Code Quality: Individual Efforts

Care about design

- Good variable names
  - No single letter variables (except for those in loops).
  - No cryptic variable names.
- Short methods
  - It doesn’t mean that they are small. It just means that they focus on a single level of abstraction.
- Smaller classes
  - Focus on single abstraction, single responsibility principle.
- Learn by reading good code
  - Share it with others.

KISS — Keep It Simple Stupid

- Don’t unnecessarily complicate code.
- Reduce code, remove things that aren’t necessary.
- Aim for simplicity.
- Clarity in code — strive for it.

Automated Tests with high coverage

- Helps make sure that our code continues to work as we make changes to it.
- Consider untested code as unfinished code.

Run tests before check-in

- Make sure your tests pass before you check in your code.

Avoid cargo cult

- When we look at some practice we like, using it without really understanding why it exists and how it helps shouldn’t be done.
- Pick and choose practices that make sense.

Court feedback and criticism

- Be comfortable asking for help/feedback.
- Build desire to improve yourself by getting feedback.

Check-in frequently

- Controversial topic.
- Make some change, check in and see if everything still works or things break.
  - If things don’t work out, you can get the previous version from version control.
  - If things work out, you can continue working on that feature.
- Other people can integrate with your code quickly and give you feedback.