

# Re: Batch file and MFC (Properly Terminating Application)

---

*Source:* <http://www.tech-archive.net/Archive/VC/microsoft.public.vc.mfc/2007-02/msg00939.html>

---

- *From:* Joseph M. Newcomer <[newcomer@xxxxxxxxxxxxx](mailto:newcomer@xxxxxxxxxxxxx)>
  - *Date:* Fri, 09 Feb 2007 23:41:38 -0500
- 

See below...

On 9 Feb 2007 11:55:10 -0800, "one-trick-pony" <[worldofpain.aamir@xxxxxxxxxx](mailto:worldofpain.aamir@xxxxxxxxxx)> wrote:

Greetings,

Jopseph:

**\*\*First, you should not ever, ever, for any reason whatsoever, modify, touch, play with, or  
\*\*actually depend on the PATH environment variable. To do so is to say "I want my program  
\*\*to break. I enjoy tech support calls to people who never heard of environment variables,  
\*\*and can't type, and will break their system even worse trying to modify the path. I enjoy  
\*\*pain, suffering, and high post-deployment support costs".**

Above statement opens up a whole new world of interesting possibilities for me. I was expecting for you to tell me to signal Windows that a global change has occurred so that it immediately adapts to PATH variable change. Now you've got me thinking in new but unfortunately totally unfamiliar direction. Usually, when I work with a project, I have some idea and I can start to study up on it like it was in the case of threads. Here, problem is that there is a way to do tasks without needing to depend on PATH environment variable which I am not familiar with. What options do I have? Can you suggest me a link where I can go and study on this? Many times such as when Windows gets updates or service packs it needs to restart itself, why can't my application do that? If I can avoid restarting, I would like to know how else I can accomplish installing software which makes changes to system and yet be able to see those changes immediately so that there is no need for restart.

\*\*\*\*\*

There are many techniques, and the exact technique depends on what you are trying to do. (I've not done anything that required PATH since about 1987, when I had to spend 5 hours on the phone, to Sweden, to an important customer, trying to teach him how to edit AUTOEXEC.BAT so that both our program and his other important program could both run. It

## Re: Batch file and MFC (Properly Terminating Application)

was not aided by the fact that he didn't speak good English, I spoke no Swedish, he only had edlin, which I didn't have, and the PATH was limited to 128 characters. After 5 hours, and a final success, I hung up, turned to my client (I was on-site at the time), and said "Version 2.0 will not use PATH". And it didn't.

We truly hated PATH in Unix; the problem is that it is a poor solution to a relatively simple problem: you want the ability to have an abbreviation. The correct solution is to create a dictionary mechanism, but obvious solutions never occurred to the Unix developers. Instead, they adopted a bankrupt paradigm that had been around (and hated) for at least 15 years before that.(it was old and hated when I started using it in 1967, and the problems were well-known).

There is a predefined PATH which includes the Windows directory, and it should not be touched, modified, or depended upon beyond the standard Microsoft directories. Note that many of these are not required because there is a special "known DLLs" mechanism which handles most of the cases anyway.

Restarting Windows is always a Fundamentally Bad Idea. Back in 1977, when we were designing personal workstations at CMU, the standard out for any problem was "hit the reboot button". I pointed out that this was totally unacceptable. It made no sense in 1977, and it makes no sense today. Mostly the reason an install needs to restart Windows is the obsolete design of the file system; the ability to replace an in-use file was well-understood in 1965, it only requires intelligence and taste to apply it. But we inherited the bad ideas of floppy disk files on the Z80 and pretend it makes sense on terabyte file systems.

Sure, you could restart Windows, but this is unbelievably tasteless. Remember that you are disrupting the user's world; if the app is installed on a 24/7 server, you do not want to shut down the server.

Bottom line: do not do anything that requires a shutdown/restart. Any effort required to make sure this doesn't happen is effort well spent.

Using PATH guarantees a tech support nightmare. I have been screwed by products that require PATH; I have had to deal with problems caused by other code that uses PATH (the Sweden incident is only the worst of the many situations I've had to deal with, in a variety of operating systems, over the last four decades).

What changes do you need to make? You no longer need to replace system DLLs because you can now do side-by-side installs, so that reason is out; you shouldn't be using PATH, so that reason is out; you aren't rolling back a device driver, so that reason is out; so what changes, other than PATH, would require a restart?

\*\*\*\*

To elaborate further, my applicatons heavily edits, modifies, adds and removes registry enteries.

\*\*\*\*

Editing the Registry requires no restart. So that reason is out.

\*\*\*\*

Re: Batch file and MFC (Properly Terminating Application)

It installs Perl and runs MS Installer file(.msi), sets up new custom-made services in SCM database and perhaps the most crucial change is change to network settings.

\*\*\*\*

The need to restart after network settings changed went away in Win2K; what network settings are being changed? Note that one of the ways of avoiding a restart is to use the appropriate mechanisms for making the changes, but I'm not a network configuration guru, so I don't know those APIs. But I do know that Microsoft made a lot of changes to make sure that reboot after network changes were not going to require a restart. Many of these programs were not changed in Win2K, but were changed by XP, so I believe there are mechanisms that avoid a restart.

\*\*\*\*

The way we have our systems is that there are a group of servers that run my application and each of them one gets certain attributes set to them so that they can identify themselves. These group of servers act as a single entity to outside world but internally they are different machines uniquely setup via my application. Some of network setting can not be modified on the fly—a reboot is required for newer settings to take effect. I can get away with not tweaking Path environment variable, but I can not get away with having invalid network settings. Therefore, there is a requirement for my machine to restart the system.

\*\*\*\*

Since this is part of an install, you have to be running with admin priviledges. I would probably use ExitWindowsEx.

\*\*\*\*

like to know, since this is forced onto my application to do a reboot, I want to do it in a good programming manner. The way I have it right now is, at the end of my application last instructions are getting the permission from OS for a reboot and then actually commencing the reboot.

\*\*\*\*

There are lots of different ways to avoid using PATH, but the exact method depends on why you need it.

joe

\*\*\*\*

Thanks

Joseph M. Newcomer [MVP]  
email: [newcomer@xxxxxxxxxxxx](mailto:newcomer@xxxxxxxxxxxx)  
Web: <http://www.flounder.com>  
MVP Tips: [http://www.flounder.com/mvp\\_tips.htm](http://www.flounder.com/mvp_tips.htm)

Re: Batch file and MFC (Properly Terminating Application)