

Forum:[Hack Forums](#)

Thread: [Multithreading in a CLR managed project?](#)

Goodies - January 24, 2016, 5:59 pm

I am writing more or less by first actual application in visual C++. I'm new to this.

I have a windows CLR application that spawns a simple form with a progress bar. The progress bar should represent some type of file processing. For now, let's just say that it increments from 1 to 100 in 5 seconds. I cannot use a timer for the actual file processing, obviously, because I don't know how long it will take.

I had it seemingly working, as in the file does get processed, but it does not update the progress bar. `<thread>` cannot be used in CLR applications.

How can I update this and/or do multithreading in a managed class? I'm stuck.

This is the kind of shit I see:

`progress()` is called upon `Form::Shown` event.

0xDEAD10CC - January 24, 2016, 8:34 pm

Show your full code, are you instantiating the newly created form entirely on it's own thread? You'll need to invoke to join the separate thread with the new form thread in order to update the progressbar visual anyways you know if you synchronize this to some file I/O.. You're hardly showing anything but designer generated code here and how you increment the progressbar's value property. Nothing as to how you create the new thread or any of that. This isn't useful.

Why managed C++?

Goodies - January 24, 2016, 10:03 pm

Quote:Why managed C++?

I'm in visual studio 2015 and there doesn't appear to be a normal Forms application :| It's not my choice. I don't even really get its purpose, but that can be googled.

I don't know if the way I've done it is what you'd call "acceptable" or not, but I did it without threading and the progress bar does not hang or stall the application (like cause the rotating wheel and whatnot), so it appears to be working exactly as intended. Any tips/criticisms are more than welcome, but try not to be overly abrasive because it's not helpful to either of us.

Basically, it's a simple application that securely deletes a provided file. I ended up verifying the files in the constructor:

PHP Code

```
<?php
```

```
Vaniche::Vaniche form(args[0], paranoia); // "paranoia" is an int
```

PHP Code

```
<?php
```

```

Vaniche(String ^fname, int p){
    this->filename = fname;
    this->paranoia = p;
    convertFileName();
// stores the filename managed string as a C string in "cfilename" private variable.
    verifyFile(); // calls ExitProcess(0) if failed
    InitializeComponent();
}

```

The Form::Shown event starts a timer. The timer immediately stops itself and performs the actual operations.

PHP Code

```

<?php
private: System::Void timer1_Tick(System::Object^ sender, System::EventArgs^ e) {
    timer1->Stop();
    FILE *f = _wfopen(cfilename, L"rb+");
    _fseeki64(f, 0, SEEK_END);
    long long fileSize = _ftelli64(f);
    long long currentLocation = 0;
    long long totalData = fileSize * paranoia;
    long long processed = 0;
    int paranoiaLevel = 0;
    int i;
    this->progErase->Maximum = PROG_MAX;
// PROG_MAX = 10,000 but it's not relevant. higher numbers make the progress bar smoother.
    std::random_device rdev;
    unsigned char buf[1];

    this->lblStatus->Text = L"Overwriting with Random Data...";
    Application::DoEvents();
    for (paranoiaLevel = 0; paranoiaLevel < paranoia - 1; ++paranoiaLevel) {
        fseek(f, 0, SEEK_SET);
        for (currentLocation = 0; currentLocation < fileSize; ++currentLocation) {
            buf[0] = rdev() % 0x100;
            fwrite(buf, 1, 1, f);
            processed += 1;
            if (((int)PROG_MAX * processed / totalData) > this->progErase->Value) {
                this->progErase->Value = (int)(PROG_MAX * processed / totalData);
                Application::DoEvents();
            }
        }
    }
    this->lblStatus->Text = L"Overwriting with Null Bytes...";
    Application::DoEvents();
    buf[0] = '\x00';
    fseek(f, 0, SEEK_SET);
    for (currentLocation = 0; currentLocation < fileSize; ++currentLocation) {
        fwrite(buf, 1, 1, f);
        processed += 1;
        if (((int)PROG_MAX * processed / totalData) > this->progErase->Value) {
            this->progErase->Value = (int)(PROG_MAX * processed / totalData);
            Application::DoEvents();
        }
    }
}

```

```

}
fclose(f);
WCHAR newFileName[FILENAME_SIZE + 1], oldFileName[FILENAME_SIZE + 1];
StrCpyW(newFileName, cfilename);
for (i = 0; i < (paranoia * 10); ++i) {
    StrCpyW(oldFileName, newFileName);
    randomFileName(newFileName, FILENAME_SIZE);
    _wrename(oldFileName, newFileName);
}
_wremove(newFileName);
//_wremove(oldFileName);
//ExitProcess(0);
}

```

Here is the full solution. Judge harshly, but not too mean-ly, I'm not a programmer.

<https://mega.nz/#!2hVC3QiK!VVyqaQDHmdrL6PdeAkIvyWS2b02kOwYEQbZ-XEnLu9k>

A TODO would be to create a way, be it a button or key press or what have you, to have the user stop the action entirely.

0xDEAD10CC - January 24, 2016, 10:30 pm

What do you mean "Normal forms application" what is "normal"? :S

Also, if Form::Shown() starts the timer and the timer stops itself on the first tick, why do you even have a timer? Why not just call a separate method because it only fires once?

What is the goal of your code?

Goodies - January 25, 2016, 2:05 am

(Jan 24, 2016 11:30 PM)0xDEAD10CC Wrote: [\[CITE\]](#) What do you mean "Normal forms application" what is "normal"? :SI may not have the correct idea of what managed and unmanaged means, but I guess "normal" means "the old Windows Forms Application" I remember from visual studio 2010. I don't remember using managed code for that.

Quote:Also, if Form::Shown() starts the timer and the timer stops itself on the first tick, why do you even have a timer? Why not just call a separate method because it only fires once?

That was adapted from earlier version where the timer did something else. Took that out.

Quote:What is the goal of your code?

I'm going to have to learn how to make an installer package, but it's basically just a secure file eraser... nothing special. It will overwrite a file with random data N-1 number of times and the Nth time it will be overwritten with \x00's and then unlinked.

0xDEAD10CC - January 26, 2016, 2:42 pm

Are you talking about MFC? I still don't know what you're referring to.

RISC - January 26, 2016, 3:10 pm

(Jan 26, 2016 03:42 PM)0xDEAD10CC Wrote: [\[CITE\]](#) Are you talking about MFC? I still don't know what you're referring to.

Managed C++/C++ CLR/Visual C++.

0xDEAD10CC - January 26, 2016, 3:52 pm

(Jan 26, 2016 04:10 PM)RISC Wrote: [\[CITE\]](#) Managed C++/C++ CLR/Visual C++.

He said:

Quote:but I guess "normal" means "the old Windows Forms Application" I remember from visual studio 2010. I don't remember using managed code for that.

RISC - January 26, 2016, 5:18 pm

(Jan 26, 2016 04:52 PM)0xDEAD10CC Wrote: [\[CITE\]](#) He said:

He must be referring to Resources then.

Goodies - January 26, 2016, 5:31 pm

Guys you're giving me too much credit. I haven't even used windows since around 2011/2012 or so... I honestly just don't know.

RISC - January 26, 2016, 8:19 pm

I wrote this up a long time ago.

<https://hackforums.net/showthread.php?tid=286074>

It uses resources to build a form application. You can do it without any CLR.

Only3 - January 27, 2016, 5:43 am

choose win32 console application project

RISC - January 27, 2016, 6:07 am

(Jan 27, 2016 06:43 AM)Only3 Wrote: [\[CITE\]](#) choose win32 console application project

I doubt he wants to get into Direct 3D or GDI with little knowledge of Windows.

Goodies - January 27, 2016, 7:27 am

(Jan 27, 2016 07:07 AM)RISC Wrote: [\[CITE\]](#) I doubt he wants to get into Direct 3D or GDI with little knowledge of Windows.

This is accurate. I'm not particularly interested in that tbh. Visual programming is okay, but I would rather look for functionality. It just happened that this needed a small amount of it.