

Forum: [Hack Forums](#)

Thread: [AV Scanning \(Work in Progress\)](#)

MODERN - February 21, 2018, 3:44 pm

Looking for the right direction to material for AV scanning. Currently been working on my UI

Ideally I wanted to replicate features that JRT has since that is getting discontinued (<https://www.malwarebytes.com/junkwareremovaltool/>)

gigajew - February 21, 2018, 3:50 pm

I like the idea and the icon

EZLib - February 21, 2018, 6:54 pm

Hey this is a cool idea!

I'm kinda doing the same, but in a different way.

<https://github.com/developer-variety/Anti-Logger>

cocaine_monkey - February 22, 2018, 2:26 am

I wonder how does it detect adware? hardcoded database?

chrizator - February 22, 2018, 4:26 am

That looks like an interesting project. UI is nice so far. I wonder if you acutally want to dig into heuristic analysis. Would be very interesting! ?

Zephyros - February 22, 2018, 9:05 am

Dont write a GUI for a desktop app, write a webservice.

Washi - February 22, 2018, 9:38 am

Don't bother with the UI for now. The engine is way more important. Do you have any plans for that already?

Ninja - February 22, 2018, 10:07 am

(Feb 22, 2018 10:38 AM)Washi Wrote: [\[CITE\]](#) Don't bother with the UI for now. The engine is way more important. Do you have any plans for that already?

You should know that on any skid infested 'hacking' forums the UI makes the program 10x better and more expensive, right?

MODERN - February 22, 2018, 10:22 am

This more of a personal project for my own use. I have a thing for UIs so that is why I started there.

The plan is for it to do the following:

- Portable
- Automatic PUP cleanup
- Automatic cleanup of adware (like conduit, coupon, etc.. type shit)
- Get rid of unwanted toolbars
- Check and cleanup browser extensions/addons

- Behavior monitor (like cyber reason's ransomfree)
- Sig Verification (like how process explorer does)
- Heuristic scans (need to figure out database stuff for this)

0xDEAD10CC - February 22, 2018, 2:49 pm

(Feb 22, 2018 10:38 AM)Washi Wrote: [\[CITE\]](#) Don't bother with the UI for now. The engine is way more important. Do you have any plans for that already?

Exactly, a lot of people don't understand that you build a UI around the functionality that you are providing. If you try to use a UI to match functionality that you haven't even built yet it doesn't work that way. I always build my class libraries first and write basic form stuff or console applications to test it out.

So I really don't see much point for this thread quite yet. There's really nothing built yet and this isn't a GDI or WPF theme thread.

Ninja - February 22, 2018, 3:02 pm

(Feb 22, 2018 03:49 PM)0xDEAD10CC Wrote: [\[CITE\]](#) Exactly, a lot of people don't understand that you build a UI around the functionality that you are providing. If you try to use a UI to match functionality that you haven't even built yet it doesn't work that way.

If it is a small application and you wrote down your requirements then it is fine. Especially for a beginner.

(Feb 22, 2018 03:49 PM)0xDEAD10CC Wrote: [\[CITE\]](#) I always build my class libraries first and write basic form stuff or console applications to test it out.

No unit tests? :(

0xDEAD10CC - February 22, 2018, 4:37 pm

(Feb 22, 2018 04:02 PM)Ninja Wrote: [\[CITE\]](#) If it is a small application and you wrote down your requirements then it is fine. Especially for a beginner.

No unit tests? :(

Unit testing is usually for portions of code that needs to fit a certain requirement. I just use a console application from a higher level perspective sometimes to just test out functionality as a whole when I'm not testing code units to see if they are fit for production. Either way, GUI is last priority. Unless he enjoys doing things differently for example, people don't start by building a car from the looks, they usually have the engine engineered first because the design of the car itself needs to accommodate this core piece. This thread seems like a ploy for the coolness factor and perhaps to gain some hype about a product that isn't going to be released anytime soon anyways, and the only thing likely to stay in that GUI are probably exactly what is seen in the image and not much more.

Cecilia - February 22, 2018, 11:58 pm

(Feb 22, 2018 05:37 PM)0xDEAD10CC Wrote: [\[CITE\]](#) Unit testing is usually for portions of code that needs to fit a certain requirement. I just use a console application from a higher level perspective sometimes to just test out functionality as a whole when I'm not testing code units to see if they are fit for production. Either way, GUI is last priority. Unless he enjoys doing things differently for example, people don't start by building a car from the looks, they usually have the engine engineered first because the design of the car itself needs to accommodate this core piece. This thread seems like a ploy for the coolness factor and perhaps to gain some hype about a product that isn't going to be released anytime soon anyways, and the only thing likely to stay in that GUI are probably exactly what is seen in the image and not much more.

I agree with this statement.

I always build the backend as a console app and then either add the ui, or turn the console app into a dll.

Ninja - February 23, 2018, 5:32 am

(Feb 22, 2018 05:37 PM)0xDEAD10CC Wrote: [\[CITE\]](#) Unit testing is usually for portions of code that needs to fit a certain requirement. I just use a console application from a higher level perspective sometimes to just test out functionality as a whole when I'm not testing code units to see if they are fit for production. Either way, GUI is last priority. Unless he enjoys doing things differently for example, people don't start by building a car from the looks, they usually have the engine engineered first because the design of the car itself needs to accommodate this core piece. This thread seems like a ploy for the coolness factor and perhaps to gain some hype about a product that isn't going to be released anytime soon anyways, and the only thing likely to stay in that GUI are probably exactly what is seen in the image and not much more.

Agreed, but for a beginner who learns on his own it is fine. He will understand eventually if he gets more proficient.

(Feb 22, 2018 05:37 PM)0xDEAD10CC Wrote: [\[CITE\]](#) Unit testing is usually for portions of code that needs to fit a certain requirement.

Unit testing is just to test your code. It is a nice way to check it even. I prefer unit testing over a console app any time. With a console app you have to edit the code a bit every time, catch exceptions, etc. With unit tests you write the test and you can select the test(s) you want to run, set up initialization, run the tests, exceptions are caught for you and displayed, and you can assert multiple conditions easily. You can even make a test expect a certain exception to be thrown. It is just easier than a console application, and more convenient.

But I partially agree, for somethings like the database handling and client-server connections you don't write unit tests or you use mock classes/data.

M O D E R N - February 23, 2018, 8:24 am

Everyone is missing that I have said this is for my personal use, nothing is going to be sold...

0xDEAD10CC - February 23, 2018, 8:30 am

(Feb 23, 2018 06:32 AM)Ninja Wrote: [\[CITE\]](#) Agreed, but for a beginner who learns on his own it is fine. He will understand eventually if he gets more proficient.

Unit testing is just to test your code. It is a nice way to check it even. I prefer unit testing over a console app any time. With a console app you have to edit the code a bit every time, catch exceptions, etc. With unit tests you write the test and you can select the test(s) you want to run, set up initialization, run the tests, exceptions are caught for you and displayed, and you can assert multiple conditions easily. You can even make a test expect a certain exception to be thrown. It is just easier than a console application, and more convenient.

But I partially agree, for somethings like the database handling and client-server connections you don't write unit tests or you use mock classes/data.

I know what unit testing is for, but I'm not saying the console app is there for testing, I use it to validate that it works the way I want without having to worry about all that UI code bloating everything.

(Feb 23, 2018 09:24 AM)M O D E R N Wrote: [\[CITE\]](#) Everyone is missing that I have said this is for my personal use, nothing is going to be sold...

Nobody missed that, all the points mentioned here are all still valid whether this is public or private product.

Ninja - February 23, 2018, 8:53 am

(Feb 23, 2018 09:30 AM)0xDEAD10CC Wrote: [\[CITE\]](#) I know what unit testing is for, but I'm not saying the console app is there for testing, I use it to validate that it works the way I want without having to worry about all that UI code bloating everything.

I dont get that. Unit tests are for testing the code. Why a console app? Unit tests are easier and better for testing anyways.

0xDEAD10CC - February 23, 2018, 9:07 am

(Feb 23, 2018 09:53 AM)Ninja Wrote: [\[CITE\]](#) I dont get that. Unit tests are for testing the code. Why a console app? Unit tests are easier and better for testing anyways.

You're missing my point though. Unit testing is for TESTING code yes. I use the console app for easy end-user interaction validation on how well my program functions and that it does what I want in the way that I want it to work; trying the program out, not testing it. Unit testing is not a substitute for my personal perspective. Sure you can write all the unit tests you want and do all the profiling you want, but without having a way to interact with your program to determine if those actions need to be taken, what's the point? Typically you use a program first before you determine whether something needs to be optimized, and unit testing is only as good as the tests you write. If you want unit tests to do everything for you so that you don't even have to look at your result program, then you'd be better off just getting into the AI world and writing a robot to replace your own hand-testing.

If you choose to use a GUI for the trial of your core logic then any feature additions you have to build into the GUI to interact with them, and after many changes this doesn't guarantee that the GUI will be your final one either so it's a waste of time.

Ninja - February 23, 2018, 10:27 am

(Feb 23, 2018 10:07 AM)0xDEAD10CC Wrote: [\[CITE\]](#) You're missing my point though. Unit testing is for TESTING code yes. I use the console app for easy end-user interaction validation on how well my program functions and that it does what I want in the way that I want it to work; trying the program out, not testing it.

I fail to see how 'trying the program out' with a console application is not testing and can't be done with a unit test.

(Feb 23, 2018 10:07 AM)0xDEAD10CC Wrote: [\[CITE\]](#) Sure you can write all the unit tests you want and do all the profiling you want, but without having a way to interact with your program to determine if those actions need to be taken, what's the point?

What do you mean an action needs to be taken? That can be done as well using unit tests and asserts.

You fail to see my point. A console application is just a small window outputting text. The only thing that unit tests does not have is text based output, or output of any kind for that matter. But you don't need that text input/output the console offers. For most things you can use Asserts anyway. Want to see if a certain value is returned? Assert. Want to check for an exception? Use the ExpectedException attribute.

(Feb 23, 2018 10:07 AM)0xDEAD10CC Wrote: [\[CITE\]](#) and unit testing is only as good as the tests you write.

Same can be said for console applications. A console application is as good as the code you write.

0xDEAD10CC - February 23, 2018, 2:14 pm

(Feb 23, 2018 11:27 AM)Ninja Wrote: [\[CITE\]](#) I fail to see how 'trying the program out' with a console application is not testing and can't be done with a unit test.

What do you mean an action needs to be taken? That can be done as well using unit tests and asserts.

You fail to see my point. A console application is just a small window outputting text. The only thing that unit tests does not have is text based output, or output of any kind for that matter. But you don't need that text input/output the console offers. For most things you can use Asserts anyway. Want to see if a certain value is returned? Assert. Want to check for an exception? Use the `ExpectedException` attribute.

Same can be said for console applications. A console application is as good as the code you write. "I fail to see how 'trying the program out' with a console application is not testing and can't be done with a unit test." -- Well just for starters, a unit test doesn't care about any irrelevant output that a normal program might output. If I'm running a console application for a project I'm writing too, personally I'm not testing anything, I'm reviewing functionality and building program flow without things being too tightly coupled with UI elements from my core logic and a console application makes this really easy to do because there IS NO UI; great SOLID principles. This kind of review, even if unintentional, can help you write better unit tests too. Until you actually use your program some unit tests that you SHOULD have might be considered more of a black box, and without a UI you'll be able to more easily write unit tests for problems that might occur only from core program logic and nothing dealing with UI issues.

Would you suggest someone using a GUI application is always "testing" it? ? I don't see why you have your mind set on that. Same principle, just no GUI. A console application structure will probably help you with designing a proper UI as well once you've reached that point: Unit tests don't do that.

As a bonus, if you wish - you can provide a console-based release along with a public release that comes packaged with a UI because by the time you've completed your program you'll already have a console-based version.

"Same can be said for console applications. A console application is as good as the code you write." - But the application and the unit tests are NOT serving the same purpose here at least for what I intend them to be used for, so this is an irrelevant comparison. Unit testing does not aid in helping you structure your program, it's for testing only.

Unit tests don't help you port program logic over to a GUI version. And a console application would make this much easier to do because the program logic is pretty much mostly complete at that point, all you need to focus on is the UI part of the code, and maybe some data binding. Think of it as a practice that forces you to consider SOLID, and this is nice if you're not using WPF because you can avoid things from being too tightly coupled--which is something a lot of poor WinForms project are guilty of.

Does that finally clarify things for you?

Friedrich Nietzsche - February 23, 2018, 2:20 pm

Why do this in C#? This is a terrible idea, most of the things you listed can't be done in .NET/is very inefficient to do in .NET. Behaviour monitoring requires native code for placing the hook and parsing the API calls, scanning and heuristic engines would be much faster if coded in a native language,

etc

Don't try making an AV when you are not familiar with native programming and windows internal. It won't end well. Learn C first for things like this.

Ninja - February 23, 2018, 6:10 pm

(Feb 23, 2018 03:14 PM)0xDEAD10CC Wrote: [\[CITE\]](#) "I fail to see how 'trying the program out' with a console application is not testing and can't be done with a unit test." -- Well just for starters, a unit test doesn't care about any irrelevant output that a normal program might output. If I'm running a console application for a project I'm writing too, personally I'm not testing anything, I'm reviewing functionality and building program flow without things being too tightly coupled with UI elements from my core logic and a console application makes this really easy to do because there IS NO UI; great SOLID principles. This kind of review, even if unintentional, can help you write better unit tests too. Until you actually use your program some unit tests that you SHOULD have might be considered more of a black box, and without a UI you'll be able to more easily write unit tests for problems that might occur only from core program logic and nothing dealing with UI issues.

Would you suggest someone using a GUI application is always "testing" it? ? I don't see why you have your mind set on that. Same principle, just no GUI. A console application structure will probably help you with designing a proper UI as well once you've reached that point: Unit tests don't do that.

As a bonus, if you wish - you can provide a console-based release along with a public release that comes packaged with a UI because by the time you've completed your program you'll already have a console-based version.

"Same can be said for console applications. A console application is as good as the code you write." - But the application and the unit tests are NOT serving the same purpose here at least for what I intend them to be used for, so this is an irrelevant comparison. Unit testing does not aid in helping you structure your program, it's for testing only.

Unit tests don't help you port program logic over to a GUI version. And a console application would make this much easier to do because the program logic is pretty much mostly complete at that point, all you need to focus on is the UI part of the code, and maybe some data binding. Think of it as a practice that forces you to consider SOLID, and this is nice if you're not using WPF because you can avoid things from being too tightly coupled--which is something a lot of poor WinForms project are guilty of.

Does that finally clarify things for you?

I still find your point vague. Why do you say it is not for testing? You want to see if the code works, right? Then use Unit Tests. Only if you are printing (debug) information then you need a console.

Everything you test in a console application can be done in unit tests; Plus; if a function requires 10 steps to get to, you can just write a unit test for it and you don't need to complete all those steps. The only advantage a console app has is that you can output information way easier.

(Feb 23, 2018 03:14 PM)0xDEAD10CC Wrote: [\[CITE\]](#) Unit tests don't help you port program logic over to a GUI version.

I fail to see how. If you abstracted your code well then this does not apply at all. You can put the same code (minus asserts) that you have in a unit tests in a event handler, or a function, or whatever. The 'hassle' of 'porting' unit test code to a winforms application for example is the same as 'porting' from a console application to a winforms application.

(Feb 23, 2018 03:14 PM)0xDEAD10CC Wrote: [\[CITE\]](#) all you need to focus on is the UI part of the code, and maybe some data binding.
With unit test as well.

Cecilia - February 23, 2018, 7:33 pm

I too enjoy discussing unit tests on a page about AV scanning.
Perhaps the discussion could move to another thread titled "[Debate] Unit Tests vs. Console Apps"?

cocaine_monkey - February 23, 2018, 7:37 pm

(Feb 23, 2018 07:10 PM)Ninja Wrote: [\[CITE\]](#) I still find your point vague. Why do you say it is not for testing? You want to see if the code works, right? Then use Unit Tests. Only if you are printing (debug) information then you need a console.

Everything you test in a console application can be done in unit tests; Plus; if a function requires 10 steps to get to, you can just write a unit test for it and you dont need to complete all those steps. The only advantage a console app has is that you can output information way easier.

I fail to see how. If you abstracted your code well then this does not apply at all. You can put the same code (minus asserts) that you have in a unit tests in a event handler, or a function, or whatever. The 'hassle' of 'porting' unit test code to a winforms application for example is the same as 'porting' from a console application to a winforms application.

With unit test as well.
You can print debug info in unit test also

Vyxn - February 24, 2018, 1:46 am

(Feb 23, 2018 11:39 PM)M O D E R N Wrote: [\[CITE\]](#) Thank you for the first legitimately helpful reply.

I am actually going through a c++ course through Udemy (warez disallowed) (warez disallowed) atm.

Don't you just love it? When two knowledgeable programmer arguing their point on your thread?

OT: Are you really making av or just a tools like adwcleaner because their different thing. Program like this can be achievable if you know how to:identify the behaviour of your target.

know where your target is hiding.

There's also a malware scanner here made in .net, you may consult him if you want: [TazzyOpz](#)

0xDEAD10CC - February 24, 2018, 9:30 am

(Feb 23, 2018 07:10 PM)Ninja Wrote: [\[CITE\]](#) I still find your point vague. Why do you say it is not for testing? You want to see if the code works, right? Then use Unit Tests. Only if you are printing (debug) information then you need a console.

Everything you test in a console application can be done in unit tests; Plus; if a function requires 10 steps to get to, you can just write a unit test for it and you dont need to complete all those steps. The only advantage a console app has is that you can output information way easier.

I fail to see how. If you abstracted your code well then this does not apply at all. You can put the same code (minus asserts) that you have in a unit tests in a event handler, or a function, or whatever. The 'hassle' of 'porting' unit test code to a winforms application for example is the same as 'porting' from a console application to a winforms application.

With unit test as well.

"With unit test as well." - but you can't build a program with unit tests is my point. You can write tests to test your program with unit tests. Console app and unit tests are not the same thing here for why I'm suggesting a console app, why are you even making this comparison? That's why I'm saying I'm not suggesting the console app for testing.

(Feb 23, 2018 08:33 PM)Cecilia Wrote: [\[CITE\]](#) I too enjoy discussing unit tests on a page about AV scanning.

Perhaps the discussion could move to another thread titled "[Debate] Unit Tests vs. Console Apps"?

"Unit Tests vs. Console Apps" - this alone should say how silly it is for anybody to even try to make this comparison... Not sure why he's suggesting it is an alternative solution here. I'm talking about building program logic into a console application and the ability of having it portable to a UI version later. I thought I made that clear a few times throughout this thread ? ?

One more time - I'm not suggesting to use console apps for "testing". I'm talking about using it as an asset to build the program into a UI later.

He's arguing a point that I'm not debating.

@OP - What else do you truly expect to gain from this thread? There's no product, only an image of some graphical interface that you have. You may want to look into YARA if you know nothing of it yet: <https://virustotal.github.io/yara/>

You're wanting static analysis only or dynamic as well?

Cecilia - February 24, 2018, 8:45 pm

(Feb 24, 2018 10:30 AM)0xDEAD10CC Wrote: [\[CITE\]](#) @OP - What else do you truly expect to gain from this thread? There's no product, only an image of some graphical interface that you have. You may want to look into YARA if you know nothing of it yet: <https://virustotal.github.io/yara/>

You're wanting static analysis only or dynamic as well?

YARA is most certainly something that could be used to store sigs for malware.

The YARA scanner is also very fast and efficient (and includes a lot of options).

Could be a simple solution.

As you said, he probably wants dynamic analysis (runtime analysis) or similar.

I think he'll find the course on C++ quite helpful (although he will need to read up on winapi and windows internals).