Is there a stable Programming Language for Web Programming?

Asked 15 years, 10 months ago Modified 8 years, 5 months ago

Viewed 2k times

7

A renowned PHP user once said: There will be a relaunch in 2 years, anyway.

Those times are gone. Web applications that are older than 5 years are common. With the original developer(s) gone.

The release cycles of the operation system, programming language, and framework are getting in the way of doing real work, if you don't have a big staff.

Is there any way to develop something that doesn't need constant porting to the next level, without the fear of losing support and backing in a community? For people who want to stay in programming instead of climbing the corporate ladder and leaving the problems to the next "generation"?

programming-languages web

Share

Improve this question

edited Sep 1, 2012 at 21:54

Follow

asked Feb 23, 2009 at 20:54

stesch

7,214 ● 6 ● 49 ● 63

Um, .NET 2 has been going for 5 years now... – annakata Feb 23, 2009 at 21:07

my suggestion, stick to the old version if you didn't make total modification. think about how much times needed to make from scratch and bug testing as well ... – nightingale2k1 Jul 3, 2009 at 2:07

11 Answers

Sorted by: Highest score (default)

20

My company codes almost exclusively in C#, however we have ColdFusion 5 apps still humming along written back in 2001 or so. Theres no need to port them.

If it ain't broke, dont fix it.

Other than security flaws (which are usually handled by an OS/Server Patch, so they dont need code changes), theres no need to change an app just because a new version of the language has come out.

If I'm not mistaken, ColdFusion has had at least 2 new releases since we stopped using it for new code. but that hasn't affected our ColdFusion sites one bit.

Share Improve this answer edited Feb 10, 2010 at 20:47

Follow

answered Feb 23, 2009 at 20:57

Neil N

25.3k ● 17 ● 86 ● 146

9

Write CGI programs in FORTRAN 77. Should be pretty stable.

Share Improve this answer

Follow

answered Feb 23, 2009 at 21:20

anon

2 These aren't going to change anytime soon. – Adam Davis Feb 23, 2009 at 21:23

All implementations I'm aware of can do so. – anon Jul 9, 2009 at 10:10

9

Firstly, it is possible to overstate the difficulty in maintaining web applications. In many cases, the changes to a language or platform are expansionary in nature rather than destructive. .NET, python, etc code from several years ago will still run, but new options are being added to make these these tools more powerful for future applications. The case where massive changes occur tends to be on the first or second iteration of a language, e.g. Rails 1 to Rails 2.

Secondly, the still active development of web programming is something to be thankful for.It means that this is a part of the industry that will remain productive and exciting for years to come.

Share Improve this answer

Follow

answered Feb 23, 2009 at 21:45

Rob Lachlan

14.4k ● 5 ● 50 ● 99

6

[Traditional CGI](#) is stable. It's not sexy, but if your OS continues to be able to run the same binaries or scripts, it's still going to work.

Share Improve this answer

Follow

answered Feb 23, 2009 at 20:58

Greg Hewgill

990k ● 191 ● 1.2k ● 1.3k

CGI is a specification for how the web server and an external program communicate with each other. It is not a language. CGI programs can be written in any language - C, Perl, shell script, even PHP. Running it through CGI doesn't make it any more (or less) stable. – Dave Sherohman Feb 23, 2009 at 21:59

2 He's just saying in a roundabout way to not even use a framework and just code CGI apps in your programming language of choice. If you program a CGI app in C or Perl, it's not gonna have to change much over the years to stay current. – DevelopersDevelopersDevelopers Feb 23, 2009 at 22:08

6

The only programming frameworks that stay stable are those that have been abandoned. A framework that stood still long enough would have no support for, say, AJAX or JSON or even XML.

You're not going to find what you're asking for. The best you can hope for is a mature framework with good support like ASP.net or JSP. And, as @Neil N said, don't keep upgrading unless there's a compelling business need.

Share Improve this answer

Follow

answered Feb 23, 2009 at 21:02

Yes - that Jake.

17.1k ● 15 ● 72 ● 99

2 I always get a bloody nose from "the community" if I mention the old version. Upgrading seems to be some kind of fetish and you are forced to take part in it. – stesch Feb 23, 2009 at 21:17

as much as I hate, hate, HATE cold fusion(with a passion), I would never recomend rewritting it unless something broke or the client wants a complete redesign. "If it aint broke dont fix it" is my #1 rule. I have almost gone to blows with coworkers who fixed something vs finishing a new task first. – Neil N Feb 23, 2009 at 21:30

2

The first web programming I ever did was writing Apache modules in C which communicated with a dBase database. I'm fairly sure that code would still run today (if the company I wrote it for still existed).

I do most of my current web-related programming in Perl, which is very stable and has an excellent track record for backwards compatibility. Most, if not all, code written for Perl 4 (released 21 March, 1991) should still run on the latest stable Perl (5.10) - although you might want to update it anyhow to take advantage of the last 18 years of improvements in both software development techniques and language features.

Share Improve this answer

Follow

answered Feb 23, 2009 at 22:07

Dave Sherohman

46.1k ● 14 ● 66 ● 103

Perl 4 was 1991? Damn, I never would have guessed it was that old. – Neil N Feb 23, 2009 at 22:10

I doubt it that typical Perl 4 code runs on Perl 5.10. Or am I missing some compatibility module? But thanks for mentioning Perl. I haven't used it for some time now, but I'm beginning to miss it a little bit. – stesch Feb 24, 2009 at 5:54

2

Consider the shearing layers. I've previously worked in large aerospace companies where the same Fortran back-end code and databases have had their front-ends evolve from the paper tape era through mainframe, client server and onto Intranet web sites.

On the outside, you have will typically have CSS and XHTML templates which can be changed to re-skin an application. These change quite rapidly, in large

organisations as upper management seems to decide the bike shed should be a different colour every few weeks.

Typically you then have some logic to combine the templates with data from the back-end, and forward user actions to the back-end. This shouldn't change that rapidly, but translate the presentation to calls into the back-end. Expect to refresh this every few years, and rewrite it once a decade. We used Java for this, starting in the late 1990s. Some parts get changed faster than others, but it's not a big issue.

The back-end is usually stable (some of the aerodynamics code dated from the 1970s; the laws of physics don't change that often), and will outlast the web UI, as it has all the other UI paradigms. Fortran is forever.

Share Improve this answer

Follow

answered Jul 9, 2009 at 9:51

Pete Kirkham

49.3k ● 5 ● 94 ● 173

modded up if only for the poetry "Fortran is Forever" well done that man. – Bob Blanchett Nov 1, 2010 at 13:52

1

Write your own web server in C then you don't have to worry about a web programming language.

(No, that's not a serious answer)

edited Jul 9, 2009 at 9:21

Share Improve this answer

Follow

answered Feb 23, 2009 at 21:34

Chris S

65.4k ● 53 ● 224 ● 240

Yeah, you'll have much bigger things to worry about
– Mike Robinson Nov 16, 2009 at 17:24

0

Have you seriously looked at what TDD, CI, pair-programming, and a solid, rapid development framework (basicaly Django or Rails) can offer to you as a developer vis-a-vis the way you write and design code? There are some really massive benefits that all of those pieces offer to the development process that make it almost a joy to be a programmer again. There are downsides, of course, but the upsides are all in support of the happiness and ease of development for the engineer, which leads to more productivity. In my book, that's a slam dunk win. And the result of my productivity and happiness, has been solid products and great engineering.

YMMV, but if you are having the serious thoughts that you are (and I take them very seriously), I think it's worth you investigating what those tools can offer. By taking the good and leaving the bad from the agile religion plus some of the things I listed above, I've returned to find the joy in programming again this last year, after a good 5 years of a downhill slide of my happiness with this career. It's about finding what works for you. I can only help and

lead the way by showing you what worked for me. I'd be more than happy to discuss at length if you want to talk offline, I think this is a really important topic...it lead me to consider a career change many times.

Share Improve this answer

Follow

answered Feb 27, 2009 at 6:06

ryan.scott

2,205 ● 2 ● 18 ● 16

> I believe so, yes, but I maybe misunderstood you. Even re-reading it now, it's not totally clear what your question is. I was honestly trying to answer what I thought you were asking. Please help me understand what you as asking, I would love to offer my thoughts if you want to hear them.
> – ryan.scott Feb 27, 2009 at 18:04

0

Java Servlets and JSPs have been in use for a decade or so, and they still work the same way like they did in '99. But honestly, can you imagine something uglier than a '90s web application without any rework done since?

Share Improve this answer

Follow

answered Nov 16, 2009 at 17:27

Erich Kitzmueller

37k ● 5 ● 84 ● 103

0

The Python web framework web2py promises backward compatibility:

> **Always backward compatible.** We have not broken backward compatibility since version 1.0

> in 2007, and we pledge not to break it in the future.

And supports Python versions from 2.4 to 2.7

EDIT: Updated an important project 2 times and every time there was a problem. Well, …

EDIT 2: Needs Python 2.6 to 2.7 now. No support for Python 3.

Share Improve this answer Follow

edited Jul 11, 2016 at 9:21

answered Oct 31, 2010 at 12:20

stesch

7,214 ● 6 ● 49 ● 63

Updated after 1 and then again after 6 years to share your experience. That's some awesome commitment, thank you! It's obvious that being super-stable / backward-compatible has strong drawbacks (no support for newer things where you need to replace an older thing). – Oliver Adria Jan 13, 2021 at 7:33