging from 95 to 105 milliseconds.
- Cancel an active timer event at any time by using the **timeKillEvent** function.
- Be sure to cancel any outstanding timers before freeing the memory containing the callback function.
- The multimedia timer runs in its own thread.

<4> Obtaining and Setting Timer Resolution

```
#define TARGET_RESOLUTION 1   // 1-milliseconds resolution
TIMECAPS tc;
UINT wTimerRes;

if ( timeGetDevCaps( &tc,sizeof(TIMECAPS) ) != TIMERR_NOERROR) {
  // Error; application can't continue.
}
wTimerRes = min(max(tc.wPeriodMin,TARGET_RESOLUTION),tc.wPeriodMax);
timeBeginPeriod(wTimerRes);
```

Note：請記得在適當的位置加上 timeEndPeriod(wTimerRes);

<5>Starting a Single Timer Event

```
UINT SetTimerCallback(NPSEQ npSeq,        // sequencer data
                      UINT msInterval)    // event interval
{
    npSeq->wTimerID = timeSetEvent(
                msInterval,               // delay
                wTimerRes,                // resolution (global variable)
```

```
                    OneShotCallback,        // callback function
                    (DWORD)npSeq,           // user data
                    TIME_ONESHOT );         // single timer event
    if(! npSeq->wTimerID)
        return ERR_TIMER;
    else
        return ERR_NOERROR;
}
```

- To start a single timer event, an application must call the **timerSetEvent** function, specifying the amount of timer before the callback occurs, the resolution, the address of the callback function, and the user data to supply with the callback function.

<6>Writing a Timer Callback Function

```
void CALLBACK OneShotTimer(UINT wTimerID, UINT msg,
    DWORD dwUser, DWORD dw1, DWORD dw2)
{
    NPSEQ npSeq;                // pointer to sequencer data
    npSeq = (NPSEQ)dwUser;
    npSeq->wTimerID = 0;        // invalidate timer ID (no longer in use)
    TimerRoutine(npSeq);        // handle tasks
}
```

- OneShotTimer, invalidates the identifier for the single timer event and calls a timer routine to handle the application-specific tasks.

<7>Canceling a Timer Event

```
void DestroyTimer(NPSEQ npSeq)
{
    if(npSeq->wTimerID) {               // is timer event pending?
```

```
            timeKillEvent(npSeq->wTimerID);   // cancel the event
            npSeq->wTimerID = 0;
        }
    }
```

□ Idea

在此不對 Event Object 做介紹與整理，直接從程式中去瞭解 Event Object 的用途，直接從下面的程式片段：

```
① hEvent = CreateEvent( NULL , FALSE , FALSE , NULL );
  while(!kbhit()) {
    RateInt = 1000 / RATE;
    targetTime = timeGetTime() + RateInt;

    // Get Local Time & Display Time

② diffTime = targetTime - timeGetTime();
    if ( diffTime > 0 ) {
        timeBeginPeriod(wTimerRes);

③ wTimerID = timeSetEvent(diffTime,wTimerRes,OneShotTimer,0,TIME_ONESHOT);
④ if ( !wTimerID ) {
            printf("Cannot Set Timer Event!\n");
            return -1;
        }
        WaitForSingleObject(hEvent,RateInt+wTimerRes);
        timeKillEvent(wTimerID);
        timeEndPeriod(wTimerRes);
```

```
    }
  }

void CALLBACK OneShotTimer(UINT wTimerID, UINT msg,
    DWORD dwUser, DWORD dw1, DWORD dw2)
{
⑤ wTimerID = 0;
    SetEvent(hEvent);
}
```

① hEvent = CreateEvent(NULL , FALSE , FALSE , NULL);

-第二個參數值 FALSE 表示產生一個 auto-reset event object, and system automatically resets the state to nonsignaled after a single waiting thread has been released.

-第三個參數值 FALSE 表示 Event Object 初始狀態爲 nonsignaled

② diffTime = targetTime - timeGetTime();

- diffTime 取得需要延遲(Delay)的時間

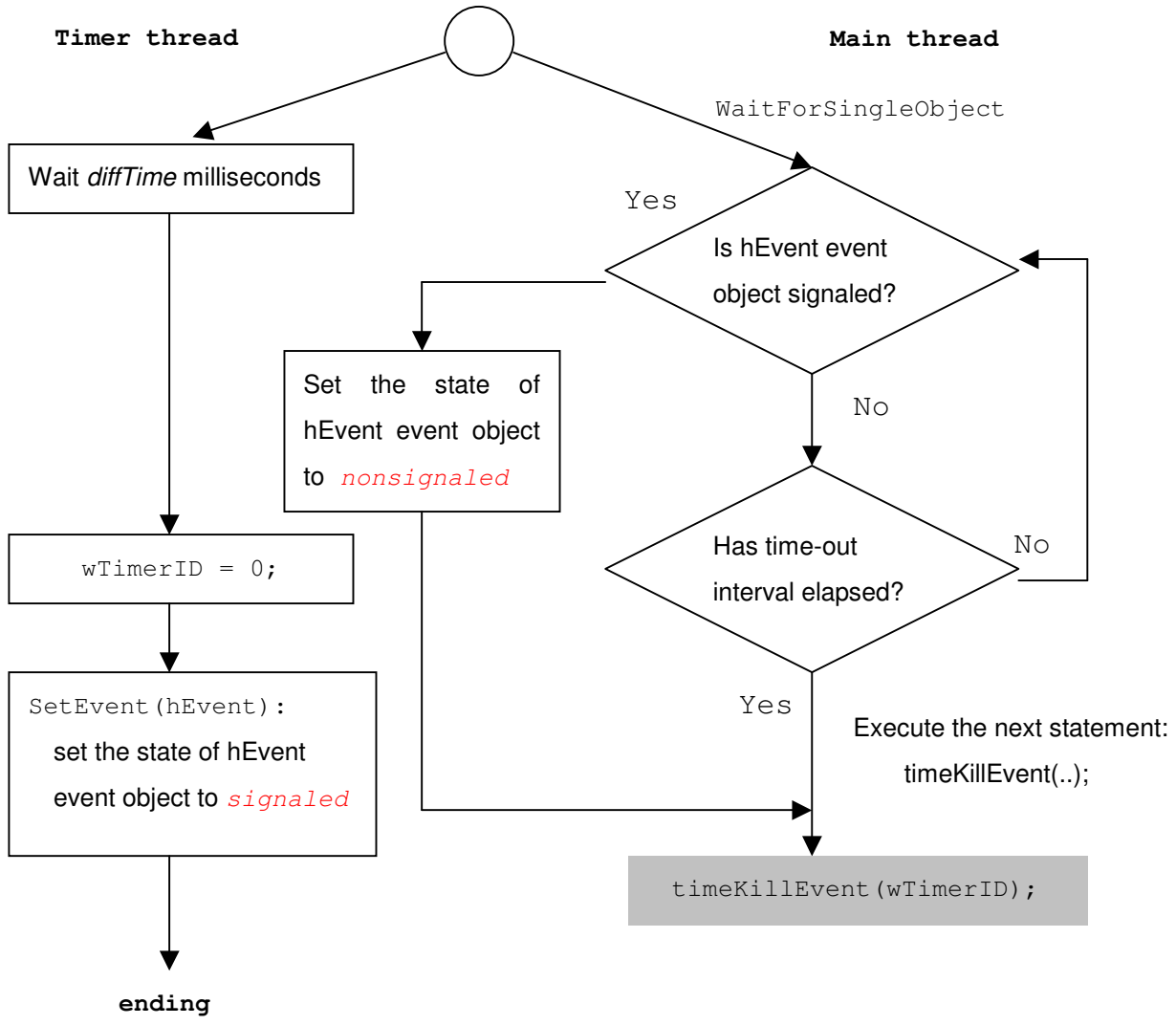
③ wTimerID = timeSetEvent(…);

- 啓動 Timer event，在 diffTime 的延遲(Delay)之後，呼叫 **OneShotTimer** callback function.
- 《Multimedia timer runs in its own thread.》還記得這句話嗎？此時會增加一個 Multimedia timer service thread，所有共有兩個 thread 同時在執行，如下：

Main thread	Timer thread

④ if (!wTimerID) {
printf("Cannot Set Timer Event!\n");
return -1;
}
WaitForSingleObject(hEvent,RateInt+wTimerRes);
	⑤ wTimerID = 0;
	SetEvent(hEvent);

Main thread 執行到 **WaitForSingleObject** 時，會等待 hEvent Event Object；而 **timeSetEvent** 執行後，經過 diffTime milliseconds 後，會呼 **OneShotTimer** callback functionTimer ，此爲 Timer thread。執行的流程圖大概如下：



以上是這個範例程式中最有趣精華的地方，可以花時間研究看看。

□ 參考文獻

1. Multimedia Timer

http://www.msdn.com

2. Event Object

http://www.msdn.com

3. XSleep

http://home.kimo.com.tw/abc9250/LBEE_XSleep.htm