

Re: Creating a simple windows messaging app

Source: <http://www.tech-archive.net/Archive/VC/microsoft.public.vc.mfc/2008-07/msg00518.html>

- *From:* "Nick Schultz" <nick.schultz@xxxxxxxx>
 - *Date:* Thu, 10 Jul 2008 15:50:33 -0700
-

"Scott McPhillips [MVP]" <org-dot-mvps-at-scottmcp> wrote in message news:%231zjOet4IHA.3484@xxxxxxxxxxxxxxxxxxxxxxxxxxxx

"Nick Schultz" <nick.schultz@xxxxxxxx> wrote in message news:eCclgps4IHA.4868@xxxxxxxxxxxxxxxxxxxxxxxxxxxx

Hi there,

a broker app that receives packets from a canbus and pushes them out to multiple applications.

there are two types of threads the broker will have: the main loop and the "app" thread

The broker has a main loop that essentially sits and waits for two types of messages:

-wm_canlib :

canlib notifies the broker by sending a wm_canlib message to a window handle that I assign. the broker then takes the packet and then gives each application thread a copy of the packet.

-(userdefined appregister) :

front end applications that want to receive packets will broadcast this message to get setup for receiving packets.

the main loop will then create a packet queue and spawns off a appthread.

appThread responds to :

-WM_COPYDATA: (Sent from clientApp)

takes a packet sent from client app and sends it off on the bus

-QUEUE_READY: (Sent from mainloop)

gets notified from mainloop that it just put something in its queue.

the appthread will then empty the queue's contents, apply any filters on it, and then send the packet off to its frontend app its responsible for.

so is there a good model to implement this scheme? there's only going to be one main loop, and multiple appthreads (one for each registering app) each thread is going to be needing the ability to send and receive windows messages, including wm_copydata. How do I create the appthreads

Re: Creating a simple windows messaging app

so that it can send/receive messages?

Each appthread should be an instance of a class that you derive from CWinThread. That gives it the ability to process incoming windows messages. You start each appthread by calling AfxBeginThread. To receive WM_COPYDATA each of these threads will need a window. It can be invisible if its only purpose is to receive messages. You don't have to do anything to give the thread an ability to send/post messages: Any thread can do that.

I initially wanted to have all of this in the background (invisible) however I now think its a good idea to at least have a small window indicating that the backend is on and that the user can start up their front end applications. I can also have the ability to display raw data packets as well.

Do all the display work in the main thread.

--

Scott McPhillips [VC++ MVP]

Since I just need a small dialog window , I'm using a dialog based mfc. should I put the wm_canlib and appregister messages in the dialog's message map? The handlers for these messages will then spawn off worker threads to do what needs to get done.

and for the appthreads, I'll extend a CWinThread that has an member that I'll create that extends CWin. I will then put the the messages that i specified above for appThread in the CWin derived class?

is this correct?

Thanks,

Nick

.