# Why code reviews matter (and actually save time!)

Spoiler alert: if you love sound architectural decisions and hate being a "critical path" developer, you're gonna love this.

BY DAN RADIGAN

Agile teams are self-organizing, with skill sets that span across the team. This is accomplished, in part, with code review. Code review helps developers learn the code base, as well as help them learn new technologies and techniques that grow their skill sets.

## So, what exactly is a code review?

When a developer is finished working on an issue, another developer looks over the code and considers questions like:

- Are there any obvious logic errors in the code?
- Looking at the requirements, are all cases fully implemented?
- Are the new automated tests sufficient for the new code? Do existing automated tests need to be rewritten to account for changes in the code?
- Does the new code conform to existing style guidelines?

Code reviews should integrate with a team's existing process. For example, if a team is using task branching workflows, initiate a code review after all the code has been written and automated tests have been run and passed–but before the code is merged upstream. This ensures the code reviewer's time is spent checking for things machines miss, and prevents poor coding decisions from polluting the main line of development.

## What's in it for an agile team?

Every team can benefit from code reviews regardless of development methodology. Agile teams, however, can realize huge benefits because work is decentralized across the team. No one is the *only* person who knows a specific part of the code base. Simply put, code reviews help facilitate knowledge sharing across the code base and across the team.

### Code reviews share knowledge

At the heart of all agile teams is unbeatable flexibility: an ability to take work off the backlog and begin execution by *all* team members. As a result, teams are better able to swarm around new work because no one is the "critical path." Full stack engineers can tackle front-end work as well as server-side work.

> As code reviews expose developers to new ideas and technologies, they write better and better code.

### Code reviews make for better estimates

Remember the section on estimation? Estimation is a team exercise, and the team makes better estimates as product knowledge is spread across the team. As new features are added to the existing code, the original developer can provide good feedback and estimation. In addition, any code reviewer is also exposed to the complexity, known issues, and concerns of that area of the code base. The code reviewer, then, shares in the knowledge of the original developer of that part of the code base. This practice creates multiple, informed inputs which, when used for a final estimate always makes that estimate stronger and reliable.

### Code reviews enable time off

Nobody likes to be the sole point of contact on a piece of code. Likewise, nobody wants to dive into a critical piece of code they didn't write–especially during a production

RELATED TUTORIAL

Learn about code review in Bitbucket Cloud

Try this tutorial →

SUBSCRIBE

Sign up for more articles

Email

email@example.com

Subscribe

code they didn't write–*especially during a production emergency.* Code reviews share knowledge across the team so that any team member can take up the reins and continue steering the ship. (We love mixed metaphors at Atlassian!) But here's the point: with no single developer the critical path, it also means team members can take time off as needed. If you find yourself tied to a desk on the version control system, code review is an excellent way to find freedom. Freedom to take that needed vacation, or freedom to spend some time working on a different area of the product.

### Code reviews mentor newer engineers

A special aspect of agile is that when new members join the team more seasoned engineers mentor the newer members. And code review helps facilitate conversations about the code base. Often, teams have hidden knowledge within the code that surfaces during code review. Newer members, with fresh eyes, discover gnarly, time-plauged areas of the code base that need a new perspective. So, code review also helps ensure new insight is tempered with existing knowledge.

> PROTIP:
>
> **Keep in mind, code review is not just a senior team member reviewing a junior team member's code. Code review should happen across the team in every direction. Knowledge knows no bounds! Yes, code review can help newer engineers, but by no means should it be used solely as a mentoring exercise.**

## But code reviews take time!

Sure, they take time. But that time isn't wasted–far from it.

> When done right, code reviews actually save time in the long run.

Here are three ways to optimize for that.

### Share the load

At Atlassian, many teams require two reviews of any code before it's checked into the code base. Sound like a lot of overhead? Really, it's not. When an author selects reviewers, they cast a wide net across the team. Any two engineers can give input. This decentralizes the process so that no one is a bottleneck, and ensures good coverage for code review across the team.

### Review before merging

Requiring code review before merging upstream ensures that no code gets in unreviewed. Which means that the questionable architectural decisions made at 2am and the improper use of a factory pattern by the intern are caught before they have a chance to make a lasting (and regrettable) impact on your application.

### Use peer pressure to your advantage

When developers know their code will be reviewed by a teammate, they make an extra effort to ensure that all tests are passing and the code is as well-designed as they can make it so the review will go smoothly. That mindfulness also tends to make the coding process itself go smoother and, ultimately, faster.

Don't wait for a code review if feedback is needed earlier in the development cycle. Feedback early and often makes for better code, so don't be shy about involving others–whenever that may be. It'll make your work better, but it also makes your teammates better code reviewers. And the virtuous cycle continues....!

SHARE THIS ARTICLE

DAN RADIGAN

Agile has had a huge impact on me both professionally and personally as I've learned the best experiences are agile, both in code and in life. You'll often find me at the intersection of technology,

photography, and motorcycling.

TUTORIAL

### Learn scrum with Jira Software

A step-by-step guide on how to drive a scrum project, prioritize and organize your backlog into sprints, run the scrum ceremonies and more, all in Jira.

Try this tutorial →

ARTICLE

### Three ingredients for great software releases

Great software releases start with modular architecture and are powered by great relationships. Learn how to have great agile software releases here.

Read this article →