Is it worth porting a small C# "engine" to C++ or similar?

Asked 16 years, 1 month ago Modified 13 years, 11 months ago

Viewed 667 times

4

So, following a tutorial I found on MSDN I've created what you might call an "engine" using DirectX and C#. I haven't seen a lot of this sort of thing (Personally, that is) done in C# and the majority seem to favour C/C++ so I'm curious as to whether using C# will come back to bite me, or whether I should just go ahead?

The reason I ask is that the tutorial stopped rather suddenly and so there's a basis of something but it's small enough to be ported without much hassle. I like C# myself, but I don't know whether there's something everyone else knows that I don't.

c# c++ directx

Share

Improve this question

Follow

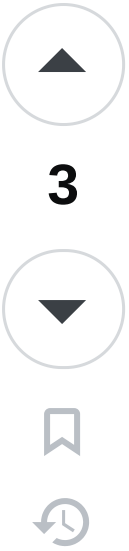
asked Nov 14, 2008 at 3:31

Peter C.

433 ● 2 ● 6 ● 12

5 Answers

Sorted by: Highest score (default)



Well Peter, the main reason is C++ is platform agnostic, while C# isn't in the true sense, despite Mono (I don't think you can run Mono C# on most cellphones, the Nintendo DS or the Playstation3, for example).

Anyways most of these engines support DirectX and OpenGL, and/or software renderers, thus they are engineered from the ground up to be agnostic. But as with any agnostic system, this requires a lot of effort, and resources.

Now from what you say, your engine is C# + DirectX, meaning your engine right now may run on PC, Xbox, Zune, and some Window's powered cell-phones. This isn't a bad range of platforms to be honest. So if your happy with this degree of freedom, and you prefer C# over C++ than you might as well stick with what you have, and spend the time improving your engine. However if you want to run your engine on anything not Microsoft, you'll need to make the jump to C/C++ sometime.

For an idea of what I'm talking about my team's engine supports Xbox,PS3,Wii,PSP,PC, and possibly other platforms as-well if we spend the effort, however we have 15 full-time engineers working on it, so... you see it's a big cost. Now if you want something like this for a free project, and you know C++ you may try any of the many open source engines out there such as Ogre or Irrlicht.

Follow

answered Nov 14, 2008 at 4:36

Robert Gould

69.7k ● 61 ● 191 ● 275

1

I suggest staying with what you are familiar with. Use of C or C++ should only be considered in your situation if you intend to do some lower level functionality, like implement your own image processing or create custom filters or transforms (depending on what you are using directx for).

Graphic applications can have computationally intensive aspects. Most of these have been addressed in the DirectX api- which is just as fast if you use it from C++ or from C#.

Where you would consider lower level language: If you are creating some image processing which could perform quicker natively- this only applies if it is not offered by the directx api, such as creating some custom encoders/decoders. In this case you could create the custom component in another language- and then load it in your C# application separately. If you are- look into using C++ Intrinsics, which allow you to program for your CPU, and also take a look at GPGPU API's like CUDA.

Share Improve this answer

Follow

answered Nov 14, 2008 at 3:52

Klathzazt

2,454 ● 19 ● 27

1

As a hobbyist game developer I greatly prefer C# (with a 3rd-party library call SlimDX) over C++. Yes, a game written in well-optimized C++ will probably run better than the same game written in well-optimized C#. But the time I save by using C# (I'm guessing very roughly 30-40%) can be used to optimize my algorithms such that a C# game will actually run faster than had I written it in C++ in the same amount of time.

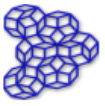
As pointed out above, there's also the cross-platform compatibility issues because for most practical purposes a C# game will only run on Microsoft platforms.

Another issue to consider is that if you write your application in C# then it will require the .Net Framework to be installed on the computers it runs on. This can make distributing your application cleanly somewhat more of a hassle, though this is by no means an insurmountable obstacle.

Share Improve this answer

Follow

edited Jan 10, 2011 at 17:52



Dennis Williamson

360k ● 95 ● 381 ● 442

answered Dec 6, 2008 at 23:50

Walt D

4,681 ● 8 ● 36 ● 46

There are a few studios using c# for games.

0

You can also have a look at XNA, there is also a port of quake3 engine flying around the internet ported to c#.

So it should be ok.

Share Improve this answer

Follow

answered Nov 14, 2008 at 3:52

Mischa Kroon

1,772 ● 1 ● 13 ● 19

0

If you feel more comfortable in C# and there are no performance issues forcing to switch to a language closer to the CPU, then I see no need to change.

However, depending on what you plan to do in the future and how far this engine should be taken, that might change dramatically and C# might turn out to be the wrong choice. But this only time can tell, unfortunately. The right educated guess requires much more detail about the current implementation and about your future plans.

Share Improve this answer

Follow

answered Nov 14, 2008 at 3:53

HS.

2,615 ● 2 ● 21 ● 29