Message Thread Using IoCompletionPort

My2Cents, 2 May 2008 CPOL

3.11 (6 votes)

Thread communication made easy!

Download source - 33.4 KB

Introduction

Windows has a simplified built-in mechanism for communication among threads, using message loops. APIs such as `GetMessage()`, `PeekMessage()` can be used to achieve this. E.g.,

```
MSG msg;
PeekMessage(
    &msg,
    NULL,
    0,
    0,
    PM_NOREMOVE);
while (true)
{
    BOOL bRet = GetMessage(
            &msg,
            NULL,
            0,
            0);
}
```

Once the loop is setup, use `PostThreadMessage()` to send the message across. E.g.,

```
PostThreadMessage(
    nThreadId,
    nMsg,
    wParam,
    lParam);
```

In fact, message based communication is the building block of Windows. Of course, it has serious security implications as the source of the message is not known to the receiver, but that's a topic for another article.

In this article, I used this concept and the MFC macro style message handler declaration and wrapped them in a class that lets you send messages in two different modes:

- `PostMessage`
- `SendMessage` with timeout

The Code

In order to construct the message loop, I used `CreateIoCompletionPort()`,`GetQueuedCompletionStatus()` instead of the above mentioned method. The reasons for this are as follows:

- `IoCompletionPort`s are fast and less resource intensive.
- No restriction as in the other methods where the thread needs to belong to the same desktop, etc.

OK, let's see an example. By the way, the attached files have a Visual Studio based project that you can compile and run. Check the `DemoThread` class.

First, derive a class from `MsgThread`:

```
// DemoThread.h
class DemoThread : public MsgThread
{
    DECLARE_MSG_MAP();
private:
    void OnMsgGoGetPizza(
        WPARAM wParam,
        LPARAM lParam);
};
// DemoThread.cpp
BEGIN_MSG_MAP(DemoThread, MsgThread)
    ON_MSG(kMsgGoGetPizza, OnMsgGoGetPizza )
END_MSG_MAP()
void
DemoThread::OnMsgGoGetPizza
(
    WPARAM wParam,
    LPARAM lParam
)
{
}
```

Elsewhere in your code:

```
DemoThread rThread1;
rThread1.Start();
rThread1.PostMessage(
    kMsgGoGetPizza,
    (WPARAM) 5,
    0);
rThread1.Stop();
```

And that's it. Just be careful; if you use `SendMessage`, you might end up in a deadlock if you haven't designed things properly.

Just include *MsgHandler.[h,cpp]* and *MsgThread.[h,cpp]* in your project. I hope you find it useful. Any comments or suggestions are most welcome.

License

About the Author

My2Cents

Software Developer

United States

No Biography provided