

Forum:[Hack Forums](#)

Thread: [Notification - A quick winforms toast notification class](#)

uint32 - December 27, 2016, 1:50 am

So basically I made this class for one of my projects and since I never contributed to this section, I thought this would be a nice little 'gift' for the holiday season.

Notification ([Click to View](#))

Gif ([Click to View](#))

I tried my best to make it easily understandable by adding as many comments as I can. If anything stands unclear just ask away on this thread.

[You can find it on my Github](#)

Class is Notification.cs if you get lost

You can set:Image

Title

Display Text

Sound to play

Border color

and time to stay open

I recommend using 64x64 images for the best result but anything else can do the job fine.

Any suggestions are appreciated. If you wanna test it out open TestForm and use some of the included images and sounds.

Baseplate - December 27, 2016, 2:24 am

That's cool!

I'm assuming that you could launch a timer when the Load event fires, to make the form slowly fade in by gradually changing the Form.Opacity property?

(Because the NodeNotification class inherits the Form class.)

ubbelol - December 27, 2016, 7:35 am

Notification looks pretty good and seems simple enough to use. The code contains a lot of redundant comments IMHO, though. :P

Ninja - December 27, 2016, 8:38 am

Is it truly necessary to make it a form? Cant you just draw graphics on screen?

Serious question, I dont know much about this stuff.

Instead of:

Code if (`_timer` != null)

`_timer.Start()`;You can just do:

Code `_timer?.Start()`;The ? is a null coalesce operator (not sure if I wrote that correctly). It doesn't do anything performance wise I dont think but I hate writing a ton of null checks personally.

Also, not an expert in this and it probably doesn't matter, but if you call Show a few times wont that

mess things up with the timer? And why not a partial class for all the designer stuff?

uint32 - December 27, 2016, 9:52 am

(Dec 27, 2016 03:24 AM)Baseplate Wrote: [\[CITE\]](#) That's cool!

I'm assuming that you could launch a timer when the Load event fires, to make the form slowly fade in by gradually changing the Form.Opacity property?

(Because the NodeNotification class inherits the Form class.)You're totally right, animations completely flew over me. I'll make sure to add your idea once I get time.

(Dec 27, 2016 08:35 AM)ubbelol Wrote: [\[CITE\]](#) Notification looks pretty good and seems simple enough to use. The code contains a lot of redundant comments IMHO, though. :PTrue, It was 4am when I was finishing up the comments haha.

(Dec 27, 2016 09:18 AM)Orka Wrote: [\[CITE\]](#) I like it's flexible :) I suggest to remove the overloaded constructors since these are setting public properties.

```
Code public NodeNotification()
{
    TopMost = true;
    DoubleBuffered = true;

    Resize += NodeNotification_Resized;
    GotFocus += NodeNotification_GotFocus;
}
```

Or make them private and keep the constructors.I agree the constructors might look somewhat messy, though they don't impact the performance and it's just the way I like it :P

(Dec 27, 2016 09:38 AM)Ninja Wrote: [\[CITE\]](#) Is it truly necessary to make it a form? Cant you just draw graphics on screen?

Serious question, I dont know much about this stuff.

Instead of:

Code if (_timer != null)
 _timer.Start();You can just do:

Code _timer?.Start();The ? is a null coalesce operator (not sure if I wrote that correctly). It doesn't do anything performance wise I dont think but I hate writing a ton of null checks personally.

Also, not an expert in this and it probably doesn't matter, but if you call Show a few times won't that mess things up with the timer? And why not a partial class for all the designer stuff?1. Yeah it's kinda necessary because it's the only parent container with the TopMost property and without it wouldn't work.

2. Drawing the graphics on the screen is annoying for nothing honestly, the form suits the role perfectly.

3. I'm aware of the null conditional operator. In some cases (like you mentioned) I use the old way since it's easy to see and with the null operator I tend to skip it when I read over my code.

4. Calling the show method multiple times would screw things up yes, but normally you shouldn't call show multiple times on the same object. I'll add some flags to fix that.

5. Didn't choose a partial class because I wanted the class to be standalone and the designer can't render a form without the *.Designer.cs and I placed that code in the NodeNotificationClass so I can easily move it to other project

Thanks for all the comments!

passwords - December 27, 2016, 11:31 am

Nice contribution, feels like it could of been made simpler but anyways! Good work! :)

uint32 - December 27, 2016, 1:10 pm

(Dec 27, 2016 12:06 PM)Orka Wrote: [\[CITE\]](#) Constructor args is to set one time instance variables you cannot afterwards access, this is what I mean in my previous post.

Well it sets the properties sooooo

BahNahNah - December 27, 2016, 3:27 pm

(Dec 27, 2016 10:52 AM)Coefficient Wrote: [\[CITE\]](#) 3. I'm aware of the null conditional operator. In some cases (like you mentioned) I use the old way since it's easy to see and with the null operator I tend to skip it when I read over my code.

There is a bonus of using the "?." operator, that it is threadsafe.

The equivalent code would be:

Code var t = _timer;

if(t != null)

t.Start();

However it dosent look like it matters in this situation, because i cant see anywhere the timer being set to null after being initialized.

-Edit:

And your properties dont actually update the form when they are set. You could do something like this:

Spoiler [\(Click to View\)](#)

Malum™ - December 27, 2016, 3:46 pm

This is really cool! I'll take a look at the code when I get the chance. Might use it in a project or two :). I don't think you made it clear but does it stack if I do multiple notifications?

uint32 - December 27, 2016, 6:13 pm

(Dec 27, 2016 04:27 PM)BahNahNah Wrote: [\[CITE\]](#) There is a bonus of using the "?." operator, that it is threadsafe.

The equivalent code would be:

Code var t = _timer;

if(t != null)

t.Start();

However it dosent look like it matters in this situation, because i cant see anywhere the timer being set to null after being initialized.

-Edit:

And your properties dont actually update the form when they are set. You could do something like this:

Spoiler [\(Click to View\)](#)

Yea the properties don't update, I could add a setter to the properties but I didn't need it. I'll make sure include that in the update. Thanks!

(Dec 27, 2016 04:46 PM)Malum™ Wrote: [\[CITE\]](#) This is really cool! I'll take a look at the code when I get the chance. Might use it in a project or two :). I don't think you made it clear but does it stack if I do multiple notifications?

No, it doesn't stack. You could make yourself a class to handle that, though. Make a list of open forms and every time a new one is opened move the other ones up.

Malum™ - December 27, 2016, 6:19 pm

(Dec 27, 2016 07:13 PM)Coefficient Wrote: [\[CITE\]](#) No, it doesn't stack. You could make yourself a class to handle that, though. Make a list of open forms and every time a new one is opened move the other ones up.

Why not you? You're the developer. Haha. It'd be something to add that would make more people use it.

Zenco - January 2, 2017, 12:32 am

looks pretty good, thanks for the share.

BoliBerrys - January 2, 2017, 5:37 pm

Damn this is a really cool thing to add on my tools.

Thanks alot !