Subject: How to check whether application is running
Posted by bushman on Mon, 09 Aug 2010 01:59:25 GMT
View Forum Message <> Reply to Message

I need to run just one single instance of a TopWindow application per user at a time.
Is there a simple way of doing so in UPP? I mean, how do I check whether the app has been started before in Windows and abort (this) execution if so?

Thanks

Subject: Re: How to check whether application is running
Posted by bushman on Mon, 09 Aug 2010 02:16:13 GMT
View Forum Message <> Reply to Message

```
// global function:
HWND GetRunningWindow() {
 // Check if exists an application with the same TITLE as this application
 HWND hWnd = FindWindow(NULL, "My TopWindow Title");
 if (IsWindow(hWnd)) {
  HWND hWndPopup = GetLastActivePopup(hWnd);
  if (IsWindow(hWndPopup))
  hWnd = hWndPopup; // Previous instance exists
 } else {
  hWnd = NULL; // Previous instance doesnt exist
 }
 return hWnd;
}

GUI_APP_MAIN
{
 // Checks for previous instance of our app
 HWND hOtherWnd = GetRunningWindow();
 // Allow only one app instance
 if (hOtherWnd) {// hOtherWnd != NULL -> Previous instance exists
  PromptOK("The program is already running");
  SetForegroundWindow(hOtherWnd); // Activate it &
  if (IsIconic(hOtherWnd))
   ShowWindow(hOtherWnd, SW_RESTORE); // restore it
  return; // Abort current app execution
 }

// ... otherwise, run app here ...

}
```

Thanks again.

Subject: Re: How to check whether application is running
Posted by dolik.rce on Mon, 09 Aug 2010 02:54:54 GMT
View Forum Message <> Reply to Message

Hi Kropniczki,

This problem was already discussed here some time ago. Have a look at this thread, there are two solutions mentioned (including code) and hopefully helpful discussion.

Honza

Subject: Re: How to check whether application is running
Posted by bushman on Mon, 09 Aug 2010 03:18:08 GMT
View Forum Message <> Reply to Message

The thread you pointed to did help. It does show a UPP-like, much "cleaner" way of doing the job. Thank you for that.

Subject: Re: How to check whether application is running
Posted by Mindtraveller on Tue, 10 Aug 2010 19:28:23 GMT
View Forum Message <> Reply to Message

The *recommended* (by M$) way of detecting other instances of application is using of named mutex. Making mutex is much cheaper and much more atomic operation than creating window. Why is that important?
Imagine you have system with many programs running. And your program makes some complex initialization prior to creating window (i.e. reading XML configuration, constructing objects, etc.)

User runs your app. But OS is busy, it starts swapping. Your app is actually started, but user doesn't see any window, because your app is still initializing (please note that creating window is very resource consuming operation, especially for "busy" OS).

What happens next? User thinks that he did something wrong and executes your app again. This is problem #1.

The problem #2 arises next. Your app which was started second checks window (remember that window of the first app is not finished it's creation), doesn't find it and creates it's own window as if it is the first.

If you think this is rare situation, I must admit it is not. Multitasking can be hard sometimes. And app #2 can potentially be initialized even BEFORE app #1.

That is all why we should use much cheaper and quicker operation in the program start instead of checking opened window.
Let's return to recommended scheme:
1. Check mutex with name "*****"
2. If exists, exit app. If not - immediately create one and start app.
3. Important: all initialization should be done AFTER (1) and (2).

I don't know which is POSIX recommended way. I think our POSIX specialists may consult us.

Subject: Re: How to check whether application is running
Posted by unodgs on Tue, 10 Aug 2010 20:47:43 GMT
View Forum Message <> Reply to Message

I used the "windows way" in ultimate player (bazzar) so you can check it there.

Subject: Re: How to check whether application is running
Posted by kohait00 on Fri, 13 Aug 2010 14:21:12 GMT
View Forum Message <> Reply to Message

i used in my app stuff like this:
which is exactly what Mindtraveler described. code snipet like this can be found online in many places..

the name define doesnt really matter, it was only for me. it has to be something unique though, beeing shure noone else will ever use on your system except you.
with SINGLE_APPINSTANCE you can enable or disable this behaviour

//at beginning of your programm#define SEMAPHORE_UNIQUE "Global\\LIPANv2Unique_2FA34E51"

```
#ifdef SINGLE_APPINSTANCE
#ifdef PLATFORM_WIN32
 USemh = OpenSemaphore(STANDARD_RIGHTS_ALL, FALSE, SEMAPHORE_UNIQUE);
 if(USemh == NULL)
  USemh = CreateSemaphore(NULL, 0, 1, SEMAPHORE_UNIQUE);
 else
 {
  wq & THISBACK1(Closer, "Application already started!!"); //prepare
  return;
 }
#elif defined(PLATFORM_POSIX)
#endif
```

```
#endif

//program runtime....


//at end of your program
#ifdef SINGLE_APPINSTANCE
#ifdef PLATFORM_WIN32
 if(USemh != NULL)
  CloseHandle(USemh);
 USemh = NULL;
#elif defined(PLATFORM_POSIX)
#endif
#endif
```

Subject: Re: How to check whether application is running
Posted by Alboni on Wed, 16 Jan 2013 11:29:13 GMT
View Forum Message <> Reply to Message

```
GUI_APP_MAIN
{
 HANDLE mutex = ::CreateMutex(0, true, "__MYAPP_EXE__");
 if (!mutex)
 {
  PromptOK("Cannot open MUTEX.");
  return;
 }
 if (GetLastError()==ERROR_ALREADY_EXISTS)
 {
  PromptOK("Program is already started.");
  CloseHandle(mutex);
  return;
 }

 App().Run();

 CloseHandle(mutex);
}
```

Subject: Re: How to check whether application is running
Posted by lectus on Wed, 16 Jan 2013 13:38:56 GMT
View Forum Message <> Reply to Message

Another way would be to create a file when the app starts.

The app then always check for the existance of this file. If it exists then there's another instance, so quit.

The app needs to delete the file before finishing.

Subject: Re: How to check whether application is running
Posted by Alboni on Wed, 16 Jan 2013 14:44:27 GMT
View Forum Message <> Reply to Message

Yes, I've seen a number of programs that do just that. Moest of them were on linux.
Disadvantage is that the program doesn't want to start anymore after a crash, and then an admin has to come who knows where that file is located and delete the file.
That could also be feature btw, if the program also checks for the existance of an application window or mutex to determine if it was a crash or a double start and then can take special measures in case of a crash such as rebuilding a databaseindex, submit a crash report or restoring the state of the program.

Subject: Re: How to check whether application is running
Posted by nlneilson on Wed, 16 Jan 2013 16:02:45 GMT
View Forum Message <> Reply to Message

Alboni wrote on Wed, 16 January 2013 03:29
CloseHandle(mutex);

lectus wrote on Wed, 16 January 2013 05:38The app needs to delete the file before finishing.

Alboni wrote on Wed, 16 January 2013 06:44
Disadvantage is that the program doesn't want to start anymore after a crash,

Spent many hours related to this. That is ending a thread correctly.

CloseHandle(mutex);
delete the file before finishing
etc.

Which results in:
program doesn't want to start anymore after a crash,

A thread in many respects is like a separate application.
If it is not ended correctly it causes problems like an app cannot be started again until that thread is closed.

Or for something like a GPS USB receiver if the thread that it is run in does not close that port connection there is no easy way to close that except to pull the plug.

If a socket server has a port open another instance of that app or any other app that uses that port will not run unless error handling changes to another port number.

The best way I have found to close a thread is from INSIDE that thread.

There are many ways to check if an app is running but if that app crashes it can leave something hanging. Task Manager in Win, System Manager in Ubuntu, etc. is good if you know what you are looking for to close that thread.

To deploy an app this needs to be considered and have error handling and even instruction for a user.

If there is some 'magic' option it would be good to know.

Subject: Re: How to check whether application is running
Posted by dolik.rce on Wed, 16 Jan 2013 16:11:19 GMT
View Forum Message <> Reply to Message

BTW, when looking at CreateMutex docs, I found this:
Quote:If you are using a named mutex to limit your application to a single instance, a malicious user can create this mutex before you do and prevent your application from starting. To prevent this situation, create a randomly named mutex and store the name so that it can only be obtained by an authorized user. Alternatively, you can use a file for this purpose. To limit your application to one instance per user, create a locked file in the user's profile directory.
It sounds reasonable and logical... and it never occurred to me before that I should do it

Honza

Subject: Re: How to check whether application is running
Posted by Alboni on Wed, 16 Jan 2013 16:38:55 GMT
View Forum Message <> Reply to Message

Except at least on Windows, if you have rights to execute the program, finding out what an application is creating is a matter of running a debugtool. The app doesn't need debugsymbols for that.
So this scheme would only work if 2 mutexes were used:
The app would then:
1)generate a new name and create a new mutex
2)try to open a mutex with the previous name stored in an encrypted file and if succeeds store this name, if not terminate.

other ideas that I used in practise
1)instead of displaying an error message, you can also bump the running instance.
2)instead of terminating, close the running instance by sending a signal to it, wait for it to

terminate, if it doesn't terminate on time, forcefully kill it. Then proceed starting the app.

Subject: Re: How to check whether application is running
Posted by Alboni on Wed, 16 Jan 2013 17:01:24 GMT
View Forum Message <> Reply to Message

Neil,
The mutex is really a good solution because Windows will destroy it for you if your app terminates incorrectly.
Another idea is the use of memoryfiles, that is files that are created in memory instead of on disk.
I know it can be done with the CreatFile() system call on Windows.
Sadly I don't have sourcecode to share for that anymore because I wrote it for a previous employer.
More info: http://msdn.microsoft.com/en-us/library/windows/desktop/aa366542%28v=vs.85%29.aspx

---

Subject: Re: How to check whether application is running
Posted by BioBytes on Wed, 16 Jan 2013 20:32:22 GMT
View Forum Message <> Reply to Message

Hi all,

I use the following code using SysInfo package:

```
bool QSkillsWin::IsSingleInstance()
{
    if(GetWindowIdFromCaption(GetTitle().ToString())>0)return true;
   else return false;
}

GUI_APP_MAIN
{
 SetLanguage(GetSystemLNG());
 SetLNGCharset(GetSystemLNG(),CHARSET_UTF8);

 if(QSkillsWin().IsSingleInstance())
 {

  return;
 }
```

```
  else QSkillsWin().Run();
}
```

hope this could be helpful

Cheers

Biobytes

Subject: Re: How to check whether application is running
Posted by lectus on Fri, 18 Jan 2013 00:56:45 GMT
View Forum Message <> Reply to Message

Alboni wrote on Wed, 16 January 2013 09:44Yes, I've seen a number of programs that do just that. Moest of them were on linux.
Disadvantage is that the program doesn't want to start anymore after a crash, and then an admin has to come who knows where that file is located and delete the file.
That could also be feature btw, if the program also checks for the existance of an application window or mutex to determine if it was a crash or a double start and then can take special measures in case of a crash such as rebuilding a databaseindex, submit a crash report or restoring the state of the program.

```
void ChangeStatus() {
 SaveFile(GetExeDirFile("lock"), "Finished succesfully");
}
```

```
if(!FileExists(GetExeDirFile("lock")))
    SaveFile(GetExeDirFile("lock"), "Finished sucessfully");

 String status = LoadFile(GetExeDirFile("lock"));

 if(status == "running")
 {
  Exclamation("There's another instance!");
  return;
 }

 if(status == "crashed") {
  Exclamation("The program crashed on the last run. Please restart the application.");
  RecoverData();
```

```
 ChangeStatus();
 return;
}

 SaveFile(GetExeDirFile("lock"), "running");

 atexit(ChangeStatus);
```

Before critical part of the program:

```
SaveFile(GetExeDirFile("lock"), "crashed");
DoLongOperations(); // if it crashes now we're safe
SaveFile(GetExeDirFile("lock"), "running"); // We survived
```

If you want to get fancy, add time stamps.
