

Forum:[Hack Forums](#)

Thread: [Twitter Scraper - Scrape Data Off Twitter Accounts](#)

fIAmingw0rm - December 29, 2017, 3:33 pm

So I just recently got back to learning C# and wrote this app to learn more about interfaces, OOP, httpclient, tasks, serialization etc. There are things that can be done better, such as using HTML Agility Pack instead of Regex to parse HTML, serialization, csv libraries and some other stuff. I'll be doing that in a separate project to learn it before applying it to an app while learning.

If you have any kind of feedback or suggestions how to improve the app or my code overall, feel free to post it. I'm trying to get better and avoid bad practices so no matter how bad it is just shoot.

Source: <https://goo.gl/U9ZDk> (first time using git so lmk if something is wrong)

CageCoin - December 29, 2017, 3:56 pm

Read over most of the code briefly, I think it looks good and you adhered to good practices pretty much everywhere. I'm not a fan of the exception class you created that does nothing but call the base class, and if it was me I would prune that but other than that good job.

Ninja - December 29, 2017, 5:32 pm

I looked over it briefly, and here is my 2 cents;

You don't use constants (for example in the TwitterScraper class).

The InvalidUsernameException does not have a property for a username string.

Directory.CreateDirectory returns the DirectoryInfo if it already exists, so you don't have to check if it exists first.

You use an awful lot of static classes, which are in most cases considered 'bad practice' in OOP.

You should consider using a BindingList for the DataGridView, so it automatically gets updated when you add to the list. The items you put in should inherit the INotifyPropertyChanged interface.

You can use object initializers (similar to the With statement in VB.net) instead of manually setting a property for an instance (for example in the Request class). This can also be collapsed in the designer. This is up to personal preference though.

In the proxy class you specify an empty setter for the webProxy property (which should start with a capital if you want to follow the Microsoft C# standard), and in the getter you create the proxy every time it is called, instead of instantiating it once and saving that instance and returning that instance in the getter.

You don't use properties for the internal variables in the Settings class.

In the export class there is no need for the DataGridView property, instead you should pass it in the constructor (if you wanna go with the style you have now) and save it in a private field.

In the export class you should use else-if statements instead of putting if statements behind each other, so you won't do checks on conditions that it won't pass anyways.

I suggest for the Export class to pass a collection instead of a whole DataGridView instance, and then iterating over that collection. This is if you go the BindingList route.

And finally, what is up with all empty methods and classes?

+Haze - December 29, 2017, 7:41 pm

(Dec 29, 2017 06:32 PM)Ninja Wrote: [CITE](#) I looked over it briefly, and here is my 2 cents;

You don't use constants (for example in the TwitterScraper class).

The InvalidUsernameException does not have a property for a username string.

Directory.CreateDirectory returns the DirectoryInfo if it already exists, so you don't have to check if

it exists first.

You use an awful lot of static classes, which are in most cases considered 'bad practice' in OOP.

You should consider using a BindingList for the DataGridView, so it automatically gets updated when you add to the list. The items you put in should inherit the INotifyPropertyChanged interface.

You can use object initializers (similar to the With statement in VB.net) instead of manually setting a property for an instance (for example in the Request class). This can also be collapsed in the designer. This is up to personal preference though.

In the proxy class you specify an empty setter for the webProxy property (which should start with a capital if you want to follow the Microsoft C# standard), and in the getter you create the proxy every time it is called, instead of instantiating it once and saving that instance and returning that instance in the getter.

You don't use properties for the internal variables in the Settings class.

In the export class there is no need for the DataGridView property, instead you should pass it in the constructor (if you wanna go with the style you have now) and save it in a private field.

In the export class you should use else-if statements instead of putting if statements behind each other, so you won't do checks on conditions that it won't pass anyways.

I suggest for the Export class to pass a collection instead of a whole DataGridView instance, and then iterating over that collection. This is if you go the BindingList route.

And finally, what is up with all empty methods and classes?

Your post would be more helpful if you actually made a reference to the lines you're talking about, we cannot read your mind.

Quote:You use an awful lot of static classes, which are in most cases considered 'bad practice' in OOP.

See you point out it's bad but you don't say why it's bad. Using statics isn't inherently bad, as long as a static class is appropriate for the task then it's fine.

Quote:The InvalidUsernameException does not have a property for a username string.

Why does he need it? Him deriving from the Exception class is clear enough.

Quote:You don't use constants (for example in the TwitterScraper class).

I've looked in the class and I see nothing that would benefit from a constant.

(Dec 29, 2017 04:33 PM)fIAmingw0rm Wrote: [\[CITE\]](#) So I just recently got back to learning C# and wrote this app to learn more about interfaces, OOP, httpclient, tasks, serialization etc. There are things that can be done better, such as using HTML Agility Pack instead of Regex to parse HTML, serialization, csv libraries and some other stuff. I'll be doing that in a separate project to learn it before applying it to an app while learning.

If you have any kind of feedback or suggestions how to improve the app or my code overall, feel free to post it. I'm trying to get better and avoid bad practices so no matter how bad it is just shoot.

Source: <https://github.com/rist0/TwitterScraper> (first time using git so lmk if something is wrong)
I shall post my feedback and I will write my own version when I wake up this morning, I've found some more things that could be improved on. I like the attitude on wanting to improve, such attitude is what helps you to grow as a developer, I did it before and got shredded for it, but it's well worth it.

0xDEAD10CC - December 30, 2017, 12:20 am

```
Code catch(InvalidUsernameException)
{
```

```
        throw new InvalidUsernameException(String.Format("Username {0} is not valid",
username));
    }
```

Why do you do this? Just re-throw normally rather than throwing a new exception in the catch block. That doesn't make sense above.

flAmingw0rm - December 30, 2017, 4:27 am

The exception class I did just to see how it works and I had no idea what else I could do inside it. Still kinda don't. What would you put inside it beside the base message? I just wanted to have different exceptions so I can handle them differently later on.

Thanks for all the feedback, keep it coming. I'll try to do a better job in my next project (and further versions of this one as well).

Ninja - December 30, 2017, 8:53 am

(Dec 29, 2017 08:41 PM)+Haze Wrote: [\[CITE\]](#) Your post would be more helpful if you actually made a reference to the lines you're talking about, we cannot read your mind.

If I see it on multiple occasions (manually setting properties over using an object initializer) I assume that is his style, thus I don't need to tell him where. If I am talking about specific things like Directory.Exists he should know where it is (hint: export class). I would like you to give me one of the remarks and tell me how that is unclear as to where it is. I will not be editing the post.

(Dec 29, 2017 08:41 PM)+Haze Wrote: [\[CITE\]](#) See you point out it's bad but you don't say why it's bad. Using statics isn't inherently bad, as long as a static class is appropriate for the task then it's fine. I don't know, all I am saying is that it is considered in most cases to be 'bad practice'. Notice the quotation marks as well?

(Dec 29, 2017 08:41 PM)+Haze Wrote: [\[CITE\]](#) Why does he need it? Him deriving from the Exception class is clear enough.

First of all, he is re-throwing, so the exception is to be handled at a higher level.

Second, why the fuck not? It is an InvalidUsernameException, it's responsibility is to tell you about an invalid username. He puts the username in the exception message, but what if he wants to handle it in the code in the future? It just makes sense. "Why does he need it?" is such a shit argument in this case.

(Dec 29, 2017 08:41 PM)+Haze Wrote: [\[CITE\]](#) I've looked in the class and I see nothing that would benefit from a constant.

In the TwitterScraper class, all the Regex patterns. He has a string variable and keeps changing its value. Imo it is better to declare a separate constant string for each pattern, and name the pattern accordingly. This looks cleaner in the code as the code becomes more descriptive.

Why do you always nit pick what I say? Whenever I post something you don't like you dissect what I say and you have an argument against every fucking sentence I type. I never see you do this with anyone else. I see people giving bad advice all the time in this section, yet you never open your mouth. When I say something that does not fit your style, which obviously is superior to mine (sarcasm by the way), you immediately hop on the rape train and talk me down.

+Haze - December 30, 2017, 6:52 pm

Ninja Wrote: [\[CITE\]](#) x

Fair enough, you're right about the regex patterns.

Quote: Why do you always nit pick what I say? Whenever I post something you don't like you dissect what I say and you have an argument against every fucking sentence I type. I never see you do this with anyone else. I see people giving bad advice all the time in this section, yet you never open your mouth. When I say something that does not fit your style, which obviously is superior to mine (sarcasm by the way), you immediately hop on the rape train and talk me down.

I'm sorry you feel this way but that definitely wasn't my intention. I know bad advice exists on the forum but I'm barely active anymore and if I were I would say something about it. I'll work on the way I type so you don't feel like I'm attacking you. ?

(Dec 29, 2017 04:33 PM)flAmingw0rm Wrote: [\[CITE\]](#) So I just recently got back to learning C# and wrote this app to learn more about interfaces, OOP, httpclient, tasks, serialization etc. There are things that can be done better, such as using HTML Agility Pack instead of Regex to parse HTML, serialization, csv libraries and some other stuff. I'll be doing that in a separate project to learn it before applying it to an app while learning.

If you have any kind of feedback or suggestions how to improve the app or my code overall, feel free to post it. I'm trying to get better and avoid bad practices so no matter how bad it is just shoot.

Source: <https://github.com/rist0/TwitterScraper> (first time using git so lmk if something is wrong)
Right, so because I am short on time I didn't have the time to give you extensive detailed feedback as I usually would. However, there was something I did want to help you "learn". You mentioned that you didn't like your use of regular expressions to parse HTML and I agree it can look very cumbersome in your example.

A reliable and great HTML parser I like to use is Angle Sharp. Using AngleSharp makes parsing the right content a breeze and easy for other developers to understand your work. It's far better than HTML Agility Pack in my opinion.

<https://github.com/AngleSharp/AngleSharp>

Here's a small git gist I made for you to show how it can be done (this is just an example and doesn't contain stuff like validation or whatever)

<https://gist.github.com/anonymous/58e0989cdeb12fc0f5d7a76f9c427230>

The reason for wanting to show you an example was before, I was one to preach about using HTML parsers to parse HTML, but I would use regular expression and part of the reason I used regex was because I thought it was easier, I thought HTML parsers were more complicated to use and it turned out I was wrong, they're far easier. Also, I didn't like importing NuGet packages before because I wanted my classes to be "lightweight" and only use libraries from the .NET framework. After working as a dev in the industry, I quickly realised that I was mistaken.

Also another tip for you is to not "over engineer" your work, you mention you wanted to learn about object oriented programming and make use of interfaces which is all good but don't use an interface when it's not needed, keep it simple. I don't think you needed an interface in your example, unless you plan on expanding your project to support more than one scraper "provider". E.g. One for instagram, twitter, facebook etc.

0xDEAD10CC - December 30, 2017, 8:34 pm

(Dec 30, 2017 05:27 AM)flAmingw0rm Wrote: [\[CITE\]](#) The exception class I did just to see how it works and I had no idea what else I could do inside it. Still kinda don't. What would you put inside it beside the base message? I just wanted to have different exceptions so I can handle them differently later on.

Thanks for all the feedback, keep it coming. I'll try to do a better job in my next project (and further versions of this one as well).

You're catching and throwing the exact same exception type. Just
Code throw;
If it can't be handled..

Ninja - December 30, 2017, 8:50 pm

(Dec 30, 2017 07:52 PM)+Haze Wrote: [\[CITE\]](#) Fair enough, you're right about the regex patterns. Only the regex patterns? :(

(Dec 30, 2017 07:52 PM)+Haze Wrote: [\[CITE\]](#) I'm sorry you feel this way but that definitely wasn't my intention. I know bad advice exists on the forum but I'm barely active anymore and if I were I would say something about it. I'll work on the way I type so you don't feel like I'm attacking you. ?
Apology accepted my nibba

(Dec 30, 2017 07:52 PM)+Haze Wrote: [\[CITE\]](#) The reason for wanting to show you an example was before, I was one to preach about using HTML parsers to parse HTML, but I would use regular expression and part of the reason I used regex was because I thought it was easier, I thought HTML parsers were more complicated to use and it turned out I was wrong, they're far easier. Also, I didn't like importing NuGet packages before because I wanted my classes to be "lightweight" and only use libraries from the .NET framework. After working as a dev in the industry, I quickly realised that I was mistaken.

This. I didn't use Nuget packages until a year ago. Before I reinvented the wheel so many times.

(Dec 30, 2017 07:52 PM)+Haze Wrote: [\[CITE\]](#) Also another tip for you is to not "over engineer" your work, you mention you wanted to learn about object oriented programming and make use of interfaces which is all good but don't use an interface when it's not needed, keep it simple. I don't think you needed an interface in your example, unless you plan on expanding your project to support more than one scraper "provider". E.g. One for instagram, twitter, facebook etc.

Agreed to some extent. If you want to follow SOLID principles then you should definitely use an interface, even if you only have one single class inherit from it, in case in the future you want to expand your program or re-use code for a different project. This does not apply to all cases though; Haze's statement is still valid for the most part. But for example you have a class that exports to CSV named "CSVExporter", you should make an interface like "IExporter" so in the future you can easily add something like "DocExporter" even though you don't use it now, later, or never will.

+Haze - December 31, 2017, 11:33 am

(Dec 30, 2017 09:50 PM)Ninja Wrote: [\[CITE\]](#) Agreed to some extent. If you want to follow SOLID principles then you should definitely use an interface, even if you only have one single class inherit from it, in case in the future you want to expand your program or re-use code for a different project. This does not apply to all cases though; Haze's statement is still valid for the most part. But for example you have a class that exports to CSV named "CSVExporter", you should make an interface like "IExporter" so in the future you can easily add something like "DocExporter" even though you don't use it now, later, or never will.

I agree to be honest, I had this same discussion with one of the senior devs at my workplace and he said the same thing as you. In the end I explained that I never had to write a "big" project, like the ones we do at work, so I never had any REAL uses for an interface because that's where they'd come in handy. The use interfaces so much here even for small things, so yeah I get it. I tend to only use interfaces when I have a specific goal in mind..

Ninja - December 31, 2017, 5:15 pm

(Dec 31, 2017 12:33 PM)+Haze Wrote: [\[CITE\]](#) I agree to be honest, I had this same discussion with

one of the senior devs at my workplace ad he said the same thing as you. In the end I explained that I never had to write a "big" project, like the ones we do at work, so I never had any REAL uses for an interface because thats where they'd come in handy. The use interfaces so much here even for small things, so yeah I get it. I tend to only use interfaces when I have a specific goal in mind..

Even in small projects it is useful. More info here:

<https://stackoverflow.com/a/1335624>