

# Can we improve the experience of first-time LLVM contributors?



**lowRISC**

Alex Bradbury

asb@lowrisc.org @asbradbury @lowRISC

# Questions for the audience

Who has used LLVM's Phabricator before?

# Questions for the audience

Who has used ever submitted a patch for LLVM?

# Questions for the audience

Who has submitted an LLVM patch and found it languishes with no reviewers?

# Current contribution process

- Write patch
- Submit to Phabricator
  - Try to identify a CODE\_OWNER to review
  - Tag people you might know to help review
  - Look at git blame, and pick on the unlucky soul who last touched the relevant file

# Potential problems

- Code owners are often busy
- Newcomers haven't yet gained "review currency" in the LLVM community
- Finding your hard work seemingly ignored can be offputting

Even if feedback is negative, it's valuable to know someone has looked at your code.

# What do others do? Case study - Rust

## This Week in Rust

---

06 SEP 2016

## New Contributors

- Abhishek Kumar
- Andrea Corradi
- athulappadan
- Eugene R Gonzalez
- Fabian Zaiser
- johnthagen
- Keunhong Lee
- king6cong
- Matt Ickstadt
- philipp
- QuietMisdreavus
- Sebastian Ullrich

# What do others do? Case study - Rust

  Add ThreadId for comparing threads

Verified


✗ 5bd834b



Notifications

 Sul

You're not receiving notifications from this thread.

  **aturon** was assigned by **rust-highfive** 8 hours ago



**rust-highfive** commented 8 hours ago



Thanks for the pull request, and welcome! The Rust team is excited to review your changes, and you should hear from **@aturon** (or someone else) soon.

If any changes to this PR are deemed necessary, please add them as extra commits. This ensures that the reviewer can see what has changed since they last reviewed the code. Due to the way GitHub handles out-of-date commits, this should also make it reasonably obvious what issues have or haven't been addressed. Large or tricky changes may require several passes of review and changes.

Please see [the contribution instructions](#) for more information.



# Conclusion

- Seems like a good idea - let's steal it!
- Need to provide
  - Phabricator bot
  - Community of volunteers to be tagged
- Potential pitfall: no use telling submitters to clean up code style if the fundamental approach will never be accepted by code owner
- I haven't surveyed potential LLVM contributors - maybe there isn't a problem that needs to be solved?
- Keen to hear your views - let's discuss at the Social