What makes code legacy?

Asked 15 years, 11 months ago Modified 5 years, 7 months ago

Viewed 10k times

46

I have heard many developers refer to code as "legacy". Most of the time it is code that has been written by someone who no longer works on the project. What is it that makes code, legacy code?

Update in response to: "Something handed down from an ancestor or a predecessor or from the past" http://www.thefreedictionary.com/legacy. Clearly you wanted to know something else. Could you clarify or expand your question? S.Lott

I am looking for the symptoms of legacy code that make it unusable or a nightmare to work with. When is it better to throw it away? It is my opinion that code should be thrown away more often and that reinventing the wheel is valuable part of development. The academic ideal of not reinventing the wheel is a nice one but it is not very practical.

On the other hand there is obviously legacy code worth keeping.

legacy

Share

Improve this question

Follow

edited Jan 26, 2009 at 12:54

asked Jan 26, 2009 at 12:15

Chris de Vries

57.6k ● 6 ● 33 ● 27

1 You clarified the question, but didn't change the title. Your question is NOW about symptoms of unmaintainable legacy code. Please update the question title, also. – S.Lott Jan 26, 2009 at 13:30

21 Answers

Sorted by: Highest score (default)

47

By using hardware, software, APIs, languages, technologies or features that are either no longer supported or have been superceded, typically combined with little to no possibility of ever replacing that code, instead using it til it or the system dies.

Share Improve this answer

Follow

answered Jan 26, 2009 at 12:19

cletus

625k ● 169 ● 917 ● 945

26

> What is it that makes code, legacy code?

As with plain legacy, when the author is dead or missing, you as a heir get all or some of his code.

You shed some tears and try to figure out what to do with all this rubbish.

Share Improve this answer

Follow

answered Jan 26, 2009 at 12:18

Quassnoi

425k ● 93 ● 625 ● 619

23

Michael Feathers has an interesting definition in his book Working Effectively with Legacy Code. According to him legacy code is code without automated tests.

Share Improve this answer

Follow

answered Jan 26, 2009 at 12:22

Brian Rasmussen

116k ● 34 ● 224 ● 323

7 What pie in the sky nonsense. Well, those who can, do, those who can't, teach (or coach software development teams). – cletus Jan 26, 2009 at 12:25

3 If you think about legacy code as code which is hard to maintain, then the definition makes perfect sense imo. – Brian Rasmussen Jan 26, 2009 at 13:19

14 Semantics. The point of Michael's definition is that if you write code that can't be easily changed, it's already legacy before you even finish. To write code that can be changed, you must have tests (at a minimum). There may be more to it, but the simplification is meant as a rhetorical device. – Adam Bellaire Jan 26, 2009 at 15:27

1 @Adam: the idea that you can unit testing any piece of code is what I was referring to as "pie in the sky nonsense". Not all code is possible (or even practical) to unit test. Javascript engines and CSS layout spring to mind. – cletus Jan 27, 2009 at 8:10

3 @Adam Bellaire: Michael wants to stress the importance of tests, and of course he does have a valid point. But as a definition for legacy its just bollocks. Semantics are the basis of a definition... – Treb Jan 27, 2009 at 13:54

17

It is a very general (and oft abused term) but any of the following would be legitimate reasons to call an app legacy:

1. The code base is based on a language/platform which is entirely unsupported by the manufacturer of the original product (often said manufacturer has gone out of business).
2. (really 1a) The code base or platform on which it is built is so old that getting qualified or experienced developers for the system is both hard and expensive.
3. The application supports some aspect of the business which is no longer actively grown and for which alterations are extremely rare, normally to fix it if something entirely unexpected changes around it (the canonical example being the Y2K issue) or if some regulation/external pressure forces it. Since both reasons are pressing and normally unavoidable but no significant development has occurred on the

project it is likely that those people assigned to deal with this will be unfamiliar with the system (and it's accumulated behaviours and intricacies). In these cases this would often be reason to increase the perceived and planned for risk associated with the project.

4. The system has/or is being replaced with another. As such the system may be used for much less than originally intended, or perhaps only as a means of viewing historical data.

Share Improve this answer

Follow

answered Jan 26, 2009 at 12:25

ShuggyCoUk

36.4k ● 6 ● 82 ● 102

12

Legacy generally refers to code that is no longer being developed - meaning that if you use it, you have to use it on its original terms - you cannot just edit it to support the way the world looks today. For example, legacy code has to run on hardware that may not exist today - or is no longer supported.

Share Improve this answer

Follow

answered Jan 26, 2009 at 12:21

user17541

139 ● 1 ● 2 ● 6

11

According to Michael Feathers, the author of the excellent Working Effectively with Legacy Code, legacy code is a

code which has no tests. When there is no way to know what breaks when this code changes.

> The main thing that distinguishes legacy code from non-legacy code is tests, or rather a lack of tests. We can get a sense of this with a little thought experiment: how easy would it be to modify your code base if it could bite back, if it could tell you when you made a mistake? It would be pretty easy, wouldn't it? Most of the fear involved in making changes to large code bases is fear of introducing subtle bugs; fear of changing things inadvertently. With tests, you can make things better with impunity. To me, the difference is so critical, it overwhelms any other distinction. With tests, you can make things better. Without them, you just don’t know whether things are getting better or worse.

Share Improve this answer

Follow

answered Jan 26, 2009 at 12:26

Igal Tabachnik

31.5k ● 15 ● 95 ● 157

1 I guess by that definition jQuery is legacy code. It's what I'd expect from a "software development coach" or an academic. – cletus Jan 26, 2009 at 12:35

The jQuery project has extensive unit tests. See docs.jquery.com/QUnit IMO, unit testing is not a universal standard yet, so it's unfair to call applications without them "legacy." I think in another 10 years or so this might be true. – davogones Feb 1, 2009 at 4:12

1 The fear of changing code because you're not sure it will be better when you're done is paralyzing! Did I make things better or did I do damage? Great products go down hill when devs are unwilling to make even the simplest changes due to the consequences they may incur! – Airn5475 Sep 11, 2018 at 19:42

11

Nobody is gonna read this, but I feel the other answers don't get it quite right:

1. It has value, if it wasn't useful it would've been thrown away long ago
2. Its hard to reason about because either of
    1. Lack of documentation,
    2. Original author cannot be found or forgot (yes 2 months later your code can be legacy code too!!),
    3. Lack of tests or typesystem
    4. Doesn't follow modern practices (ie no context to hold on too)
3. There is a requirement to change or extend it. If there isn't a requirement to change it, it isn't legacy code since nobody cares about it. It does its thing and there is nobody around to call it legacy code.

Share Improve this answer

Follow

answered Dec 6, 2016 at 15:48

Jappie Kerk

1,307 ● 13 ● 16

One of the most balanced answers around. +1 – Sam Jun 28, 2019 at 0:22

8

A colleague once told me that legacy code was any code that you hadn't written yourself.

Arguably, it's just a pejorative term for code that we don't like any more for whatever reason (typically because it's not cool or fashionable but it works).

The TDD brigade might suggest that any code without tests is legacy code.

Share Improve this answer

Follow

answered Jan 26, 2009 at 14:25

Dan Dyer

54.4k ● 19 ● 135 ● 166

I'll let Twitter put things in perspective: "Code you haven't touched in 6+ months, might as well been written by someone else." – Airn5475 Sep 11, 2018 at 19:47

7

Legacy code is source code that relates to a no-longer supported or manufactured operating system or other computer technology.

Share Improve this answer

Follow

answered Jan 26, 2009 at 12:20

Galwegian

42.2k ● 16 ● 113 ● 158

5

http://en.wikipedia.org/wiki/Legacy_code

"Legacy code is source code that relates to a no-longer supported or manufactured "

Share Improve this answer

Follow

answered Jan 26, 2009 at 12:19

Sergio

8,251 ● 10 ● 48 ● 77

6 ...operating system or other computer technology" Interesting style of quotation you've got going there. :P. Who needs nouns after all. – RJFalconer Jan 26, 2009 at 15:13

5

Any code with support (or documentation) missing. Be it:

- inline comments
- technical documentation
- spoken documentation (the person who wrote it)

- unit tests documenting the workings of the code

Share Improve this answer

Follow

answered Jan 26, 2009 at 12:22

Gerrie Schenck

22.4k ● 21 ● 69 ● 96

5

For me legacy code is code that was written prior to some paradigm shift. It may still be very much in use but it is in the process of being refactored to bring it into line.
e.g. Old procedural code hanging around in an otherwise OO system.

Share Improve this answer

Follow

answered Jan 26, 2009 at 14:47

meouw

42.1k ● 11 ● 54 ● 69

2

Code (or anything else, really) becomes "legacy" when it has been replaced by something newer/better, and yet despite this it's still used and kept alive "in the wild".

Share Improve this answer

Follow

answered Jan 26, 2009 at 14:41

Wayne Molina

19.6k ● 26 ● 102 ● 164

2

Preserving legacy code is not so much an academic ideal as it is keeping code that works, no matter how poorly. In many conservative enterprise situations, that would be

considered more practical than throwing it away and starting again from scratch. Better the devil you know...

Share Improve this answer

Follow

answered Jan 26, 2009 at 15:22

Relic

166 ● 1 ● 2

> It usually *is* more practical than throwing it away and starting from scratch. – MarkJ May 5, 2009 at 20:45

2

Legacy code is code that is painful/expensive to keep current with changing requirements.

There are two ways that this can happen:

1. The code is unsuitable for change
2. The semantics of the code have been swapped out to silicon

1) is the easier of the two to recognize. It is software that has fundamental limits making it unable to keep up with the ecosystem around it. For example, a system built around O(n^2) algorithm won't scale beyond a certain point and must be re-written if requirements move in that direction. Another example is code using libraries that are not supported on the latest OS versions.

2) Is harder to recognize, but all code of this kind shares the characteristic that people are afraid to change it. This could be because it was badly written/documented to begin with, because it is untested, or because it is non-

trivial and the original authors who understood it left the team.

The ASCII/Unicode chars that comprise living code have semantic meaning, the "why's", "what's" and to some degree the "how's", in the minds of people associated with it. Legacy code is either un-owned or the owners do not have meaning associated with large portions of it. Once this happens (and it could happen the next day with really poorly-written code), to change this code, someone must learn it and understand it. This process is a significant fraction of the time it takes to write it in the first place.

Share Improve this answer

Follow

answered Jun 1, 2015 at 22:49

David Gladfelter

4,213 ● 2 ● 26 ● 25

2

The day you're afraid to refactor your code is the day when your code has become legacy.

Share Improve this answer

Follow

answered May 14, 2018 at 10:48

Mikhail Kholodkov

25.1k ● 17 ● 62 ● 81

I consider code "legacy" if any or all of the following conditions apply:

1

- It was written using a language or methodology that is a generation behind current standards
- The code is a complete mess with no planning or design behind it
- It is written in outdated languages and in an outdated, non object-oriented style
- It is difficult to find developers who know the language because it is so old

Unlike some of the other opinions here, I've seen plenty of modern applications that work decently without unit tests. Unit testing still has not caught on with everyone. Perhaps ten years from now the next generation of programmers will look at our current applications and consider them "legacy" for not containing unit tests, just as I consider non object-oriented applications to be legacy.

If few changes need to be made to a legacy codebase, it's better to simply leave it as-is and go with the flow. If the application needs drastic functionality changes, a GUI overhaul, and/or you can't find anyone who knows the programming language, it's time to throw away and start over. A word of warning, however: rewriting from scratch can be very time-consuming, and it's difficult to know if you've replicated all functionality. You'll probably want to have test cases and unit tests written for the legacy application and the new application.

 answered Feb 1, 2009 at 4:08

Follow

davogones
7,379 ● 32 ● 36

0

Quite honestly legacy code is any code, framework, api, of other software construct thta's not "cool" anymore. For example COBOL is unanimously regarded as legacy while APL is not. Now one can also make the case that COBOL is consideed legacy and APL not because it has about 1m times the install base as APL. However, if you say that you need to work on APL code the reply would not be "oh no, that legacy stuff" but rather "oh my god, guess you won't be doing anything for the next century" see the difference?

Share Improve this answer

Follow

answered Jan 26, 2009 at 15:06

klyde
215 ● 3 ● 14

0

This is a general term thrown around quite often (and quite generically) in the software ecosystem.

Well, I like to think of legacy code as **inherited code**. This is simply code that was written in the past. In most cases, legacy code do not follow new/current practices and is often considered archaic.

Share Improve this answer

Follow

answered May 4, 2019 at 12:37

Taslim Oseni
6,245 ● 10 ● 50 ● 72

Legacy code is anything written more than a month ago :-)

-1

Share Improve this answer

Follow

answered Feb 23, 2009 at 13:52

Roger Lipscombe

91.6k ● 59 ● 252 ● 396

It's often any code that isn't written in the trendy scripting language du jour, and I'm only half joking.

-2

Share Improve this answer

Follow

answered Oct 7, 2011 at 9:28

Alan B

4,278 ● 27 ● 37