How do you flag code so that you can come back later and work on it?

Asked 10 years, 7 months ago Active 1 year, 11 months ago Viewed 42k times

60

9

In C# I use the `#warning` and `#error` directives,

```
#warning This is dirty code...
#error Fix this before everything explodes!
```

This way, the compiler will let me know that I still have work to do. What technique do you use to mark code so you won't forget about it?

c# comments

edited May 3 '15 at 2:37

community wiki
5 revs, 4 users 63%
Jon Tackabury

21 Answers

84

Mark them with `// TODO` , `// HACK` or other comment tokens that will show up in the task pane in Visual Studio.

See Using the Task List.

edited Jul 31 '17 at 7:39

community wiki
3 revs, 3 users 50%
Wouter Huysentruit

2 # @todo: in Python – S.Lott Dec 2 '08 at 20:44

3 I used to do the //TODO: as well, but sometimes I would forget to check the task pane. – Jon Tackabury Dec 2 '08 at 20:45

2 @Jon T: how about a throw new NotImplementedException(). Would that help you? I sometimes do that too. – Guge Dec 2 '08 at 20:47

1 TODO comes up with a nasty brown background in vim - visual code smells – Ken Dec 2 '08 at 21:09

@S.Lott: any particular reason why you use @todo, instead of the typical TODO? (i'm just curious) – Jeremy Cantrell Dec 2 '08 at 21:20

26

Todo comment as well.

We've also added a special keyword NOCHECKIN, we've added a commit-hook to our source control system (very easy to do with at least cvs or svn) where it scans all files and refuses to check in the file if it finds the text NOCHECKIN anywhere.

This is very useful if you just want to test something out and be certain that it doesn't accidentaly gets checked in (passed the watchful eyes during the diff of everything thats commited to source control).

answered Dec 2 '08 at 20:48 community wiki Ulf Lindback

14

I use a combination of `//TODO:` `//HACK:` and `throw new NotImplementedException();` on my methods to denote work that was not done. Also, I add bookmarks in Visual Studio on lines that are incomplete.

edited Aug 22 '14 at 23:04 community wiki 2 revs, 2 users 67% Chris Lees

10

//TODO: Person's name - please fix this.

This is in Java, you can then look at tasks in Eclipse which will locate all references to this tag, and can group them by person so that you can assign a TODO to someone else, or only look at your own.

answered Dec 2 '08 at 20:46 community wiki Elie

That's a cool idea - I've never thought of assigning things ad hoc in the code. – Jon Tackabury Dec 2 '08 at 20:49

Thanks, we use it quite heavily where I work as a fast way of marking code for other people so that they don't have to search for it. – Elie Dec 2 '08 at

20:52

We've done this but created custom tags for everyone so it's just //NAME: blah blah blah and we share Eclipse configurations – Instantsoup Feb 10 '09 at 17:01

6

'To do' comments are great in theory, but not so good in practice, at least in my experience. If you are going to be pulled away for long enough to need them, then they tend to get forgotten.

I favor Jon T's general strategy, but I usually do it by just plain breaking the code temporarily - I often insert a deliberately undefined method reference and let the compiler remind me about what I need to get back to:

```
PutTheUpdateCodeHere();
```

answered Dec 2 '08 at 20:48 community wiki McKenzieG1

6

If I've got to drop everything in the middle of a change, then

```
#error finish this
```

If it's something I should do later, it goes into my bug tracker (which is used for all tasks).

edited Dec 2 '08 at 21:23 community wiki 2 revs John MacIntyre

---

5

Add a test in a disabled state. They show up in all the build reports.

If that doesn't work, I file a bug.

In particular, I haven't seen TODO comments ever decrease in quantity in any meaningful way. If I didn't have time to do it when I wrote the comment, I don't know why I'd have time later.

answered Dec 2 '08 at 20:46

community wiki
Dustin

5

An approach that I've really liked is "Hack Bombing", as demonstrated by Oren Eini here.

```
try
{
   //do stuff
   return true;
}
catch // no idea how to prevent an exception here at the moment, this make it work for
now...
{
  if (DateTime.Today > new DateTime(2007, 2, 7))
    throw new InvalidOperationException("fix me already!! no catching exceptions like
this!");
  return false;
}
```

answered Dec 2 '08 at 20:53

community wiki
idan315

5 +1 For humour value, even though this is absolutely horrible! – Dan J Dec 8 '09 at 21:07

4

```
//TODO: Finish this
```

If you use VS you can setup your own Task Tags under Tools>Options>Environment>Task List

answered Dec 2 '08 at 23:09

community wiki
Brian Rudolph

---

gvim highlights both "// XXX" and "// TODO" in yellow, which amazed me the first time I marked some code that way to remind myself to

3

come back to it.

answered Dec 2 '08 at 20:48

community wiki
Paul Tomblin

2

I use // TODO: or // HACK: as a reminder that something is unfinished with a note explaining why. I often (read 'rarely') go back and finish those things due to time constraints. However, when I'm looking over the code I have a record of what was left uncompleted and more importantly WHY.

One more comment I use often at the end of the day or week:

// START HERE CHRIS

^^^^^^^^^^^^^^^^^^ Tells me where I left off so I can minimize my bootstrap time on Monday morning.

answered Dec 2 '08 at 21:27

community wiki
Chris Nava

2

It's not a perfect world, and we don't always have infinite time to refactor or ponder the code.

I sometimes put `//REVIEW` in the code if it's something I want to come back to later. i.e. code is working, but perhaps not convinced it's the best way.

```
// REVIEW - RP - Is this the best way to achieve x? Could we use algorithm y?
```

Same goes for `//REFACTOR`

```
// REFACTOR - should pull this method up and remove near-dupe code in XYZ.cs
```

answered Jun 22 '10 at 5:07

community wiki
Robert Paulson

1

Todo Comment.

answered Dec 2 '08 at 20:44 community wiki GurdeepS

1

```
// TODO: <explanation>
```

if it's something that I haven't gotten around to implementing, and don't want to forget.

```
// FIXME: <explanation>
```

if it's something that I don't think works right, and want to come back later or have other eyes look at it.

Never thought of the #error/#warning options. Those could come in handy too.

answered Dec 2 '08 at 21:38 community wiki GalacticCowboy

1

I use //FIXME: xxx for broken code, and //CHGME: xxx for code that needs attention but works (perhaps only in a limited context).

answered Dec 3 '08 at 5:45 community wiki Lawrence Dol

1

These are the three different ways I have found helpful to flag something that needs to be addressed.

1. Place a comment flag next to the code that needs to be looked at. Most compilers can recognize common flags and display them in an organized fashion. Usually your IDE has a watch window specifically designed for these flags. The most common comment flag is: //TODO This how you would use it:

//TODO: Fix this before it is released. This causes an access violation because it is using memory that isn't created yet.

2. One way to flag something that needs to be addressed before release would be to create a useless variable. Most compilers will warn you if you have a variable that isn't used. Here is how you could use this technique:

   int This_Is_An_Access_Violation = 0;

3. IDE Bookmarks. Most products will come with a way to place a bookmark in your code for future reference. This is a good idea, except that it can only be seen by you. When you share your code most IDE's won't share your bookmarks. You can check the help file system of your IDE to see how to use it's bookmarking features.

answered Dec 3 '08 at 15:07 community wiki Jeremiah

1

If it's some long term technical debt, you can comment like:

```
// TODO: This code loan causes an annual interest rate of 7.5% developer/hour. Upfront fee as stated by the current implementation. This contract is subject of prior authorization from the DCB (Developer's Code Bank), and tariff may change without warning.
```

... err. I guess a TODO will do it, as long as you don't simply ignore them.

answered Jun 22 '10 at 4:44 community wiki Chubas

1

I'm a C++ programmer, but I imagine my technique could be easily implemented in C# or any other language for that matter:

I have a `ToDo(msg)` macro that expands into constructing a static object at local scope whose constructor outputs a log message. That way, the first time I execute unfinished code, I get a reminder in my log output that tells me that I can defer the task no longer.

It looks like this:

```
class ToDo_helper
{
  public:
     ToDo_helper(const std::string& msg, const char* file, int line)
     {
       std::string header(79, '*');
```

```
        Log(LOG_WARNING) << header << '\n'
                         << "  TO DO:\n"
                         << "    Task:  " << msg << '\n'
                         << "    File:  " << file << '\n'
                         << "    Line:  " << line << '\n'
                         << header;
    }
};

#define TODO_HELPER_2(X, file, line) \
  static Error::ToDo_helper tdh##line(X, file, line)

#define TODO_HELPER_1(X, file, line) TODO_HELPER_2(X, file, line)
#define ToDo(X) TODO_HELPER_1(X, __FILE__, __LINE__)
```

... and you use it like this:

```
void some_unfinished_business() {
  ToDo("Take care of unfinished business");
}
```

answered Jun 22 '10 at 4:58 community wiki Drew Hall

Wow, that's nice Mister – Jeancarlo Fontalvo Sep 18 '16 at 18:24

0

I also use TODO: comments. I understand the criticism that they rarely actually get fixed, and that they'd be better off reported as bugs. However, I think that misses a couple points:

- I use them most during heavy development, when I'm constantly refactoring and redesigning things. So I'm looking at them all the time. In situations like that, most of them actually do get addressed. Plus it's easy to do a search for TODO: to make sure I didn't miss anything.
- It can be very helpful for people reading your code, to know the spots that you think were poorly written or hacked together. If I'm reading unfamiliar code, I tend to look for organizational patterns, naming conventions, consistent logic, etc.. If that consistency had to be violated one or two times for expediency, I'd rather see a note to that effect. That way I don't waste time trying to find logic where there is none.

answered Jan 7 '09 at 18:12 community wiki

JW.

0

As most programmers seem to do here, I use TODO comments. Additionally, I use Eclipse's task interface Mylyn. When a task is active, Mylyn remembers all resources I have opened. This way I can track

1. where in a file I have to do something (and what),
2. in which files I have to do it, and
3. to what task they are related.

answered Feb 10 '09 at 14:04 community wiki migu

0

Besides keying off the "TODO:" comment, many IDE's also key off the "TASK:" comment. Some IDE's even let you configure your own special identifier.

answered Feb 10 '09 at 16:55 community wiki Kurt W. Leucht

protected by George Stocker ♦ Mar 14 '11 at 19:51

Thank you for your interest in this question. Because it has attracted low-quality or spam answers that had to be removed, posting an answer now requires 10 reputation on this site (the association bonus does not count).

Would you like to answer one of these unanswered questions instead?