Process Guidance

General Guidance

Code reviews should be part of the software engineering team process regardless of the development model. Furthermore, the team should learn to execute reviews in a timely manner. Pull requests (PRs) left hanging can cause additional merge problems and go stale resulting in lost work. Qualified PRs are expected to reflect well-defined, concise tasks, and thus be compact in content. Reviewing a single task should then take relatively little time to complete.

To ensure that the code review process is healthy, inclusive and meets the goals stated above, consider following these guidelines:

- Establish a service-level agreement (SLA) for code reviews and add it to your teams working agreement.
- Although modern DevOps environments incorporate tools for managing PRs, it can be useful to label tasks pending for review or to have a dedicated place for them on the task board - Customize AzDO task boards
- In the daily standup meeting check tasks pending for review and make sure they have reviewers assigned.
- Junior teams and teams new to the process can consider creating separate tasks for reviews together with the tasks themselves.
- Utilize tools to streamline the review process - Code review tools
- Foster inclusive code reviews - Inclusion in Code Review

Measuring Code Review Process

If the team is finding that code reviews are taking a significant time to merge, and it is becoming a blocker, consider the following additional recommendations:

1. Measure the average time it takes to merge a PR per sprint cycle.
2. Review during retrospective how the time to merge can be improved and prioritized.
3. Assess the time to merge across sprints to see if the process is improving.
4. Ping required approvers directly as a reminder.

Code Reviews Shouldn't Include too Many Lines of Code

It's easy to say a developer can review few hundred lines of code, but when the code surpasses certain amount of lines, the effectiveness of defects discovery will decrease and there is a lesser chance of doing a good review. It's not a matter of setting a code line limit, but rather using common sense. More code there is to review, the higher chances there are letting a bug sneak through. See PR size guidance.

Automate Whenever Reasonable

Use automation (linting, code analysis etc.) to avoid the need for "nits" and allow the reviewer to focus more on the functional aspects of the PR. By configuring automated builds, tests and checks (something achievable in the CI process), teams can save human reviewers some time and let them focus in areas like design and functionality for proper evaluation. This will ensure higher chances of success as the team is focusing on the things that matter.

Role specific guidance

- Author Guidance
- Reviewer Guidance

Last update: August 26, 2024