Mastering Efficient Code Reviews

A PATH TO SUPERIOR CODEBASE

ISRAEL HERINGER

About me

- Family
- Brazilians living in the UK
- Games & Travelling
- Software Engineer @ Meta

Introduction

“A code review is a process where someone other than the author(s) of a piece of code examines that code.”

https://google.github.io/eng-practices/review/

Why?

- Discover bugs earlier
- Improve code quality
- Enhance security
- Share knowledge
- Mentor newer engineers
- Maintain compliance

Code Review is not

- Blaming or shaming someone’s code
- Showing off skills
- Executing the code

3 main approaches

- Pair programming
- Over-the-shoulder reviews
- Tool-assisted reviews

Looking inward

UNDERSTANDING THE CONTEXT

Looking inward… the team

- Seniority
- Code familiarity
- Work dynamics
- Communication style
- Global distribution

Looking inward… the code

- Coding standards
- Testability
- Riskier paths
- Legacy vs new code

Looking inward… the goals

- Finding bugs?
- Code quality?
- Testing coverage?
- Onboarding newer engineers?

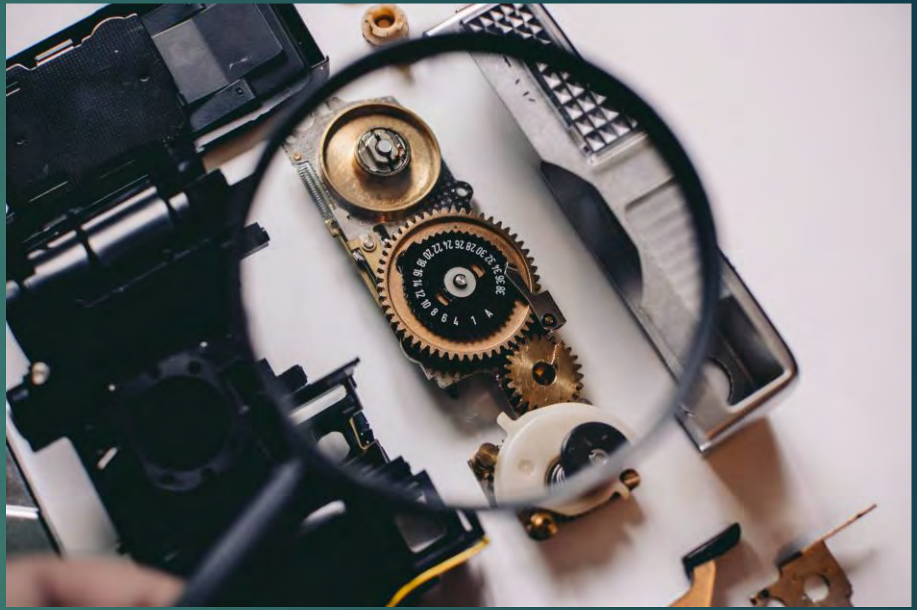
Main issues and goals change over time

Efficient code reviews

10 lines: 10 comments
500 lines: Looks good to me

Review carefully

- Actually read the code
- Efficient ≠ Fast
- Not a stamp competition
- Leave clear and actionable comments



Focus on what’s important

- Functionality
- Design
- Complexity
- Tests & Evidences
- Style
- Consistency
- Naming
- …

What about…

- Comments
- Documentation
- …

Relevance changes with context

Be explicit when nitpicking

Automate what’s possible

- Auto-formatters
- Linters
- CI warning/errors

Code Review is not an individual task

- Write code for others
  - Code is more often read than written
- Big changes mean harder and longer reviews
  - Split into smaller and meaningful changes
- Avoid judgemental and bossy language
  - Have you considered…
- Don’t forget testing scenarios and evidences

Kicking off the process

Keep engineers' autonomy

- Avoid the "gatekeeper"
- Map exceptional scenarios and workarounds
- Code Review should not be a high-friction process

Encourage open discussions

- No finger-pointing
- Team process
- Review pain points
- Evaluate results
- Iterate on the process

Recognize contributions

Some ideas to unblock the start

- Selected projects
- Critical code paths
- Sampling reviews
- Non-blocking reviews

Start small, grow as the team gets used to the process

Wrap up

Wrap up

- What is (and what is not) Code Review and why
- Understanding the context of the team, code and goals
- What to look for when reviewing code
- What to avoid during code review
- Code Review is a team process
- Some ideas to help with the process kick-off

Thank you

- /israelhlc