

Forum:[Hack Forums](#)

Thread: [\[Release\] TinyInstall.cs - AUTO INSTALLER CLASS - SUPPORTS .NET 2.0+++](#)

gigajew - August 16, 2017, 1:35 pm

Do you ever rewrite the installation logic for your projects? No more. Let this class do the job for you from now on. Whether it's relying on .NET 2.0 or .NET 4.6.1, this class will automatically install your software on the local machine. With it's easy-to-use and slim design, this code will lift your project above others.

Features:Melting mechanism (by moving and not copying)

Set file attributes to hidden automatically

Usage

Code

0xDEAD10CC - August 17, 2017, 9:03 am

Quote:With it's easy-to-use and slim design, this code will lift your project above others.

? That is laughable.. It isn't even generic enough to deal with the majority of things required by an installer. :/

gigajew - August 17, 2017, 12:10 pm

(Aug 17, 2017 09:03 AM)0xDEAD10CC Wrote: [\[CITE\]](#) ? That is laughable.. It isn't even generic enough to deal with the majority of things required by an installer. :/

Yes it's very amazing, I am astounded with you

0xDEAD10CC - August 17, 2017, 12:27 pm

(Aug 17, 2017 12:10 PM)gigajew Wrote: [\[CITE\]](#) Yes it's very amazing, I am astounded with you
Please explain how this is good in any way? Or how it would be better than all of the real installer setups that are available out there? There's a reason a few of them have been around for years and have been used for years. What makes this good? Everytime I see a post by you here the percentage of quality source code snippets on this forum must decrease.

Code private string _appName;

```
public string AppName {
    get {
        return GetAppName();
    }
    private set {
        _appName = value;
    }
}
```

```

public TinyInstall() { }
public TinyInstall(string appname) {
    if(!SetAppName(appname)) {
        throw new Exception("Invalid app name, or app name already taken!");
    }
}

public string GetAppName() {
    if(!string.IsNullOrEmpty(_appName))
    {
        return _appName;
    }else {
        return typeof(TinyInstall).FullName;
    }
}

```

For instance, a lot of the problems implied by this absolute mess of code could be prevented if you understood constructors, and didn't provide a parameterless one that didn't set the backing field to the default value you use within the function...

Once again even your logic isn't even accurate:

Code public bool Uninstall()

```

{
    if (!AppNameTaken(AppName))
    {
        return true; // how is this indication of a true uninstall?
    }
    ...
}

```

ForceJew - August 17, 2017, 3:59 pm

Sadly Oxdead is right.

The code is messy

DeadKube - August 18, 2017, 8:15 pm

This isn't an installer by the definition of the word.

This does not install anything. All it does is check if a particular registry key has been added to the startup and possibly move itself to the appdata folder.

(Also, what do you think will happen if you call your app ccleaner but ccleaner is already been added to startup, it will fail for all your malicious intents and purposes)

(Also what do you think would happen if another piece of malware using the same name as yours which is also in the appdata folder when calling the installexecutable method?)

And yes, you will get a failure if you do not initialize your string value using the default constructor and then asking for the app name.

And yes, your uninstall feature does not even remove the file from disk which means for all your malicious intents and purposes that your evidence gets left behind.

Again, this is not an installer. If anything it's basically a module that could be used for malware to add or remove itself to the startup sequence of Windows. Honestly, any decent malware developer will already know how to handle this and how to handle duplicate registry key names and the likes. So really, I see no use for this, sorry.

But yes, adding your software to the startup sequence of Windows has genuine purposes, the other function you have there, copying to the appdata folder in combination with the adding to startup does not.

Edit: You move the file to the appdata folder yet the registry key will still point to the old location, in essence this won't even work to add yourself to startup. Lol. Am I missing something here? I didn't actually run your code, just going off pure logic.

masterpill - *August 19, 2017, 2:31 pm*

Hmm nice share just for a simple installation on an app and making start on computer boot

SNAKEGOTCHA - *August 19, 2017, 3:20 pm*

This is pretty nice man. Good share.

DeadKube - *August 21, 2017, 6:36 pm*

How can this be nice if it doesn't do what it's supposed to do? It can't add itself to startup. Please, am I missing it? Is my logic corrupted? I can't test it. Please save me from OCD.