Code Reviews Guide

The purpose of this document is to explain the reasoning behind XLA team's position on code reviews -- a position that has grown from years of collective experience working on open source projects in general and XLA in particular.

Different open-source projects have different cultural expectations for how much reviewers can ask of code authors. In some projects, reviewers will take a "mostly correct" pull request (PR), modify it themselves, and submit it. XLA takes the opposite approach: We expect authors to iterate on PRs until they're good enough to submit without additional changes.

The main reason for this approach is that we want PR authors to learn to be fully-fledged XLA contributors. If reviewers fix issues in the PR themselves, then it's much harder for authors to learn. The XLA approach can be challenging for both reviewers and reviewees, but we believe it ultimately helps us grow the community.

Learning to be a "fully-fledged XLA contributor" isn't just about writing code that doesn't have bugs. There's much more to learn about "how to modify XLA." This includes:

- coding style,
- which edge cases to look out for,
- expectations around writing tests,
- expectations around comments and PR descriptions,
- and expectations around building infrastructure to support your change.

As you build knowledge of the project and trust from reviewers, you can expect that you'll get fewer comments, because you're naturally writing code more aligned with your reviewer's expectations.

Like many open-source projects, XLA has a few highly-experienced people and many relatively new people. Those of us who are highly experienced have many demands on our time. To keep

PRs moving forward in a timely manner you can help reduce the time reviewers need and the number of iterations required by:

- *Carefully reviewing and/or having your PR reviewed by a colleague before sending it:* Try to remove as many trivial mistakes (code style, spelling and grammar mistakes, etc.) before sending the PR for review. Make sure all tests pass.
- *Carefully reading your reviewer's comments:* Try to understand what the reviewer is asking for and attempt to address all comments before you push a new version.
- *Avoiding tangential discussions (bikeshedding):* Technical discussions and disagreements are highly valuable and nobody is perfect. However, avoid discussions that don't make a difference or are merely stylistic. If you disagree with a reviewer's comment, try to detail your reasons as precisely and comprehensively as possible to avoid long back-and-forth discussions.
- *Avoiding asking the "commonly asked review questions" listed below:* We have listed some answers to common questions and our rationale below.

In general, we invite you to try to make reviewing your PRs take as little time for us as possible. Then we will want to review your changes quickly!

Thank you for contributing to XLA and happy hacking!

Commonly asked review questions

"This infrastructure change is not related to my PR, why should I do it?"

The XLA team doesn't have a dedicated infrastructure team, so it's up to us all to build helper libraries and avoid technical debt. We consider it to be a regular part of making changes to XLA, and everyone is expected to participate. We generally build infrastructure as-needed when writing code.

XLA reviewers may ask you to build some infrastructure (or otherwise make a large change to a PR) along with a PR that you've written. This request may seem unnecessary or orthogonal to the change you're trying to make. This is likely because of a mismatch between your expectations about how much infra you need to build and your reviewer's expectations for the same.

A mismatch in expectations is okay! That's expected when you're new to a project (and it sometimes even happens to us old hats). It's likely that projects you've worked on in the past have different expectations. That's also okay and expected! It doesn't mean either one of these projects has the wrong approach; they're just different. We invite you to take infra requests alongside all other review comments as an opportunity to learn what we expect on this project.

"Can I address your comment in a future PR?"

A frequent question with respect to infrastructure requests (or other large requests) in PRs is whether or not the change must be made in the original PR, or whether it can be done as a follow-up in a future PR.

In general, XLA does not allow PR authors to address review comments as a follow-up PR. When a reviewer decides that something needs to be addressed in this PR, we generally expect authors to address it in the original PR, even if what's requested is a large change. This standard applies externally and also internally within Google.

There are a few reasons that XLA takes this approach.

- *Trust:* Having earned the reviewer's trust is a key component. In an open-source project, contributors can appear or disappear at will. After we approve a PR, reviewers have no way to ensure that any promised follow-ups actually get done.
- *Impact on other developers:* If you have sent a PR touching a particular part of XLA, there's a good chance other people are looking at the same part. If we accept technical debt in your PR, then everyone who's looking at this file will be impacted by this debt until the follow-up is submitted.
- *Reviewer bandwidth:* Deferring a change to a follow-up imposes multiple costs on our already-overloaded reviewers. Reviewers will probably forget what this PR was about while waiting for the follow-up, making the next review more difficult. Also, reviewers will have to keep track of the follow-ups they're waiting on, making sure that they actually happen. If the change can be made such that it is truly orthogonal to the original PR so that some other reviewer could review it, bandwidth would be less of a problem. In our experience, this is rarely the case.