How do you write code that is easily read by other people who have had no hand in writing any part of it?

Asked 16 years, 3 months ago Modified 10 years, 10 months ago

Viewed 1k times

How do you write code that is easily read by other people and who have had no hand in writing any part of it?

5

readability

Share

Improve this question

Follow

edited Dec 13, 2011 at 13:38

Erick Robertson

33k ● 13 ● 71 ● 98

asked Sep 2, 2008 at 12:29

Codeslayer

3,393 ● 7 ● 37 ● 42

13 Answers

Sorted by: Highest score (default)

The best way to ensure that others can read your code is to make sure that it is clear and concise. Namely,

9

- Use self documenting names for variables, functions, and classes.
- Comment complex algorithms so that the reader doesn't have to spend to long figuring out what it does.
- Ensure that tabbing and line breaks are constant throughout the code.

Beyond that you start to get in to the areas that might be a bit subjective, most people should agree on these items.

Share Improve this answer

Follow

answered Sep 2, 2008 at 12:34

rjzii

14.5k ● 12 ● 81 ● 122

6

You may want to take a look at Clean Code by Robert C. Martin. It offers up a lot of useful practices for ensuring your code is readable.

Additionally, if your code is supported by a number of unit tests that thoroughly test your code, it offers a way for your user to understand the code by looking at what the tests are doing. You will also find that if you follow the Test Driven Development process, and you write tests for each bit of functionality, your functions tend to be small, do one thing only and do it well, and tend to flow more like a story than simply a large complex web of "stuff".

Tests tend to stay up-to-date more than comments. I often ignore comments anymore due to simple fact that they become obsolete very quickly.

Share Improve this answer Follow

edited Jan 6, 2009 at 18:51

answered Jan 6, 2009 at 17:03

borq

681 ● 7 ● 16

6

This question is subjective, and should be avoided on StackOverflow, as per the FAQ

> **What kind of questions should I not ask here?**
>
> Avoid asking questions that are **subjective**, argumentative, or require extended discussion. This is a place for questions that can be answered!

The short answer would be:

- Avoid excessive commenting:

```
// add one to the count:
i++;
```

- Use good variable and method names:

```
int x = i + j;
int runSum = prevSum += newValue;
```

- Use coding shorthand where available:

```
if (x == y)
{
  z = a;
}
else
{
  z = b;
}
z = (x == y) ? a : b;
```

Share Improve this answer

Follow

edited May 23, 2017 at 10:32

Community Bot

1 ● 1

answered Sep 2, 2008 at 12:39

Rob Cooper

28.9k ● 26 ● 105 ● 142

1 You'll want to change z == a & z == b to z = a and z = b. – mbillard Jan 6, 2009 at 16:28

1 -1: You began your answer by arrogantly asserting SO policy. Then, as @Crossbrowser pointed out, you posted an illogical code snippet. – Jim G. May 29, 2010 at 11:41

2 Hello again Jim, are you just trolling my profile? Thanks - I appreciate the attention! :) There's nothing arrogrant about trying to help build a community so we can all work together. And secondly, the code wasn't illogical, it was incorrect! Sadly, with my ever-so-low programming skills, I hit "=" twice for assessment rather than assignment. How will I ever cope! Especially since no one can read it and get the point of what I

was trying to say. I will do the honorable thing and hand in my notice at work now. Thanks to @Crossbrowser for pointing it out in the first instance :) – Rob Cooper Jun 4, 2010 at 7:18

Not a bad answer but I'm not so sure about the last part. I believe both syntax's are just as easy to read as each other. – Ash Burlaczenko Dec 10, 2010 at 16:44

4

Keep code nice, clear and simple. Don't comment what you're doing when it's obvious (for instance I know what a foreach or if does, I don't normally need an explanation).

Code tricks (such as auto properties) that make simple things take up fewer lines are good too.

Share Improve this answer

Follow

answered Sep 2, 2008 at 12:33

Keith

155k ● 82 ● 306 ● 446

4

Buy & read Code Complete 2. There's loads of stuff in there about writing easy to read / maintain code.

Share Improve this answer

Follow

answered Sep 2, 2008 at 12:44

Robin Barnes

13.5k ● 15 ● 45 ● 45

I don't think it's a subjective question, but it's too broad! It's not just about commenting and giving good variables

2

names. It deals with how humans comprehends code. So your system must be implemented in a way that the reader can easily construct a mental model of its design in two way:

- Top-down: assuming the user knows the system domain, he tends to make assumptions on how it would be implemented, so he'll scan the system packages and classes looking for entities he can identify. Giving good names to your classes and properly modularizing it would help very much.
- Bottom-up: once the user reaches a portion of code he'll start navigation from there, building chunks of knowledge. If your system has low cohesion and lots of implicit dependencies the user will be lost.

Kent Beck adopts three principles: Communication, Simplicity and Flexibility. Of course, sometimes you'll have to trade simplicity for flexibility, and vice-versa.

This could go on and on. The answer to this question fits in a large book. As @rmbarnes suggested, buy and read Code Complete 2. I also suggest Implementation Patterns by Kent Beck - its highly related to your question.

Share Improve this answer

Follow

edited Jan 6, 2009 at 16:37

Darron

21.6k ● 5 ● 51 ● 54

answered Sep 2, 2008 at 13:10

Marcio Aguiar
14.5k ● 6 ● 40 ● 42

1. Document the code as to why it does what it does.
2. Make sure that all variables functions etc. are named consistently and descriptively
3. Use white space to group logical portions of code together, so it flows while reading.
4. Place the functions/methods etc. in a logical order.
5. (this one is my personal preference) Make sure that code can easily be read on the screen without having to scroll horizontally (some people say vertically too, but this doesn't seem to bother me).

Share Improve this answer

Follow

answered Sep 2, 2008 at 12:37

kemiller2002
115k ● 28 ● 199 ● 253

-1: *1.Document the code as to why it does what it does.* - Good luck keeping that documentation up-to-date. *some people say vertically too, but this doesn't seem to bother me* - So long functions and methods don't bother you? – Jim G. May 29, 2010 at 11:39

1 1. My method names are not paragraphs, so why would it bother me? I have the IDE to auto-populate method names for me. I'll take longer names any day of the week to code which is confusing. Where I work, we take time to make sure that all the code we write is documented and reviewed, so yeah, it's pretty much always up to date. If it's not, it's noted in the review and the programmer goes back and fixes it, and this was decided by the programmers not forced down by

managers, so everyone follows it, because we know it's importance. I take it you don't do that? – kemiller2002 May 29, 2010 at 13:00

1

Since everyone else said pretty much what I'm thinking when I read this question, I'll just share two books related to this subject that you might be interested in reading. These books use open source code examples to explain how to read and write high quality code. In addition to Code Complete, I think they are valuable resources when you want to write good code in any language.

- Code Reading: The Open Source Perspective
- Code Quality: The Open Source Perspective

Share Improve this answer Follow

answered Sep 2, 2008 at 13:26

Thomas Owens
116k ● 99 ● 317 ● 436

1

My rules:

1. Give everything a meaningful name, and call it what it is. Avoid using "x" and "y" for variables.
2. Don't abbreviate ANYTHING. I don't care how long the variable name is, don't abbreviate, even with comments. Interpretation of abbreviations is subjective. Does Cmp mean computer? Computer? Company? Compliment? Make it a strong rule, no exceptions, and its easy to follow.

3. Don't put multiple statements on the same line. Each line performs a single action.
4. Avoid Hungarian Notation like the plague. Or is it ntHungarian?
5. Use brackets even for single-line (if, for) substructures. Indentation differences are too easy to lose.

Share Improve this answer

Follow

answered Jan 6, 2009 at 16:22

Tom Moseley

591 ● 7 ● 15

1

A lot of good answers here, I would like to add something from the perspective of an engineer who likes the big picture. I frequently found that getting a high level overview, in terms of class diagram or a package level overview (diagram/comments etc), heck if nothing exists a 10 line header comments in a file to help me a lot. We can use Doxygen/Javadocs to generate them, or spend 10-15 minutes to just jot down something in comments section.

They dont have to be 100% accurate, and I doubt the overall structure of classes/packages will change without a complete rewrite.

I personally found this kind of big picture overview very helpful and am sure there are others who feel the same.

Share Improve this answer

answered Jan 6, 2009 at 17:16

Follow

omermuhammed
7,375 ● 4 ● 29 ● 40

0

Probably the most important point is to keep your syntax consistent. I would also have a look at the design guidelines for the language you are writing in.

Share Improve this answer

Follow

answered Sep 2, 2008 at 12:33

Christian Hagelid
8,335 ● 4 ● 41 ● 63

0

I am most likely in the minority, but I don't mind whitespace. I LOVE WHITESPACE. Since the compiler takes it out and HD space being so cheap I like to have white space in my code.

For example I like:

```
        int total = 10;
        int sum = 0;

        for (int i = 0; i < total; i++)
        {
            sum += i;
        }

        // Next coding statement is a space below
the bracket
        return sum;
```

I do not like:

```
        int total = 10;int sum = 0;
        for (int i = 0; i < total; i++)
        {
            sum += i;
        }
        return sum;
```

What I also put in Brackets even though technically they are not needed. The best example is the if statement. I find it greatly helps readability.

```
if(true)
    // some action

if(true)
{
   // Some action
}
```

The best code to me, is one that as simple as possible. With the least comments as possible, and most importantly ***works***.

Share Improve this answer

Follow

answered Sep 2, 2008 at 14:22

David Basarab

73.2k ● 43 ● 130 ● 157

0

From being a developer with several years under the belt, this used to be a real question for me. I couldn't even say how many hours I passed thinking about this and trying different things in my code. The above answers are very nice too. I just want to add a thing or two.

- We each have different things that make our reading different than the others. Something that you find easy to read, might really be hard for somebody else to read.
- Cleanliness of your code is a very important aspect. Soon as it gets too cramped just forget about it.
- Most important: You are you own teacher. No matter what style you follow, you will want to change a thing or two based on your experience. As months pass and you have to go back to your old for fixes or documentation, you will have the "I can't believe I wrote code that reads like that" effect. Take notes of what was bugging you with the code readability and make sure not to write like that again.

Share Improve this answer

Follow

edited Feb 3, 2014 at 12:58

BenMorel

36.3k ● 51 ● 202 ● 334

answered Sep 2, 2008 at 13:17

Dewm Solo

691 ● 2 ● 7 ● 13