Code Reviews

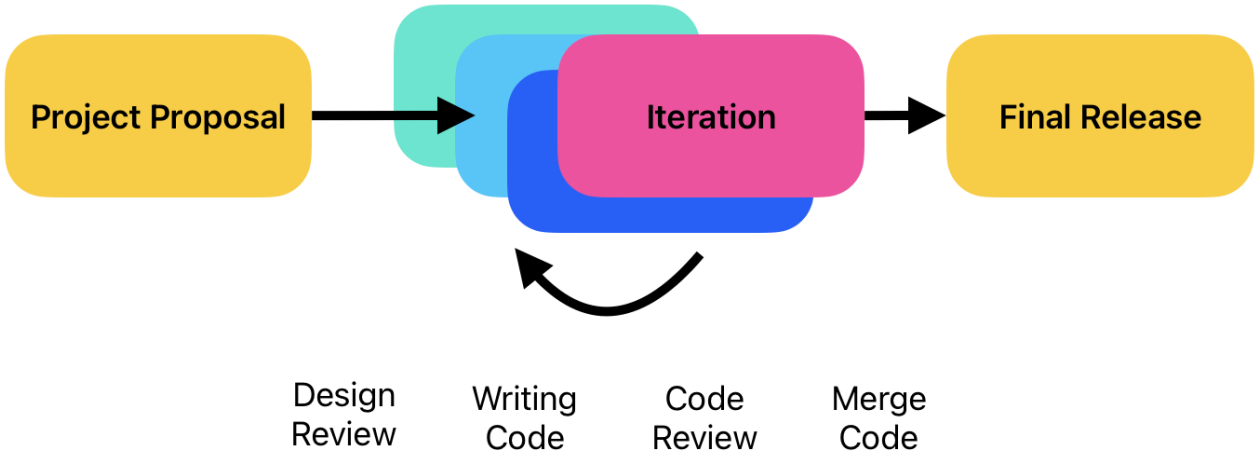
CS 346: Application Development

What is a “code Review”?

A code review is a **peer review of code** that helps developers ensure or improve the code quality before they merge and ship it.

- gitlab.com

- A code review is not a design review.
- The intention of a code review is to review code that was produced to solve a particular problem.
- They are optional in this course; may be mandatory @work.



Code reviews are done at the end of a development iteration, before the code is merged. This provides time to accept the changes or revise them before they are accepted.

Why perform code reviews?

Benefits to the Code

- Ensures consistency in design and implementation.
- Can lead to more optimal code (“more eyes on the problem”).

Benefits to the Team

- Collaborating and sharing new techniques.
- Reinforces mutual understanding of the code (team ownership).

- What is NOT beneficial?
  - Using them to gatekeep code. Micro-managing changed to the codebase.

Code reviews are Challenging

Code reviews are challenging for both the person who has produced the code, and the reviewer providing feedback.

- As the code producer:
  - It's stressful to have someone critique your work ("maybe I'm terrible at this?!")
- As the reviewer:
  - Code reviews are time-consuming. You need to:
  - Understand the problem (i.e. the underlying requirements).
  - Understand the design (i.e. "is this the best way to design it?").
  - Review the code well enough to understand the implementation.

When to do code reviews

Save **formal code** reviews for major, significant features.

- Code producer needs to submit detailed documents + code ahead of time.
- Formal reviewer reviews everything and comes with written feedback.
- Some/all of the team participates - although only reviewer is expected to provide detailed feedback.

For lightweight or simple features, consider an **informal review**.

- One developer asking another for ad hoc feedback:
  - e.g., “Can you look at this and tell me if it’s clear?”
  - e.g., “After lunch, would you mind reviewing this function for security implications?”

Pair programming is not a substitute for a formal code review but can reduce the need to have a formal review in the first place.

Code Producer: Preparing for a Review

Document your **design**

- Summarize your understanding of the issue.
  - State any assumptions that you've made.
- Describe your solution. Detail any issues or limitations to your approach.
  - If it will help clarify your design, generate one or more UML diagrams.

Document your **code**

- Provide guidance on how the code is structured; what you changed.
- Code comments should be injected to help readability and understanding.
  - Add inline explanations for *why* something is designed the way that it is.
  - Say "no" to formulaic design recipes (unless required to generate documentation).

Code Reviewer: Preparing for a Review

Review the materials that the code producer has created.

- This includes reviewing the code ahead of time.

Write a summary of your feedback.

- Tell the presenter that you have your feedback written down.
- Verbally present your concerns in the session and help find a resolution.
- Provide it at the end of the review.

Be respectful of the effort that someone has put in.

- Don’t just point out flaws; point out what they did well & what you like!
- Treat code reviews can be a learning opportunity for you and the entire team.

Running the Code Review Session

Book a meeting time. In-person is preferred.

- Code producer and reviewer are required attendees.
- Have a projector and a laptop to share design/code.
- Team members are optional (but should attend if they can).

If it's just the producer/reviewer, they can discuss written feedback.
If other team members are present, I'd suggest the following format:

1. The code producer presents the problem & an overview of the solution.
2. The code producer presents the code - UML diagrams + high-level code.
3. The reviewer then presents their feedback one item at a time.
4. Outcome: the team decides (based on pros/cons).

Code Reviews in GitLab

Code reviews are tied to the `Merge Request` workflow.

When you `Create a Merge Branch`, you select a single Reviewer.

- Reviewer and Coder `communicate` by adding comments to the issue.
- Coder can close the merge request when any problems have been addressed.