# Why are only a few video games written in Java? [closed]

Asked 15 years, 6 months ago Modified 9 years ago Viewed 226k times

175

As it currently stands, this question is not a good fit for our Q&A format. We expect answers to be supported by facts, references, or expertise, but this question will likely solicit debate, arguments, polling, or extended discussion. If you feel that this question can be improved and possibly reopened, visit the help center for guidance.

Closed 12 years ago.

Why aren't many commercial, 3D video games (not random open source 2D ones) written in Java? In theory, it makes a lot of sense: you get a productivity boost and a cross-platform application almost for free, among other things, such as the vast amount of Java libraries, and built-in garbage collection (although I admit I'm not sure if the latter is a good thing). So why is it rarely used? I can only think of a couple popular commercial games written for the Java platform.

Is it because of performance? If so, wouldn't most of the heavy lifting be done by the GPU anyway?

java

Share

Improve this question

Follow

edited Dec 17, 2015 at 17:00

asked Jun 23, 2009 at 18:55

Sasha Chedygov

131k ● 26 ● 105 ● 116

11 en.wikipedia.org/wiki/Tribal_trouble – Michael Myers ♦ Jun 23, 2009 at 18:56

2 Re: mmyers; I'm somewhat in shock that THAT game won a "best graphics" award, even in 2005... – Cloudy Jun 23, 2009 at 18:59

3 Yeah but most "real games" aren't made in managed .net right? They're made in old school c/c++? – Hardwareguy Jun 23, 2009 at 19:19

15 Runescape is written in java. – GameFreak Jun 23, 2009 at 19:33

47 Minecraft is written in Java! – daGrevis Apr 28, 2011 at 12:18

## 22 Answers

Sorted by: Highest score (default)

162

The game development world is a funny one: On one hand, they're often quick to accept new ideas, on the other hand, they're still in the stone age.

The truth is, there's rarely that much incentive in switching to .NET/Java/anything other than C/C++.

Most game companies license parts of the game engine from other companies. These parts are written in C++, and although you might have access to the source so you could port it, that takes a lot of effort (and of course, the license needs to allow it).

Also, a lot of legacy code already exists in C++. If code from previous projects can be reused (say, if you're writing a sequel), that counts even more in favor of sticking with the same language, instead of rewriting it in a new language (more so since you'll likely reintroduce a ton of bugs which you'll need to spend time ironing out.

Finally, it's rare for games to be written in 100% C++ anyway - a lot is done using scripting languages, whether they're custom or just integrating an existing languages (Lua being one of the more popular ones these days).

As far as garbage collection is concerned, that can be a bit of a problem. The problem is not so much that it exists, it's more how it works - the garbage collector MUST be non-blocking (or at least be guaranteed to only block very briefly), since it's simply unacceptable to have the game freeze for 10 seconds while it scans all the allocated memory to see what can be freed. I know Java tends to choke quite a bit in GC'ing when it's close to running out of memory (and for some games out there, it will).

You're also a bit more restricted in what you can do: you can't fully exploit the hardware due to the overhead of the runtime. Imagine Crysis being written in Java... even if that's the only visible difference, it just wouldn't be the same (I'm also pretty sure you'd need a Core i7 to run it.).

This doesn't mean these languages don't have their place in game development - and no, I'm not just referring to tool programming. For most games, you don't need that extra bit of performance you get from C++, including 3D games, and if you're writing it all from scratch, it can make perfect sense to use something like XNA - in fact, there's a good chance it will.

As far as commercial games are concerned - does RuneScape count? That may well be the most succesful Java game out there.

Share Improve this answer Follow

answered Jun 23, 2009 at 19:37

Michael Madsen
55k ● 8 ● 74 ● 83

16 Well obviously you wouldn't run Crysis on the JVM; hell, if you coded that game in assembly language you'd still need a supercomputer to run it on full settings. But +1 for the excellent insight, thank you. – Sasha Chedygov Jun 23, 2009 at 19:42

16 You can't compare Unreal Tournament 3 or Crysis with Runescape. If graphic quality is a concern, you need to stick with a low level language with as little overhead as possible. Of course, for Indy or games where graphics are not the main selling point, Java is an excellent alternative to C/C++. – GuiSim Jun 23, 2009 at 20:07

6 @GuiSim: For most games, graphics quality is NOT a major selling point. There are only a handful of games I can think of that were created with graphics in mind (I'm thinking of Crysis, but Half-Life 2 as well, at the time). I don't think most game developers care that much about graphics, as long as they are "good enough" (aka on par with most other games). – Sasha Chedygov Jun 23, 2009 at 20:54

4 Graphics really have very little to do with the language. Physics, AI, yes. Graphics, no. – JulianR Jun 23, 2009 at 21:01

10 @JulianR there can be a significant workload for preparing and maintaining a scene to be rendered efficiently, so the language and associated language overhead does matter for graphics. – KSchmidt Dec 8, 2009 at 17:24

95

I think John Carmack said it best with:

> The biggest problem is that Java is really slow. On a pure cpu / memory / display / communications level, most modern cell phones should be considerably better gaming platforms than a Game Boy Advanced. With Java, on most phones you are left with about the CPU power of an original 4.77 mhz IBM PC, and lousy control over everything. [...snip...] Write-once-run-

> anywhere. Ha. Hahahahaha. We are only testing on four platforms right now, and not a single pair has the exact same quirks. All the commercial games are tweaked and compiled individually for each (often 100+) platform. Portability is not a justification for the awful performance.

(source)

Granted, he was talking about mobile platforms, but I've found similar problems with Java as a whole coming from a C++ background. I miss being able to allocate memory on the Stack/Heap on my own terms.

Share Improve this answer Follow

edited Jun 23, 2009 at 19:22
Sasha Chedygov
131k ● 26 ● 105 ● 116

answered Jun 23, 2009 at 19:10
Marc
2,603 ● 2 ● 19 ● 21

65 That quote was from 2005. Both Java technology and Cellphone power has considerably improved since then. Comparing cellphone gaming vs PC gaming is comparing apples to oranges. – Chris Dail Jun 23, 2009 at 19:31

80 John Carmack said it. Case closed. – GuiSim Jun 23, 2009 at 19:31

42 I just get uneasy when I read "Java is really slow". It's like saying a $50k sports car is slow compared to a $100k sports car. Sure, it's slower, but 90% of the time, the job it does is still great and at half the cost ;) No flame war intended. I

agree that the above reasons are why Crysis and like-games are not written in Java. – Ross Jun 23, 2009 at 20:16

17 @Chris Dail, this underscores the whole issue with Java performance. Has Java performance improved? No, cell phones just got faster. Games are supposed to push the limits of realism, and therefore push the limits of hardware, and throwing away %30-%40 of your performance before you've even written a line of code is unacceptable. – cgp Jun 24, 2009 at 17:57

10 I find this dispute very strange. Java ME is not the same as Java in Android and also not the same as Java on PCs. Java ME usually relied on phone manufacturers to come up with a JVM. Some did good job, some didn't. No wonder Carmack was complaining about them. Android has it's own VM which is not a JVM. And it has some serious issues (from my point of view). Oracle's HotSpot VM is completely different from both cases. If people compare all these things, the only thing I can conclude is that they don't know what they are talking about. – Malcolm Jul 17, 2013 at 19:29

55

For one thing, Java's lack of operator overloading makes all of the math you have to deal with to get a working graphics pipeline very, very annoying and hard to read.

All of the matrix multiplication and affine vectors you need to deal with are a lot easier to follow if they're in well-formed mathematical expressions rather than object-oriented expressions like

```
product = vector.multiply(projectionMatrix).dotProduct
```

That's just terrible. Math shouldn't look like that.

Share Improve this answer Follow

answered Jun 23, 2009 at 18:59

Welbog
60.3k ● 9 ● 113 ● 124

---

19 I remember back in '96 I think it was, some of the designers from Sun were giving a presentation on Java at Berkeley. William Kahan (en.wikipedia.org/wiki/William_Kahan) was giving them sh*t over this very issue. :) – JP Alioto Jun 23, 2009 at 19:09

13 i think there is a good reason not to allow operator overloading in a language: to prevent people from using it. it is a powerful tool and very cool for maths, but is dangerous for everything else. lazy as coders are, they tend to missuse it for shortening down code, and the moment people start performing a map multiplying an iterable with a function, or even when all arithmetic ops are defined for functions, code readability is about to reach 0. and yes, i've spent considerable amounts of time porting code like that. :-S it's a design choice. and design choices always tend to be disputable. – back2dos Oct 19, 2009 at 10:39

19 Punish everyone else for a few bad apples? This is one reason I prefer C#. If I really need operator overloading it is there. – ChaosPandion Dec 8, 2009 at 17:22

1 Basically, operator overloading is only really appropriate for 2-3 different situations in OOP design (Vectors, Matrices, Complex numbers). Most other situations, it's too loosely defined and only leads to sloppy code, weak syntax, and poor documentation, even from people who know how to use it. I think that's why Sun opted to not use it in Java, and I think that's a valid decision. – bgroenks Mar 16, 2013 at 16:20

2 "That's just terrible." Is just an oppinion, yours and probably millions of peoples but not everyone hate OO expressions, I

dont find it horrible. – eldo Oct 4, 2016 at 13:50

29

I think .NET had (has) a lot of the same perceived issues that Java has. Microsoft has just done a better job at marketing to developers with XNA :-)

Share Improve this answer Follow

answered Jun 23, 2009 at 18:58
Joel Martinez
47.7k ● 26 ● 134 ● 185

10 XNA also makes it possible to deploy your .NET application to XBox. I haven't seen anything quite that smooth for Java. – StriplingWarrior Jun 23, 2009 at 19:44

1 You can also deploy to the Zune as well. – cbeuker Jun 24, 2009 at 16:41

Bit of an older question, but just to update, you can now write XNA games for Windows Phone as well :-) – Joel Martinez May 4, 2011 at 13:05

3 @JoelMartinez another update: it's not possible to write XNA games for Windows Phone 8. – Tomas Andrle Sep 27, 2012 at 10:28

@TomA It is possible now to write monogame games for WP8 – Alex Lapa Nov 24, 2014 at 21:59

19

Minor points first:

- any productivity boost from Java is hypothetical. The syntax is almost identical to C++ so you're really just banking on savings from memory management and standard libraries. The libraries have little to offer games developers and memory management is a contentious issue due to garbage collection.

- cross-platform "for free" is not as good as you think because few developers want to use OpenGL and several key platforms probably lack a good Java implementation or wrappers for their native libraries, whether for graphics, audio, networking, etc.

But mainly, the issue is backwards compatibility. Games developers moved to C++ from C and to C from assembly purely because the migration route was smooth. Each interoperates closely with the previous, and all their previous code was usable in the new language, often via a single compiler. Therefore migration was as slow or as fast as you liked. For example, some of our old headers in use today still have *#ifdef WATCOMC* in, and I don't think anybody has used the Watcom compiler here in a decade or more. There is a massive investment in old code and each bit is only replaced as needed. That process of replacing and upgrading bits and pieces from one game to the next is nowhere near as practical if you changed to a language that doesn't natively interoperate with your existing code. Yes, C++/Java interoperability is possible, but very impractical by comparison to simply writing "C with a bit of C++" or embedding asm blocks in C.

To properly supercede C++ as the game developers' language of choice, it must do one of two things:

1. Be easily interoperable with existing legacy code, thus preserving investment and maintaining access to existing libraries and tools, OR
2. Demonstrably show up-front enough of a productivity boost that the cost of rewriting all your own code (or reworking the interfaces into reusable components that can be used from that language) is more than covered.

Subjectively, I don't think Java meets either of those. A higher-level language might meet the 2nd, if someone is brave enough to be the pioneer. (EVE Online is probably the best example we have of Python being usable, but which uses a fork of the main Python language, many C++ components for performance, and even that is for a fairly undemanding game in modern terms.)

Share Improve this answer Follow

answered Jun 24, 2009 at 16:29

Kylotan
**18.4k** ● 7 ● 49 ● 75

---

Just wanted to add, EVE Online is an 'online' space simulation, where 1000s vs 1000s player vs player battles are common, which can be counted as a demanding scenario in terms of performance. Although it's speed intensive parts are written in C/C++, it's still an interesting study on the challenges of using a high level language (Python) in games. – Hakan Deryal Oct 15, 2012 at 20:05

---

However, remember that performance in multiplayer server-side games is measured by slightly different metrics to performance in single-player client-side games - the former is

more concerned with throughput, the latter with latency.
– Kylotan Oct 16, 2012 at 0:07

Yep that's true, but that battles includes 2000+ ships on screen, with 2000+ projectiles(missiles, with animations), explosions etc. which requires some heavy graphics performance. Anyways, thanks for the detailed answer, it still holds true. – Hakan Deryal Oct 16, 2012 at 8:16

1 If you think that C & Java syntax are the same and therefore that has some relationship to performance you really don't understand what is going on. How could C possibly decide at runtime that a given function is being called with the same parameters repeatedly and replace the entire function call with a constant while retaining the function call when there is a deviation in the parameters? I'm not saying the runtime is always better or alwyas worse, just that it has no relationship whatsoever to the syntax! – Bill K Sep 26, 2014 at 21:19

1 @BillK - you appear to have misread. I mentioned syntax only with reference to 'productivity' - not to 'performance'. It's true that JIT optimizations could make Java faster in theory, but this does not occur in practice, at least not in game software. – Kylotan Oct 1, 2014 at 13:57

13

I'm playing the Sims 3, and I did some poking around. The graphics engine is C++, while the scripting and behavior engine is C#/Mono. So while C++ is there for time critical bits, other stuff like .interaction, game logic, AI is in an object oriented managed language.

Share Improve this answer Follow

answered Jun 23, 2009 at 19:06

John Simon
329 ●1 ●6

5 and then for the Mac version, they shove the entire thing inside a modified Wine virtual machine. Still faster than it would be in straight Java I think :-) – Ben Gotow Jun 23, 2009 at 19:52

11 Wine is not a virtual machine, it is a runtime library that mimics the behavior of the Windows runtime libraries. Hence the name (Wine Is Not an Emulator). – Nate C-K Nov 29, 2009 at 4:56

2 This is very common in games, quite often logic that isn't time-critical is written in some sort of scripting language, commonly lua or python. – KSchmidt Dec 8, 2009 at 17:21

It's not vanilla Mono, though. EA needed a special team working on their own custom CLR full-time to make it work. – Crashworks Dec 9, 2009 at 2:40

4 Just a side note that the Sims 3 is notorious for underperforming even on excellent computers. – Lotus Notes May 5, 2011 at 22:39

13

- Are there any good ports of gaming engines/libraries?
- Many C/C++ developers, particularly the ones on Windows (where most commercial games are written) are familiar with Visual Studio. There is no comparison in IDEs.
- In general, Java has been sold to businesses because of it's solid typing and it has a perception of not having memory management issues.

- And yes, Java still suffers from a perception that it is slow, and it's memory management is poor, and for games, it probably is ill-suited to the task. As stated in some of the other answers, garbage collection just isn't going to cut it when you are dealing with real-time high-performance requirements. **Video games push CPUs and GPUs to their limits.**

Share Improve this answer Follow

edited Jun 23, 2009 at 19:12

answered Jun 23, 2009 at 18:59

cgp
**41.4k** ● 12 ● 105 ● 131

1 +1 for the bold text. People don't seem to realize that when your game is running at 20 fps, it's often hardware bound at 20 fps. It really want to get to 30+ fps.. but it can't. – GuiSim Jun 23, 2009 at 19:34

I don't think it's just the GC that's the problem performance-wise though...nor even that coupled with the slow startup phase...it's general performance issues, but that's just me. – rogerdpack May 14, 2011 at 16:33

2 I think at this point I am more likely to agree than in the past. Optimization of the JVM has improved; however, in light of the performance improvements to loosely typed languages like JavaScript and others, Java's performance in comparison is pretty inexcusable. There are plenty of apologists for the performance of Java. (but the perceived performance in the end is all that matters)' – cgp May 18, 2011 at 14:30

10

One of the biggest reasons Java and other Virtual Machine languages are not used for games is due to Garbage Collection. The same thing goes for .NET. Garbage collection has come a long ways and works great in most types of applications. In order to do garbage collection though, you do need to pause and interrupt the application to collect the trash. This can cause periodic lag when collection happens.

Java has the same problem for realtime applications. When tasks must run at a specific time, it is hard to have an automated task such as garbage collection respect that.

It is not that Java is slow. It is that Java is not good at handling realtime tasks.

Share Improve this answer Follow

answered Jun 23, 2009 at 19:06

Chris Dail
26k ● 9 ● 68 ● 74

1 You can, however, write your own scheduler for the garbage collector if you're going to go so far as to port Java to a new environment. The memory has to be reclaimed either way, and in a real-time environment, you could have the option of when to schedule your gc... best of both worlds. I have to go back to the point that there isn't much reason to port Java over to an architecture to do the things you want it to do when C/C++ already do those things for you. Java shines in other places. – user124493 Jun 23, 2009 at 20:36

5 This is not the 1990s. Garbage collectors are pretty good now when tuned for low pause. – Tom Hawtin - tackline Jun

23, 2009 at 21:06

8

A large reason is that video games require direct knowledge of the hardware underneath, often times, and there really is no great implementation for many architectures. It's the knowledge of the underlying hardware architecture that allows developers to squeeze every ounce of performance out of a gaming system. Why would you take the time to port Java to a gaming platform, and then write a game on top of that port when you could just write the game?

edit: this is to say that it's more than a "speed" or "don't have the right libraries" issue. Those two things go hand-in-hand with this, but it's more a matter of "how do I make a system like the cell b.e. run my java code? there aren't really any good java compilers that can manage the pipelines and vectors like i need.."

Share Improve this answer Follow

answered Jun 23, 2009 at 19:05

user124493

7

Performance issue is the first reason. When you see the kind of hyper optimized C++ code that are in the Quake engines ( [http://www.codemaestro.com/reviews/9](http://www.codemaestro.com/reviews/9) ), you know they're not gonna waste their time with a virtual machine.

Sure there may be some .NET games (which ones ? I'm interested. Are there some really CPU/GPU-intensive ones ?), but I guess it's more because lot of people are experts in MS technologies and followed Microsoft when they launched their new technology.

Oh and cross-platform just isn't in the mind of video games companies. Linux is just around 1% of market, Mac OS a few % more. They definitely think it's not worth dumping Windows-only technologies and librairies such as DirectX.

Share Improve this answer Follow

answered Jun 23, 2009 at 19:06

Ksempac

**1,892** ● 2 ● 18 ● 24

3 "cross-platform just isn't in the mind of video games companies" -- That's why I fully respect the companies that do. :) – Sasha Chedygov Jun 23, 2009 at 19:25

I'm definitely grateful to Carmack for being so committed to both cross-platform and open-source. I simply stated what most companies think. – Ksempac Jun 23, 2009 at 19:27

1 That's true. You don't see a lot of popular video games ported to Linux. :( – Sasha Chedygov Jun 23, 2009 at 19:34

Cross-platform is not just cross OS. Think of PS3, Xbox 360, Wii. – JulianR Jun 23, 2009 at 20:55

"they're not gonna waste their time with a virtual machine." en.wikipedia.org/wiki/Quake_III_Arena#Virtual_machine, Carmack built his own for the game logic. – James McMahon Aug 4, 2009 at 2:18

5

Misconceptions about performance and poor JVM optimizations would be my guess. I say misconceptions about performance because there are some Java ports of C++ games that perform faster than their C++ counterparts (see Jake 2). The real problem, IMHO, is that many Java programmers aren't focused so much on bleeding edge performance as they are with ease of use and understandability/maintainability of code. On the C/C++ side of things you're essentially coding in a slightly higher level assembly language and its about as close to the hardware as you can get without writing in assembly or straight machine code.

Share Improve this answer Follow

answered Jun 23, 2009 at 19:37

illvm
1,346 ● 13 ● 28

If it's "about as close to the hardware as you can get without writing in assembly", then Java won't be able to beat it, unless your coding is terrible. The closer you get to the hardware, the faster you're going to be able to get.
– Josh Johnson May 7, 2014 at 19:43

---

4

You can ask why web applications aren't written in C or C++, too. The power of Java lies in its network stack and object oriented design. Of course C and C++ have that, too. But on a lower abstraction. Thats nothing negative, but you don't want to reinvent the wheel every time, do you?

Java also has no direct hardware access, which means you are stuck with the API of any frameworks.

Share Improve this answer Follow

answered Jun 23, 2009 at 19:08
Markus
1,822 ● 1 ● 13 ● 20

Java can call native code through "JNI" – Bart van Heukelom Nov 29, 2009 at 22:06

4 ... and lose portability while you're at it! – LiraNuna Dec 9, 2009 at 2:45

1 You really don't lose much portability when you use JNI. Provided that you are still able to compile the native libraries on platforms you want to support, it basically just means you only have to port/recompile 1% of your code rather than all of it. You still get a lot of benefit from Java's portability. – bgroenks Mar 16, 2013 at 16:13

4

List of game engines on Wikipedia lists many game engines along with the programming language that they are written in.

**There are several Java game engines listed.**

Clicking some of the links will lead you to examples of games and demos written in Java. Here's a couple:

- Ardor3D
- Bytonic Software

For certain games and situations, Java's trade-offs might be acceptable.

Share Improve this answer Follow

edited Nov 17, 2011 at 1:57

answered Nov 17, 2011 at 0:33

JohnB
18.9k ● 17 ● 101 ● 113

I know games *exist* that are written in Java. But besides Minecraft and Runescape, very few mainstream commercial games are written for the Java platform. How many AAA titles were written in Java? And why so few? Hence my question.
– Sasha Chedygov Nov 18, 2011 at 9:41

---

3

.NET definitely has some of the same issues that Java has when it comes to intense 3D performance. Microsoft has also invested a lot more time and money in the development of the libraries when it comes to working with 3D heavy operations.

(...personally, I also think they had a leg up when it comes to the magic between DirectX and .NET)

Share Improve this answer Follow

answered Jun 23, 2009 at 18:59

Justin Niessner
245k ● 40 ● 414 ● 544

3

1. Java is slow, most of the heavy lifting is not handled by the GPU. There's still animation, physics, and AI hitting the CPU, all of which are very time-consuming.
2. Java doesn't exist on consoles, and consoles are a major target for commercial games. If you use Java on PC, you're eliminating your ability to port to consoles within reasonable time and budget.
3. Many of the more experienced coders in the game industry have been using C and C++ long before Java became popular. The two points above may contribute to this, but I expect that many professional game coders just don't really know Java all that well.
4. Someone else's point about middleware above was a good one, so I'm adding it to my answer. There's a lot of legacy code and middleware written specifically to link with C/C++, and last I checked Java doesn't have good interoperability. Using Java for most companies would involve throwing out a lot of code, much of which has been paid for in one way or another.

Share Improve this answer Follow

edited Jun 23, 2009 at 20:14

answered Jun 23, 2009 at 19:47

Dan Olson
23.3k ● 4 ● 43 ● 56

3 You can use JavaCL, JOCL, or APARAPI to offload a lot of that to the GPU. – bgroenks Mar 16, 2013 at 16:14

2

I'd guess that speed is still the issue. Cross platform is going to be an issue isn't it since you don't know what 3d card is available when you write the code? Does java have anything to support auto discovery of 3d capabilities? And I'd guess that there are tools to ease porting a game between the wii, xbox, and ps3, but expensive I'll bet.

The ps3 has java, via the blue ray support. Check the bd-j site.

Share Improve this answer Follow

answered Jun 23, 2009 at 19:03

user98989

2

Actually, it is very possible for managed code to do 3d games, the problem is the back engines. With .Net, for a brief period, there was a Managed DirectX wrapper to DirectX 9 by Microsoft. This was before the abstraction that is now XNA.

Being given total access to DirectX api's, .Net games work a treat. The best example I know of is www.entombed.co.uk, which is written in VB.Net.

Unfortunately, on the Java side, it is seriously lacking - mainly for the reason that DirectX isn't available for Java, and games programmers know and understand the DirectX api - why learn yet another api when you will be returning to DirectX?

Share Improve this answer Follow

answered Dec 8, 2009 at 17:14
Foz
21 ● 1

---

2

Game marketing is a commercial process; publishers want quantifiable low-risk returns on their investment. As a consequence, the focus is usually on technology gimmicks (with exceptions) that consumers will buy to produce reliable return - these tend to be superficial visual effects such as lens glare or higher resolution. These effects are reliable because they simply use increases in processing power - they exploit the hardware/Moore's law increases. this implies using C/C++ - java is usually too abstracted from the hardware to exploit these benefits.

Share Improve this answer Follow

answered May 10, 2012 at 20:58
Daniel Collicott
69 ● 3

---

1

Runescape by Jagex is written in Java, the "video game" tag might not specifically apply it being an on-line game, but it does have a decent following.

Share Improve this answer Follow

answered Jun 24, 2009 at 3:05

Mark Schultheiss
34.1k ● 12 ● 72 ● 108

Sorry, but this doesn't really answer my question at all.
– Sasha Chedygov Jun 24, 2009 at 12:09

2 But the blind statement of the question leads to the assumption that NO games are written in Java, just pointing out the successful case of where one is. – Mark Schultheiss Nov 18, 2009 at 20:26

---

1

Even games written on the .Net platform are often highly optimized for speed like direct access to memory and bus. .Net allows to use C / C++ and mix it with higher level languages such as C#.

Game development studios often work close together with hardware vendors, which do provide access to low level interfaces of their products. This is a world, where you have to use ASM and C for device communication. A virtual environment would slow down these program parts.

Anyway, modern 3D games in fact do use higher level languages. Often, you'll find the game logic written in languages like Lua or Python. But the core (I/O, threads, task scheduling) of the typical 3D game will be written in low level languages for the next 25 years or as long devices do not allow abstraction and virtualization by themself (which will come).

Share Improve this answer Follow

answered Jun 25, 2009 at 14:34

cafebabe
1,400 ● 8 ● 9

---

1

I agree with the other posts about leveraging elements of a preexisting/licensed codebase, performance, etc.

One thing I'd like to add is it's hard to pull nasty DRM tricks through a virtual machine.

Also I think there's a hubris component where project managers think they can make stable/reliable code with C++ with all the perks like having absolute control over their tools and resources, BUT without all the negatives that complicate and bog down their competition because "we're smarter than they are".

Share Improve this answer Follow

answered Jun 25, 2009 at 18:30

JP772
73 ● 1 ● 2

---

-2

It was talked about it a lot already, u can find even on Wiki the reasons...

- C/C++ for the game engine and all intensive stuff.
- Lua or Python for scripting in the game.

- Java - very-very bad performance, big memory usage + it's not available on Game Consoles(It is used for some very simple games(Yes, Runescape counts in here, it's not Battlefield or Crysis or what else is there) just because there are a lot of programmers that know this programming language).
- C# - big memory usage(It is used for some very simple games just because there are pretty much programmers that know this programming language).

And I hear more and more Java programmers that try to convince people that Java is not slow, it is not slow for drawing a widget on the screen and drawing some ASCII characters on the widget, to receive and send data through network(And it is recommended to use it in this cases(network data manipulation) instead of C/C++)... But it is damn slow when it comes to serious stuff like math calculations, memory allocation/manipulation and a lot of this good stuff.

I remember an article on MIT site where they show what C/C++ can do if u use the language and compiler features: A matrix multiplier(2 matrices), 1 implementation in Java and 1 implementation in C/C++, with C/C++ features and appropriate compiler optimisations activated, the C/C++ implementation was ~296 260 times faster than the Java implementation.

I hope you understand now why people use C/C++ instead of Java in games, imagine Crysis in Java, there would not be any computer in this world which could handle that... + Garbage collection works ok for Widgets which just destroyed an image but it's still cached in there and needs to be cleaned but not for games, for sure, u will have even more lags on every garbage collection activation.

**Edit**: Because somebody asked for the article, here, I searched in the web archive to get that, I hope you are satisfied...MIT Case Study

And to add, no, Java for gaming is still an awful idea. Just a few days ago a big company that I will not name started rewriting their game **client** from Java to C++ because a very simple game(In terms of Graphics) was lagging and heating i7 Laptops with powerful nVidia GT 5xx and 6xx generation video cards(not only nVidia, the point here is that this powerful cards that can handle on Max settings most of the new games and can't handle this game) and the memory consumption was ~2.5 - 2.6 GB Ram. For such simple graphics it needs a beast of a machine.

Share Improve this answer Follow

edited Jul 26, 2013 at 15:14
nanofarad
41.3k ● 4 ● 91 ● 126

answered Oct 30, 2011 at 14:03
Lilian A. Moraru
1,066 ● 1 ● 12 ● 20

12 You clearly know very little about the modern Java runtime and virtual machine. The article you mentioned is more than likely from a decade ago or more, of course no one can

know because you didn't cite it. Your perception of Java is outdated. – bgroenks Feb 1, 2013 at 0:01

2 Ok so that study proves that for large scale matrix multiplication Java loses to C when 2-dimensional array access is being used to access data. Yes I would've guessed that too. And if that really is a problem for you, which I doubt it would be, that's why you have JNI. The bounds checking overhead for arrays does add up in that situation, although his Java code could've been optimized to significantly improve results. Likewise, I question his understanding of JIT when he states, "faster compilation = not the best code generated." Go read IBM's specification's to prove otherwise. – bgroenks Mar 15, 2013 at 4:24

4 Java is NOT a bad choice for game development. There are a lot of successful games out there that run Java. Typically, you do need a little bit of help from native code (especially with LWJGL and such) to really get the best results. But if I only have to port and recompile 1% of my code rather than 100%, that sounds like a great deal to me. – bgroenks Mar 15, 2013 at 4:26

1 @bgroenks "100%" - looks like you have no idea about C/C++... And when making a game you can always use a cross-platform library(SDL and a bunch of others) or framework(Qt for example). For example: EA uses Qt for absolutely every game they have... Qt is WAY more cross-platform than Java and it compiles to native code. – Lilian A. Moraru Mar 15, 2013 at 20:57

2 I don't really see the point in Java when you have Qt. I find Qt code more clear, easier to understand and maintain than Java code. When I ask my friends why they are so afraid of C++ they always tell me that they hate pointers and making sure to deallocate memory. Looks like a lot of people don't know about shared_ptr in C++... For me, Qt and C# .NET/C++ .NET are the nicest to write in. Java code usually is very bloated with exception handling, has usually outdated

libraries(It is doing good mostly only on the server side but the rest...) and frequently outdated documentation.
– Lilian A. Moraru Mar 16, 2013 at 8:47