Inclusion in Code Review

Below are some points which emphasize why inclusivity in code reviews is important:

- Code reviews are an important part of our job as software professionals.
- In ISE we work with cross cultural teams from across the globe.
- How we communicate affects team morale.
- Inclusive code reviews welcome new developers and make them comfortable with the team.
- Rude or personal attacks doing code reviews alienate - people can unknowingly make rude comments when reviewing pull requests (PRs).

Types and Examples of Non-Inclusive Code Review Behavior

- Inequitable review assignments.
  - Example: Assigning most reviews to few people and dismissing some members of the team altogether.
- Negative interpersonal interactions.
  - Example: Long arguments over subjective topics such as code style.
- Biased decision making.
  - Example: Comments about the developer and not the code. Assuming code from developer X will always be good and hence not reviewing it properly and vice versa.

Examples of Inclusive Code Reviews

- Anyone and everyone in the team should be assigned PRs to review.
- Reviewer should be clear about what is an opinion, their personal preference, best practice or a fact. Arguments over personal preferences and opinions are mostly avoidable.
- Using inclusive language and tone in the code review comments. For example, being suggestive rather being prescriptive in the review comments is a good way to get the point across the table.

- It's a good practice for the author of a PR to thank the reviewer for the review, when they have contributed in improving the code or you have learnt something new.
- Using the sandwich method for recommending a code change to a new developer or a new customer: Sandwich the suggestion between 2 compliments. For example: "Great work so far, but I would recommend a few changes here. Btw, I loved the use of XYZ here, nice job!"

Guidelines for the Author

- Aim to write a code that is easy to read, review and maintain.
- It's important to ensure that whoever is looking at the code, whether that be the reviewer or a future engineer, can understand the motivations and how your code achieves its goals.
- Proactively asking for targeted help or feedback.
- Respond clearly to questions asked by the reviewers.
- Avoid huge commits by submitting incremental changes. Commits which are large and contain changes to multiple files will lead to unfair review of the code. Biased behavior of reviewers may kick in while reviewing such PRs. For e.g. a huge commit from a senior developer may get approved without thorough review whereas a huge commit from a junior developer may never get reviewed and approved.

Guidelines for the Reviewer

- Assume positive intent from the author.
- Write clear and elaborate comments.
- Identify subjectivity, choice of coding and best practice. It is good to discuss coding style and subjective coding choices in some other forum and not in the PR. A PR should not become a ground to discuss subjective coding choices and having long arguments over it.
- If you do not understand the code properly, refrain from commenting e.g., "This code is incomprehensible". It is better to have a call with the author and get a basic understanding of their work.
- Be suggestive and not prescriptive. A reviewer should suggest changes and not prescribe changes, let the author decide if they really want to accept the changes proposed.

Culture and Code Reviews

We in ISE, may come across situations in which code reviews are not ideal and often we are observing non inclusive code review behaviors. Its important to be aware of the fact that culture and communication style of a particular geography also influences how people interact over pull requests. In such cases, assuming positive intent of the author and reviewer is a good start to start analyzing quality of code reviews.

Dealing with the Impostor Phenomenon

Impostor phenomenon is a psychological pattern in which an individual doubts their skills, talents, or accomplishments and has a persistent internalized fear of being exposed as a "fraud" - Wikipedia.

Someone experiencing impostor phenomenon may find submitting code for a review particularly stressful. It is important to realize that everybody can have meaningful contributions and not to let the perceived weaknesses prevent contributions.

Some tips for overcoming the impostor phenomenon for authors:

- Review the guidelines highlighted above and make sure your code change adhere to them.
- Ask for help from a colleague - pair program with an experienced colleague that you can learn from.

Some tips for overcoming the impostor phenomenon for reviewers:

- Anyone can have valuable insights.
- A fresh new pair of eyes are always welcome.
- Study the review until you have clearly understood it, check the corner cases and look for ways to improve it.
- If something is not clear, a simple specific question should be asked.
- If you have learnt something, you can always compliment the author.
- If possible, pair with someone to review the code so that you can establish a personal connection and have a more profound discussion about the code.

Tools

Below are some tools which may help in establishing inclusive code review culture within our teams.

- Anonymous GitHub

- Blind Code Reviews
- Gitmask
- inclusivelint

Last update: April 24, 2023