13 Shares

DEVELOPMENT

Code Review: What Is It and Why Is It Important?

AGILE DEVELOPMENT CODE CODE REVIEW DEVELOPMENT EXTERNAL REVIEW MOZILLA PEER REVIEW

Darío Macchi Sep 22 Share on

New to the concept of code review? This post explains what code review is and why it's important.

Disclaimer: The following document is heavily based on the Mozilla Code Review FAQ [1][2]. But my team at VAIRIX has made many adaptations in order to reflect the two-level review process that is part of our development methodology.

What is code review?

As Wikipedia puts it, "Code review is systematic examination … of computer source code. It is intended to find and fix mistakes overlooked in the initial development phase, improving both the overall quality of software and the developers' skills."

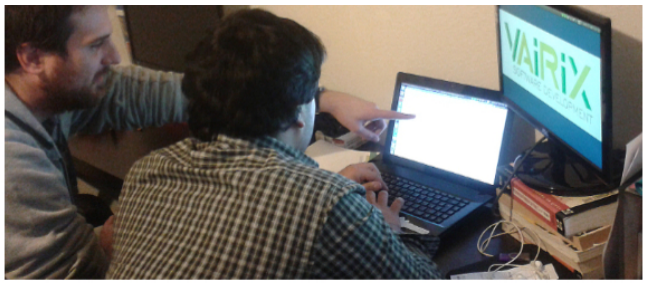
What is the purpose of code review?

Code review is the most commonly used procedure for validating the design and implementation of features. It helps developers to maintain consistency between design and implementation "styles" across many team members and between various projects on which the company is working.

We perform code review in two levels. The first is known as peer review and the second is external review.

The code review process doesn't begin working instantaneously (especially with external review), and our process is far from being perfect — although we have done some serious research around the topic [3]. So, we are always open to suggestions for improvement. 🙂

Having said that, let's dig into peer reviews.



What is a peer review?

A peer review is mainly focused on functionality, design, and the implementation and usefulness of proposed fixes for stated problems.

The peer reviewer should be someone with business knowledge in the problem area. Also, he or she may use other areas of expertise to make comments or suggest possible improvements.

In our company, this is necessary because we don't do design reviews prior to code reviews. Instead, we expect developers to talk to each other about their design intentions and get feedback throughout the (usually non-linear) design/implementation process.

Accordingly, we don't put limitations on what comments a reviewer might make about the reviewed code.

What do peer reviewers look for?

Feature Completion

The reviewer will make sure that the code meets the requirements, pointing out if something has been left out or has been done without asking the client.

Potential Side Effects

The reviewer will check to see whether the changed code causes any issues in other features.

Readability and Maintenance

The reviewer will make sure the code is readable and is not too complicated for someone completely new to the project. Model and variable names should be immediately obvious (again, even to new developers) and as short as possible without using abbreviations.

Consistency

Conducting peer reviews is the best approach for achieving consistency across all company projects. Define a code style with the team and then stick to it.

Performance

The reviewer will assess whether code that will be executed more often (or the most critical functionalities) can be optimized.

Exception Handling

The reviewer will make sure bad inputs and exceptions are handled in the way that was pre-defined by the team (it must be visible/accessible to everyone).

Simplicity

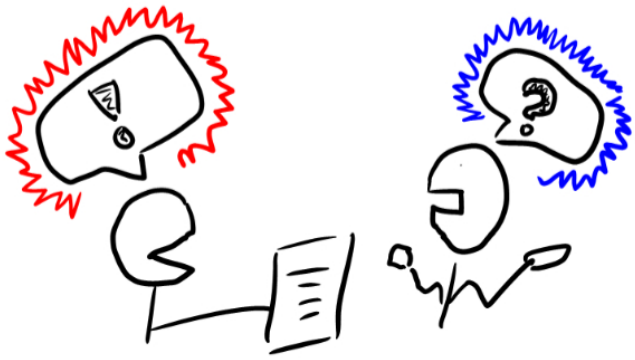
The reviewer will assess whether there are any simpler or more elegant alternatives available.

Reuse of Existing Code

The reviewer will check to see if the functionality can be implemented using some of the existing code. Code has to be aggressively “DRYed” (as in, Don’t Repeat Yourself) during development.

Test Cases

Finally, the reviewer will ensure the presence of enough test cases to go through all the possible execution paths. All tests have to pass before the code can be merged into the shared repository.



What is an external review?

An external review addresses different issues than peer reviews. Specifically, external reviews focus on how to increase code quality, promote best practices, and remove “code smells.”

This level of review will look at the quality of the code itself, its potential effects on other areas of the project, and its adherence with company coding guidelines.

Although external reviewers may not have domain expertise, they do have discretion to raise red flags related to both the design and code and to suggest ways to solve problems and refactor code as necessary.

What do external reviewers look for?

Readability and Maintenance

Similar to above, the reviewer will make sure the code is readable and is not too complicated for someone completely new. Again, all model and variable names have to be immediately obvious (even to new developers) and as short as possible without using abbreviations.

Coding Style

The reviewer will ensure that everyone adheres to a strict coding style and will use code editors’ built-in helpers to format the code.

Code Smells

Finally, the reviewer will keep an eye out (or should that be a nose out?) for code smells and make suggestions for how to avoid them.

In case the term is new to you, a code smell is “a hint that something has gone wrong somewhere in your code. Use the smell to track down the problem.”

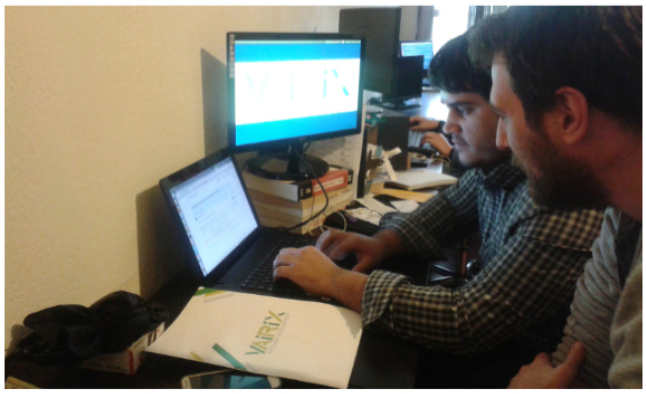
Must external reviewers be “domain

experts”?

External reviewers don’t have to have domain knowledge of the code that they will be reviewing. [4].

If they know about the domain, they will feel tempted to review it at a functional level, which could lead to burnout. However, if they have some business knowledge, they can estimate more easily how complex the review will be and can quickly complete the review, providing a more comprehensive evaluation of the code.

So, domain expertise is a bonus, not a requirement.



What if an external reviewer misses something?

We do not expect an external reviewer to make everything perfect. Something will most likely be missed. The external reviewer does not become responsible for the developer’s work by reviewing it.

How fast should developers receive a response from the external reviewer?

If a developer has requested an external review, he can expect some type of response within two hours. At the very least, the response should tell him a timeframe for completion.

In some cases, the external reviewers might not respond. They’re not perfect and might have too much work to do. Developers should feel free to ping them again if they don’t hear back within two hours or try with another external reviewer.

Why can’t developers simply merge their code into the main branch now and ask for an external review later?

There are many reasons this is a bad idea, but here are two of the most important:

1. External reviews catch problems that would affect everyone if the code were merged into the main repository. It doesn’t make sense to cause everyone to suffer for problems that could have been caught by an external review.
2. The process of merging code causes the developer to feel that the work is done, and it’s time to go on to the next thing. It’s silly to have people feeling like something is checked off the task list when it’s really not.

Can the external reviewer ask the developer to do something that is not precisely related to the code?

Yes, the external reviewer has some discretion here.

We don't think that continuously making auxiliary changes that are unrelated to the core functionality is the right thing to do on reviews. On the other hand, small changes (or changes that help the code maintain a consistent style) may be requested.

There should be a reasonable relationship between the scope of the developed functionality and the scope of the requested change.

References

[1] Knous, M. & Dbaron, A. (2005). Code Review FAQ. Mozilla Development Network. Retrieved from https://developer.mozilla.org/en/docs/Code_Review_FAQ.

[2] Rigby, C., German, D. (2006). "A preliminary examination of code review processes in open source projects." University of Victoria Technical Report: DCS-305-IR. Retrieved from http://ifipwg213.org/system/files/Rigby2006TR.pdf.

[3] Macchi, D., & Solari, M. (2012). Software inspection adoption: A mapping study. In Conferencia Latinoamericana de Informática (CLEI 2012). http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=6427197.

[4] Mozilla (2012). Retrieved from http://www.mozilla.org/hacking/reviewers.html.

Feature image credit. All other images were provided by the author.

PREVIOUS ARTICLE

The Essential Guide to Building an Activity Stream into Your App

NEXT ARTICLE

How Deep Linking Leads to More App Downloads and Higher Retention

Ready to start your project?

Learn how ThinkApps can get your product launched faster, better, and with more value than you knew was possible.

Click here

ABOUT THE AUTHOR

Darío Macchi

Darío Macchi is the COO/Scrum Master of VAIRIX Software Development, a boutique Ruby on Rails development company from Uruguay. Also he is a Software Engineering Professor, researcher (MSc, starting his PhD) and independent advisor/consultant. He believes in the usefulness of bringing tools from scientific research environments to daily activities of software development companies.

Twitter LinkedIn

@THINKAPPS ON TWITTER

22 Oct
FREE this weekend: Award winning book,"How to Start a Startup" on Amazon kindle. Snag your free copy today! https://t.co/YmEkcYKp3l

21 Oct
FREE this weekend: Award winning book,"How to Start a Startup" on Amazon kindle. Snag your free copy today! https://t.co/YmEkcYKp3l

20 Oct
Our new book, "How to Start a Startup," is the ultimate reference guide on tech startups. https://t.co/YmEkcYKp3l

buildblog

About us

Since 2012, leading companies have put their trust in us to build their products. We're on a mission to change the way companies design, develop, and launch software products. Our unique solution gives companies access to the world's best design and development and helps take them from idea to launch.

Contact us

ThinkApps
1211 Folsom Street, Floor 3
San Francisco
CA 94103
1-415-742-2499

Copyright ThinkApps © 2014. All rights reserved.

Top