

Forum:[Hack Forums](#)

Thread: [\[RELEASE\]\[UPDATED\] AutoUpdate class \[READY TO USE \]\[NO FILE GENERATED!\]](#)

Gr0wlit - March 29, 2015, 2:53 pm

UPDATED: Now it doesn't generate any file.

I've applied He B TeMy and Only3 suggestions.

Hi all,

one of the most important feature that your application can have is the AutoUpdate feature: each application, even if developed by the most experienced coder, can have bugs; it is developer's work trying to build the more "bugs free" software to release to his costumers.

Since I don't see a lot of thread about it, I'm currently releasing a perfectible ready to use class that will help you to implement the AutoUpdate feature to your products.

WHAT YOU NEED:

URL of a text file that display the most updated version (for example 2.0.0.1)

URL of the updated application ready to download

HOW IT WORKS:

Basically, the idea is simple (and I'm sure already used by someone):
since creating 2 different exe (main application and updater) is a TERRIBLE idea in my opinion, we will work with cmd commands;

first, the class will check the most updated version, if there is a new version, will download the updated software in the Temp folder (all asynchronously to avoid UI freezing).

Then it will generate some CMD instructions that will (in order):

Delete your application;

Move the updated application from the Temp folder to the current folder;

Open the updated application;

All this will be launched in HIDDEN mode, the final user will not see anything (as you can see in the .gif);

before running these instructions, it will close your application (otherwise the update process will not work).

As a result, you will have your updated application opened, without any other file!

ANIMATED GIF:

Spoiler ([Click to View](#))

[img]http://i.minus.com/ifJO6xGX5Oj4H.gif[/img]

HOW TO USE IT:

Just import the class into your project, to call it you just need to instantiate the class and pass 3 parameters:

URL of the version txt file on your server

URL of the updated application in your server

Application.ExecutablePath

Pretty cool things that I liked to add:

my autoupdate class provide 3 events that you can call from your application: UpdatePercent (percent as integer): this method is fired everytime the downloading percent of the updated application changes (you can use it to update the progressbar as shown in the animated .gif)

Status(status as string): fired everytime the routine changes (useful in updating the status in your main app as shown in animated .gif)

DownloadComplete: fired when the updated application is downloaded

First: declare the class:

PHP Code

```
<?php
```

```
Private WithEvents AU As AutoUpdateClass
```

then, when you want to check if there are updates (for example in a button):

PHP Code

```
<?php
```

```
AU = New AutoUpdateClass(URLVersionTxt, _  
    URLUpdatedApplication, _  
    Application.ExecutablePath)  
AU.Run()
```

Finish! My autoupdate class will do the rest.

To call the event provided by the class:

PHP Code

```
<?php
```

```
Private Sub DownloadProgress(ByVal nPercent As Integer) Handles AU.UpdatePercent  
    pbLoading.Value = nPercent  
End Sub
```

```
Private Sub DownloadComplete() Handles AU.DownloadComplete
```

```
End Sub
```

```
Private Sub CheckStatus(ByVal sStatus As String) Handles AU.Status
```

```
    lblStatus.Text = sStatus
```

```
End Sub
```

DOWNLOAD

<https://mega.co.nz/#!I4E2CaAD!mmIXJ7ojNf-OECsxJlFnZizUbW8M4XyLMnKy0FG47XM>

He B TeMy - March 29, 2015, 3:24 pm

You talk about not using an Updater because that's a bad idea, but you're fine with writing a bat file to the disk and running it?

```
Code .AppendLine("TASKKILL /F /IM " & _CurrentPath.FileName)
```

Using this to kill your program inside the .bat? lol, what if another program has the same name of yours?

```
Code AppendLine("del "" & _CurrentPath.Path & _CurrentPath.FileName & """)
.AppendLine("move /-y "" & _TempPath.Path & _TempPath.FileName & "" "" &
_CurrentPath.Path & """)
```

Can you read what the commands do please?

Why are you using /-y parameter ? You know what happens if the file exists with that parameter?

```
Code .AppendLine("sleep 2")
```

Does this even work?

```
Code .AppendLine("call :deleteSelf&exit /b")
.AppendLine(":deleteSelf")
.AppendLine("start /b "" "" cmd /c del ""%~f0""&exit /b")
```

Wtf is this for? What does the last line do?

```
.CreateNoWindow = True
```

Gr0wlit - March 29, 2015, 3:44 pm

That is why I wrote the "Things to consider" section. Anyway, let me answer your questions.

(Mar 29, 2015 03:24 PM)He B TeMy Wrote: [\[CITE\]](#) You talk about not using an Updater because that's a bad idea, but you're fine with writing a bat file to the disk and running it?

Uhhh yes? What's the problem? A batch file is more elegant than a separated exe that do the job. I don't see the problem.

Quote:Code .AppendLine("TASKKILL /F /IM " & _CurrentPath.FileName)

Using this to kill your program inside the .bat? lol, what if another program has the same name of yours?

Basically the only way to delete an .exe from a batch script that I know is what I wrote (Things to consider section).

```
Quote:Code AppendLine("del "" & _CurrentPath.Path & _CurrentPath.FileName & """)
.AppendLine("move /-y "" & _TempPath.Path & _TempPath.FileName & "" "" &
_CurrentPath.Path & """)
```

Can you read what the commands do please?

Why are you using /-y parameter ? You know what happens if the file exists with that parameter?

Yes, why it is a bad idea?

```
Quote:Code .AppendLine("sleep 2")
```

Does this even work?

My fault, I was trying different ways and I forgot to delete that line.

Quote:Code .AppendLine("call :deleteSelf&exit /b")
 .AppendLine(":.deleteSelf")
 .AppendLine("start /b """" cmd /c del ""%~f0""&exit /b")
Wtf is this for? What does the last line do?

.CreateNoWindow = True

That line call a subroutine immediately below that will delete the script itself. I used to call a subroutine to avoid problems.

evilivid - March 29, 2015, 3:52 pm

LOL this is so clever, I might use this.

Yeah it works, the thread is 10/10 good work OP.

He B TeMy - March 29, 2015, 5:42 pm

(Mar 29, 2015 03:44 PM)GrOwlit Wrote: [\[CITE\]](#) Uhhh yes? What's the problem? A batch file is more elegant than a separated exe that do the job. I don't see the problem.

Is more elegant? Because it's smaller?

How about you run it directly on cmd without writing to disk?

(Mar 29, 2015 03:44 PM)GrOwlit Wrote: [\[CITE\]](#) Basically the only way to delete an .exe from a batch script that I know is what I wrote (Things to consider section).

How about you close the application gracefully just after you launch the script? And perhaps do a little pause on the script to let the application close first.

(Mar 29, 2015 03:44 PM)GrOwlit Wrote: [\[CITE\]](#) Yes, why it is a bad idea?

No you don't, otherwise you'd know the bad idea, there's no point in supplying that parameter because it's the default, you probably know now but i tell you anyway, that forces every overwrite action to prompt the user, i know you do a "delete" before, but if for some reason that doesn't work, you're stuck with prompting the user something which will possibly break the application.

... And, since i forgot to finish that last line above... you can remove your .CreateNoWindow = true, it does nothing on your code.

Edit: Yes.

```
Code Private _CurrentPath As CurrentPath
Private Structure CurrentPath
Dim Path As String
Dim FileName As String
End Structure
```

What is this for? Ever heard of Code Application.ExecutablePath ?

Hawk HF - March 29, 2015, 7:24 pm

This is a nice thread and is a much greater contribution than many members have given. While there may be a few areas that could be improved upon, I appreciate the time you have taken to contribute to the .NET section. Nice release.

nuk - March 29, 2015, 7:49 pm

I agree with Hawk... this is very useful and is going to help a lot of members out ;) Thanks for sharing this OP! ;)

Gr0wlit - March 29, 2015, 10:32 pm

(Mar 29, 2015 05:42 PM)He B TeMy Wrote: [\[CITE\]](#) Is more elegant? Because it's smaller?
How about you run it directly on cmd without writing to disk?
No, because is a smart and clean way to do that operation.
Run directly from CMD can be a great idea, I will give a try

Quote:How about you close the application gracefully just after you launch the script? And perhaps do a little pause on the script to let the application close first.

I wanted to do an independent class, without this kind of interaction with the main app.

Quote:No you don't, otherwise you'd know the bad idea, there's no point in supplying that parameter because it's the default, you probably know now but i tell you anyway, that forces every overwrite action to prompt the user, i know you do a "delete" before, but if for some reason that doesn't work, you're stuck with prompting the user something which will possibly break the application.

Didn't think that could break the application, that's why I've used it. I will try

Quote:Code Private _CurrentPath As CurrentPath
Private Structure CurrentPath
Dim Path As String
Dim FileName As String
End Structure

What is this for? Ever heard of Code Application.ExecutablePath ?

Are you able to read the whole code? You will see that I use it to valorize some other parts of code, in order to avoid to write a code full of GetDirectoryName ecc, that make more confusion.
Yes I've heard of it (as you can read is a parameter that you pass at class initialization).

Anyway, other parts of code that you want to criticize? Maybe some name of variables?

Only3 - March 30, 2015, 3:15 am

u dont need bat file <https://hackforums.net/showthread.php?tid=1341791>

Gr0wlit - March 30, 2015, 2:22 pm

I've UPDATED the class. Now it doesn't generate any file, it will run an hidden cmd window that do the trick.

Important thing to consider is that the class will try to close your application, and after 2 seconds it will run the code..

The application must be closed in order to update.

He B TeMy - March 30, 2015, 2:41 pm

You should give atleast 10s for it to close tho...

I appreciate you taking feedback, just one more thing as i told you, you can remove Code .Append("del "" & _CurrentPath.Path & _CurrentPath.FileName & "" && ") since the next line will overwrite it anyway, unless you want to allow the application name to change...

Gr0wlit - March 30, 2015, 2:54 pm

(Mar 30, 2015 02:41 PM)He B TeMy Wrote: [\[CITE\]](#) You should give atleast 10s for it to close tho...

I don't know, 10s can be to much for a kind of application and not enough for other.. Maybe I will put is as a parameter..

Quote:I appreciate you taking feedback, just one more thing as i told you, you can remove Code .Append("del "" & _CurrentPath.Path & _CurrentPath.FileName & "" && ") since the next line will overwrite it anyway, unless you want to allow the application name to change...

Yes, it can be useless but I haven't deleted it just because in this manner I have one more confirm that the file is removed..

TazzyOpz - March 30, 2015, 2:57 pm

(Mar 30, 2015 03:15 AM)Only3 Wrote: [\[CITE\]](#) u dont need bat file <https://hackforums.net/showthread.php?tid=1341791>

This is a much better method. I've also used that it in the past.

(Mar 30, 2015 02:22 PM)Gr0wlit Wrote: [\[CITE\]](#) I've UPDATED the class. Now it doesn't generate any file, it will run an hidden cmd window that do the trick.

Important thing to consider is that the class will try to close your application, and after 2 seconds it will run the code..

The application must be closed in order to update.

Indeed. You could always force close your application though as well to ensure the process is no longer running. Just Kill the process via it's Directory. However I'd only do that if you're running into issues where your program stays running in memory.

He B TeMy - March 30, 2015, 3:07 pm

(Mar 30, 2015 02:54 PM)Gr0wlit Wrote: [\[CITE\]](#) Yes, it can be useless but I haven't deleted it just because in this manner I have one more confirm that the file is removed..

That doesn't make sense man... "one more confirm"?

If the first fails, the move will fail too, because it can't overwrite if something bad happens anyway. But hey, i'm just sayin'.

And the 10s stuff i'd put it because of a generalization, i know some apps need 1s, others may need more, but Application.Exit() should terminate it in 10seconds (*should*).

But that must be adjusted anyway if people do you use your class, they'll know better. (i hope)

(Mar 30, 2015 02:57 PM)TazzyOpz Wrote: [\[CITE\]](#) Indeed. You could always force close your application though as well to ensure the process is no longer running. Just Kill the process via it's Directory. However I'd only do that if you're running into issues where your program stays running in memory.

You should never kill a process like that.

Don't suggest a kill method on a Release of a class for public usage, it's just bad.

Whoever uses this class should know how many seconds to wait in the batch file if they don't really want to use a separate application to update... if they fail the calculations, that's their fault, but don't just kill an app.

TazzyOpz - March 30, 2015, 4:00 pm

(Mar 30, 2015 03:07 PM)He B TeMy Wrote: [\[CITE\]](#) You should never kill a process like that.

Don't suggest a kill method on a Release of a class for public usage, it's just bad.

Whoever uses this class should know how many seconds to wait in the batch file if they don't really want to use a separate application to update... if they fail the calculations, that's their fault, but don't just kill an app.

Quote:"However I'd only do that if you're running into issues where your program stays running in memory"

I'm not saying you should do that I'm say you could do that as an options if a task is still running in

the program while you want to update. I know it's not the best. I'm just saying it can be done that way if needed.

ManDoR - March 30, 2015, 5:47 pm

Actually, having an updater application running in the background is much cleaner than what you're doing. Nowadays, performances aren't an issue anymore. It could be seen as micro-services, which tend to be the new standard. Your solution involves too many flaws, which have already been mentioned above. What if the application couldn't close on time? What if you're not running on a Windows operating system (thanks to Mono)? But, more importantly, you should never force an application to close by killing its process.

These flaws are all solved very easily with an independent application running in the background. In fact, the main app could look for some events that the updater would throw, such as when a new update is available. This would lead to the main application gracefully closing. The updater would then proceed to the downloading and updating sequences, and launch back the application. This is the way to go, and this is why pretty much every application do it this way.

Gr0wlit - March 31, 2015, 4:16 am

(Mar 30, 2015 05:47 PM)ManDoR Wrote: [\[CITE\]](#) Actually, having an updater application running in the background is much cleaner than what you're doing. Nowadays, performances aren't an issue anymore. It could be seen as micro-services, which tend to be the new standard. Your solution involves too many flaws, which have already been mentioned above. What if the application couldn't close on time? What if you're not running on a Windows operating system (thanks to Mono)? But, more importantly, you should never force an application to close by killing its process.

These flaws are all solved very easily with an independent application running in the background. In fact, the main app could look for some events that the updater would throw, such as when a new update is available. This would lead to the main application gracefully closing. The updater would then proceed to the downloading and updating sequences, and launch back the application. This is the way to go, and this is why pretty much every application do it this way.

Of course my solution isn't perfect, but it isn't so bad as you think.

First, I don't kill any process, the application is closed just by the Application.Exit instruction from my class, which isn't an hardcore way to do it.

Yes, what happens if the application isn't closed in time?

It simply download the updated application in the temp folder and doesn't replace the current one, the updated version will be of course deleted when PC will be shutted down, since it is in the TEMP folder, just this, the current application will continue to work correctly, no errors will be displayed.

But sincerely, it's not my problem if the application could be not closed in time. If before closing I need to do some stuffs, I just do these stuffs and then I will call my class. All can customize their classes since I've released the source code.

Since this is mainly Visual Basic.NET section, I supposed that this will be run on Windows machines. I can't calculate Every casuistry.

This isn't the best way to do an autoupdate, but it is simple, clean enough and it works well.

Only3 - March 31, 2015, 10:28 am

the way i will do it

He B TeMy - March 31, 2015, 3:27 pm

(Mar 31, 2015 10:28 AM)Only3 Wrote: [\[CITE\]](#) the way i will do it

That is fragile code at best.

And since you apparently want to shorten everything up, you can replace the Environment. stuff

with End.

Only3 - *March 31, 2015, 3:41 pm*

yhe i use it cuz Application.Exit() show the the form close end seems to work too

Gr0wlit - *April 1, 2015, 4:23 am*

(Mar 31, 2015 10:28 AM)Only3 Wrote: [\[CITE\]](#) the way i will do it

I Don't see any point in your solution. It's just a more "hard to read" class that basically doesn't provide anything more than my class, but it removes some good features like asynchronous downloading.

Only3 - *April 1, 2015, 4:42 am*

it wasnt mean to be better or shorter then urs any one and his way just show u simple code of doing it didnt want to make thread for this