C# .Net exe doesn't close when PC is restarted, keeping the machine from restarting

Asked 16 years, 2 months ago Modified 11 years, 7 months ago

Viewed 1k times

5

We have a SmartClient built in C# that stubornly remains open when the PC its running on is being restarted. This halts the restart process unless the user first closes the SmartClient or there is some other manual intervention.

This is causing problems when the infrastructure team remotely installs new software that requires a machine reboot.

Any ideas for getting the SmartClient app to recognize the shutdown/restart event from Windows and gracefully kill itself?

UPDATE: This is a highly threaded application with multiple gui threads. yes, multiple gui threads. Its really a consolidation of many project that in and of themselves could be standalone applications - all of which are launched and managed from a single exe that centralizes those management methods and keeps track of those threads. I don't believe using background threads is an option.

c# multithreading events smartclient restart

Share

Improve this question

Follow

edited Oct 27, 2008 at 20:53

asked Oct 3, 2008 at 18:47

ScottCher

14.9k ● 6 ● 28 ● 26

Are you running stuff on non-background threads?
– user1228 Oct 3, 2008 at 18:52

Do you have access to the source code for the SmartClient app? – Vincent McNabb Oct 3, 2008 at 19:17

6 Answers

Sorted by: Highest score (default)

5

OK, if you have access to the app, you can handle the SessionEnded event.

```
...
Microsoft.Win32.SystemEvents.SessionEnded +=new
  Microsoft.Win32.SessionEndedEventHandler(shutdownHan

...

private void shutdownHandler(object sender,
Microsoft.Win32.SessionEndedEventArgs e) {
  // Do stuff
}
```

Share Improve this answer Follow

edited Oct 3, 2008 at 19:10

answered Oct 3, 2008 at 18:58

Vincent McNabb

34.5k ● 7 ● 33 ● 54

1 Just a note: You probably want the SessionEndING event, not the SessionEnded (if you need to abort threads and close forms etc). – Sire Jan 27, 2010 at 14:13

5

It must be a thread that continues to run preventing your application to close. If you are using threading an easy fix would be to set it to background.

> A thread is either a background thread or a foreground thread. Background threads are identical to foreground threads, except that background threads do not prevent a process from terminating. Once all foreground threads belonging to a process have terminated, the common language runtime ends the process. Any remaining background threads are stopped and do not complete.
>
> http://msdn.microsoft.com/en-us/library/system.threading.thread.isbackground.aspx

Share Improve this answer

Follow

edited Oct 3, 2008 at 19:03

Chris Marasti-Georg

34.6k ● 17 ● 93 ● 137

answered Oct 3, 2008 at 18:55

Catalin DICU

4,638 ● 5 ● 37 ● 47

> Indeed, this is a threaded app. See update to original post. However, given what we're doing with the threads, is backgrounding them still viable? Ie, multiple GUI threads.
> – ScottCher Oct 3, 2008 at 19:00

5

When a user is logging off or Windows is being shut down, `WM_QUERYENDSESSION` message is sent to all top-level windows. See MSDN documentation here.

The default behavior of a WinForm application in response to this message is to trigger the `FormClosing` event with `CloseReason == WindowsShutDown` or others. The event handler though can choose to be stubborn and refuse to shut the app down, thus keeping the system running.

Check `FormClosing` handlers of your applications. Maybe there is something in there. I've seen this kind of stuff a couple of times.

Share Improve this answer

Follow

edited Apr 28, 2013 at 17:37

Patrick D'Souza

3,553 ● 2 ● 24 ● 39

answered Oct 3, 2008 at 19:27

liggett78

11.4k ● 2 ● 31 ● 29

1

Or maybe the .Net app is ignoring close or quit messages on purpose?

Share Improve this answer

Follow

answered Oct 3, 2008 at 18:52

Hristo Deshev

907 ● 6 ● 10

1

Background threads was a quick and dirty solution, best solution is to use synchronization objects (`ManualResetEvent`, `Mutex` or something else) to stop the other threads;

Or else keep track of all your opened windows and sent `WM_CLOSE` message when main app closes.

You have to give more information about how do you start those GUI applications. maybe you start one thread for each application and call `Application.Run(new Form1());` ?

You may also look into creating a `AppDomain` for each GUI Application

Share Improve this answer

edited Apr 28, 2013 at 17:33

Follow

Patrick D'Souza
3,553 ● 2 ● 24 ● 39

answered Oct 3, 2008 at 19:09

Catalin DICU
4,638 ● 5 ● 37 ● 47

0

Normally a .Net app would respond correctly- at least, that's the 'out of the box' behavior. If it's not, there could be a number of things going on. My best guess without knowing anything more about your program is that you have a long-running process going in the main UI thread that's preventing the app from responding to window messages.

Share Improve this answer

Follow

answered Oct 3, 2008 at 18:49

Joel Coehoorn
415k ● 114 ● 577 ● 813

> Not long-running processes but do have Threading as Catalin mentioned above. Still, that doesn't really answer the question - I'm really looking for a way to detect when the machine is being restarted. – ScottCher Oct 3, 2008 at 19:03