Handling pushback in code reviews

Sometimes a developer will push back on a code review. Either they will disagree with your suggestion or they will complain that you are being too strict in general.

Who is right?

When a developer disagrees with your suggestion, first take a moment to consider if they are correct. Often, they are closer to the code than you are, and so they might really have a better insight about certain aspects of it. Does their argument make sense? Does it make sense from a code health perspective? If so, let them know that they are right and let the issue drop.

However, developers are not always right. In this case the reviewer should further explain why they believe that their suggestion is correct. A good explanation demonstrates both an understanding of the developer's reply, and additional information about why the change is being requested.

In particular, when the reviewer believes their suggestion will improve code health, they should continue to advocate for the change, if they believe the resulting code quality improvement justifies the additional work requested. **Improving code health tends to happen in small steps.**

Sometimes it takes a few rounds of explaining a suggestion before it really sinks in. Just make sure to always stay polite and let the developer know that you *hear* what they're saying, you just don't *agree*.

Upsetting Developers

Reviewers sometimes believe that the developer will be upset if the reviewer insists on an improvement. Sometimes developers do become upset, but it is usually brief and they become very thankful later that you helped them improve the quality of their code. Usually, if you are polite in your comments, developers actually don't become upset at all, and the worry is just in the reviewer's mind. Upsets are usually more about

the way comments are written than about the reviewer's insistence on code quality.

Cleaning It Up Later

A common source of push back is that developers (understandably) want to get things done. They don't want to go through another round of review just to get this CL in. So they say they will clean something up in a later CL, and thus you should LGTM *this* CL now. Some developers are very good about this, and will immediately write a follow-up CL that fixes the issue. However, experience shows that as more time passes after a developer writes the original CL, the less likely this clean up is to happen. In fact, usually unless the developer does the clean up *immediately* after the present CL, it never happens. This isn't because developers are irresponsible, but because they have a lot of work to do and the cleanup gets lost or forgotten in the press of other work. Thus, it is usually best to insist that the developer clean up their CL *now*, before the code is in the codebase and "done." Letting people "clean things up later" is a common way for codebases to degenerate.

If a CL introduces new complexity, it must be cleaned up before submission unless it is an emergency. If the CL exposes surrounding problems and they can't be addressed right now, the developer should file a bug for the cleanup and assign it to themselves so that it doesn't get lost. They can optionally also write a TODO comment in the code that references the filed bug.

General Complaints About Strictness

If you previously had fairly lax code reviews and you switch to having strict reviews, some developers will complain very loudly. Improving the speed of your code reviews usually causes these complaints to fade away.

Sometimes it can take months for these complaints to fade away, but eventually developers tend to see the value of strict code reviews as they see what great code they help generate. Sometimes the loudest protesters even become your strongest supporters once something happens that causes them to really see the value you're adding by being strict.

Resolving Conflicts

If you are following all of the above but you still encounter a conflict between yourself and a developer that can't be resolved, see The Standard of Code Review for guidelines and principles that can help resolve the conflict.