Commenting code that is removed [closed]

Asked 16 years, 1 month ago Modified 6 years, 2 months ago

Viewed 4k times

34

votes

Closed. This question is opinion-based. It is not currently accepting answers.

Closed 7 years ago.

Locked. This question and its answers are locked because the question is off-topic but has historical significance. It is not currently accepting new answers or interactions.

Is it a good practice to comment code that is removed? For example:

```
// Code to do {task} was removed by Ajahn on
10/10/08 because {reason}.
```

Someone in my developer group during a peer review made a note that we should comment the lines of code to be removed. I thought this was a terrible suggestion, since it clutters the code with useless comments. Which one of us is right?

comments

Share

edited Oct 5, 2018 at 17:22

community wiki
9 revs, 5 users 40%
David Koelle

Comments disabled on deleted / locked posts / reviews

28 Answers

Sorted by: Highest score (default)

106

votes

Generally, code that is removed should not be commented, precisely because it clutters the codebase (and, why would one comment on something that doesn't exist?).

Your defect tracking system or source control management tools are where such comments belong.

Share

answered Nov 11, 2008 at 15:24

community wiki
David Koelle

5 However, there may be situations in which you say "I removed what seems like it SHOULD go here because of this"

especially if it is unintuitive. These situations are VERY few and far between, however. – Mark S. Nov 11, 2008 at 22:08

1 That's a fair point. Marking WHY the code does what it does is much more intuitive than hoping the developer will ready the entire change history of the file. – Mr. Boy Dec 7, 2009 at 15:15

29 votes

There are some (rare) situations when commenting code out (instead of deleting) is a good idea. Here's one.

I had a line of code that seemed good and necessary. Later I realized that it is unnecessary and harmful. Instead of deleting the line, I commented it out, adding another comment: "The line below is wrong for such and such reason". Why?

Because I am sure next reader of the code will first think that *not* having this line is an error and will try to add it back. (Even if the reader is me two years from now.) I don't expect him to consult source control first. I need to add comment to warn him of this tricky situation; and having wrong line and the reason why it is wrong happened to be the best way to do so.

Share

answered Nov 11, 2008 at 16:55

community wiki
buti-oxa

1 Why not just add a comment to the method or class or function etc. stating in a meta way what has been tried before and why it didn't work, maybe with a pointer to a given revision, without leaving the 'code' in the source file? That's what I would lean towards doing. – Tim Visher Nov 11, 2008 at 19:49

Because it is just line of code, any meta-way description would end up being longer and would clutter the code more. – buti-oxa Nov 11, 2008 at 19:54

6 Why not just include a test for the broken behavior that line caused in the unit tests? – RHSeeger Dec 2, 2008 at 12:47

1 You should add better documentation of why it's working like this then showing a list of examples of ways it's not working. – Paco Dec 2, 2008 at 22:52

14 votes

I agree that it is not a good idea to leave code removed in comments.

Code history should be viewed through a version control system, which is where old code can be found, as well as the reason it was removed.

Share

answered Nov 11, 2008 at 15:25

community wiki
Avi

8 You should delete the code always.

votes

As for being able to see old/removed code, that's what revision control is.

Share

answered Nov 11, 2008 at 15:25

community wiki
Marko

> Yeah, because developers always look at the version history of every file before changing it. The most 'correct' solution needs to be tweaked to take human nature into account. – Mr. Boy Dec 7, 2009 at 15:17

> Well, if you need old code, revision control will help find it. Commenting large amount of code just makes more noise which kills signal. – Marko Dec 7, 2009 at 17:13

6

votes

Depends on the reason for removal.

I think of comments as hints for people maintaining the code in the future, if the information that the code was there but was removed can be helpful to someone maintaining the code (maybe as a "don't do that" sign) then it should be there.

Otherwise adding detailed comments with names and dates on every code change just make the whole thing unreadable.

Share

answered Nov 11, 2008 at 15:27

community wiki
Nir

4 votes

I think it's pretty useless and make the code less readable. Just think what it will be like after some monthes....

```
// removed because of this and that
/*
      removed this stuff because my left leg...
*/
 doSomething();
// this piece of has been removed, we don't need
it...
```

You'll spend half an hour to find out what's going on

Share

answered Nov 11, 2008 at 15:27

community wiki
andy.gurin

3 votes

The question is, why do you remove code?

Is it useless? Was it a mistake to put it there in the first place?

No comments needed from my point of view.

Share

answered Nov 11, 2008 at 15:25

community wiki
Burkhard

2
votes

It's useful when debugging, but there's no reason to *check in* code that way. The whole point of source control is being able to recover old versions without cluttering up the code with commented-out code.

Share

answered Nov 11, 2008 at 15:25

community wiki
Ryan Lundy

2
votes

I would suggest that, yes it's good practice to comment on code that has been removed but **not in the code itself**.

To further clarify this position, you should be using a source code control system (SCCS) that allows some form of check-in comment. That is where you should place the comments about why code was removed. The SCCS will provide the full contextual history of what has happened to the code, including what has been removed. By adding check-in comments you further clarify that history.

Adding comments in the code directly simply leads to clutter.

Share

answered Nov 11, 2008 at 15:26

community wiki
Scott Dorman

---

2 votes

The recent consensus (from other discussions on here) is that the code should just be removed.

I personally will comment out code and tag it with a date or a reason. If it's old/stale and I'm passing through the file, then I strip it out. Version control makes going back easy, but not as easy as uncommenting...

Share

answered Nov 11, 2008 at 15:26

community wiki
Brian Knoblauch

---

2 votes

It sounds like you are trying to get around versioning your code. In theory, it sounds like a great idea, but in practice it can get very confusing very quickly.

I highly recommend commenting code out for debugging or running other tests, but after the final decision has been made remove it from the file completely!

Get a good versioning system in place and I think you'll find that the practice of commenting out changes is messy.

Share

answered Nov 11, 2008 at 15:28

community wiki
JoshFinnie

2
votes

Nobody here has written much about *why* you shouldn't leave commented-out code, other than that it looks messy. I think the biggest reason is that the code is likely to stop working. Nobody's compiling it. Nobody's running it through unit tests. When people refactor the rest of the code, they're not refactoring it. So pretty soon, it's going to become useless. Or worse than useless -- someone might uncomment it, blindly trusting that it works.

There *are* times when I'll comment out code, if we're still doing heavy design/development on a project. At this stage, I'm usually trying out several different designs, looking for the right approach. And sometimes the right approach is one I had already attempted earlier. So it's nice if that code isn't lost in the depths of source control. But once the design has been settled, I'll get rid of the old code.

Share

answered Nov 11, 2008 at 16:57

community wiki
JW.

2

votes

In general I tend to comment very sparsely. I believe good code should be easy to read without much commenting.

I also version my code. I suppose I could do diffs over the last twenty checkins to see if a particular line has changed for a particular reason. But that would be a huge waste of my time for most changes.

So I try comment my code smartly. If some code is being deleted for a fairly obvious reason, I won't bother to comment the deletion. But if a piece of code is being deleted for a subtle reason (for example it performed a function that is now being handled by a different thread) I will comment-out or delete the code and add a banner comment why:

```
    // this is now handled by the heartbeat thread
    // m_data.resort(m_ascending);
```

Or:

```
    // don't re-sort here, as it is now handled by
the heartbeat thread
```

Just last month, I encountered a piece of code that I had changed a year ago to fix a particular issue, but didn't add a comment explaining why. Here is the original code:

```
    cutoff = m_previous_cutofftime;
```

And here is the code as it was initially fixed to use a correct cutoff time when resuming an interrupted state:

```
   cutoff = (!ok_during) ? m_previous_cutofftime :
0;
```

Of course another unrelated issue came up, which happened to touch the same line of code, in this case reverting it back to its original state. So the new issue was now fixed, but the old issue suddenly became rebroken. D'oh!

So now the checked-in code looks like this:

```
   // this works for overlong events but not
resuming
// cutoff = m_previous_cutofftime;
   // this works for resuming but not overlong
events
// cutoff = (!ok_during) ? m_previous_cutofftime :
0;
   // this works for both
   cutoff = (!resuming || !ok_during) ?
m_previous_cutofftime : 0;
```

Of course, YMMV.

Share

answered Nov 11, 2008 at 17:21

community wiki
CompaniaHhill

2 votes

As the lone dissenting voice, I will say that there is a place for commenting out code in special circumstances. Sometimes, you'll have data that continues to exist that was run through that old code and the clearest thing to do is to leave that old code in with source. In such a case I'd probably leave little note indicating why the old code was simply commented out. Any programmers coming along after would be able to understand the still extant data, without having to psychically detect the need to check old versions.

Usually though, I find commented out code completely odious and I often delete it when I come across it.

Share

answered Nov 11, 2008 at 18:49

community wiki
DaveB

---

1 vote

If you are removing code. You should not comment it that you removed it. This is the entire purpose of source control (You are using source control? Right?), and as you state the comment just clutters up the code.

Share

answered Nov 11, 2008 at 15:26

community wiki

grieve

1 vote

I agree that it's a terrible suggestion. That's why you have Source Control that has revisions. If you need to go back and see what was changed between two revisions, diff the two revisions.

Share

answered Nov 11, 2008 at 15:26

community wiki
LeppyR64

1 vote

I hate seeing code that's cluttered with commented out code. Delete the code and write a commit message that says why it was removed. You do use source control, don't you?

Don't litter active code with dead code.

Share

answered Nov 11, 2008 at 15:27

community wiki
John Topley

1 vote

I'll add my voice to the consensus: put the comments on why code was deleted in the source control repository, not in the code.

Share

answered Nov 11, 2008 at 15:27

community wiki
David Arno

---

1 vote

This is one of those "broken" windows thinkgs like compiler hints/warnings left unaddressed. it will hurt you one day and it promotes sloppiness in the team.

The check in comment in version control can track what/why this code was removed - if the developer didnt leave a note, track them down and throttle them.

Share

answered Nov 11, 2008 at 17:53

community wiki
MikeJ

---

1 vote

A little anecdote, for fun: I was in a company, some years ago, knowing nothing of source code version control (they got such tool later...).
So they had a rule, in our C sources: "when you make a change, disable the old code with preprocessor macros":

```
#ifdef OLD /* PL - 11/10/1989 */
void Buggy()
{
```

```
// ...
}
#else
void Good()
{
// ...
}
#end
```

No need to say, our sources quickly became unreadable! It was a nightmare to maintain...
That's why I added to SciTE the capacity to jump between nested #ifdef / #else / #end and such... It can be still useful in more regular cases.
Later, I wrote a Visual Studio macro to happily get rid of old code, once we got our VCS!

Now, like buti-oxa, sometime I felt the need to indicate why I removed some code. For the same reason, or because I remove old code which I feel is no longer needed, but I am not too sure (legacy, legacy...). Obviously not in all cases!
I don't leave such comment, actually, but I can understand the need.
At worse, I would comment out in one version, and remove everything in the next version...
At my current work, for important local changes, we leave the old code but can reactivate it by properties, in case of emergency. After testing it some time in production, we eventually remove the old code.

Of course, VCS comments are the best option, but when the change is a few lines in a big file with other changes, referencing the little removal can be hard...

Share

answered Nov 25, 2008 at 15:08

community wiki
PhiLho

---

1 vote

If you are in the middle of major changes, and need to make a fix to existing functionality, commenting out the future code is a reasonable thing to do, provided you remark that this is future functionality, at least until we have futures friendly source control systems.

Share

answered Dec 2, 2008 at 12:24

community wiki
John with waffle

---

0 votes

I comment unnused code because you never know when will you have to fallback on the ancient code, and maybe the old code will help other people to understand it, if it was simpler back then.

Share

answered Nov 11, 2008 at 15:26

community wiki
LuRsT

0

votes

I agree with you Andrew; IMO this is why you use version control. With good checkin/commit comments and a diff tool you can always find out why lines were removed.

Share

answered Nov 11, 2008 at 15:27

community wiki
Patrick Cuff

0

votes

If you are using any form of Source Control then this approach is somewhat redundant (as long as descriptive log messages are used)

Share

answered Nov 11, 2008 at 15:27

community wiki
DilbertDave

0

votes

I also think it's a terrible suggestion :)

You should use source control and if you remove some code you can add a comment when you commit. So you still have the code history if you want...

Share

edited Nov 11, 2008 at 16:28

community wiki
2 revs, 2 users 80%
pgras

0
votes

There's a general "clean code" practice that says that one should never keep removed code around as commented out since it clutters and since your CVS/SVN would archive it anyway.

While I do agree with the principle I do not think that it is an acceptable approach for all development situations. In my experience very few people keep track of all the changes in the code and every check-in. as a result, if there is no commented out code, they may never be aware that it has ever existed.

Commenting code out like that could be a way of offering a general warning that it is about to be removed, but of course, there are no guarantees that interested parties would ever see that warning (though if they frequently work with that file, they will see it).

I personally believe that the correct approach is to factor that code out to another private method, and then contact relevant stakeholders and notify them of the pending removal before actually getting rid of the function.

Share

answered Nov 11, 2008 at 18:54

community wiki
Uri

0
votes

Where I am at we comment out old code for one release cycle and then remove the comments after that. (It gives us quick fix ability if some of the new code is problematic and needs to be replaced with the old code.)

Share

answered Nov 11, 2008 at 22:20

community wiki
The Sasquatch

0
votes

In almost all cases old code should of course be removed and tracked in your RCS.

Like all things though, I think that making the statement 'All deleted code will ALWAYS be removed' is an incorrect approach.

The old code might want to be left in for a miriad of reasons. The prime reason to leave the code in is when you want any developer who is working in that section of code in the future to see the old code.

Relying on source tracking obviously does not give this.

So, I believe the correct answer is:

-Delete old code unless leaving it in provides crucial information that the next developer in the code would require. Ie, remove it 99% of the time but don't make a draconian rule that would remove your ability to provide much needed documentation to the next developer when circumstances warrant it.

Share

answered Nov 25, 2008 at 15:29

community wiki
Troy