Create Your Own Open Source Project

Cliff Green

GDG ABQ, June 1, 2023, updated Aug 25, 2023

What am I going to talk about?

- Open source projects
  - Motivations and ideas
  - Quality recommendations
  - Benefits
  - Constraints
- Software engineering best practices (use these on your projects, whether open source or not)
  - Unit testing
  - Branching strategy
  - Continuous integration
- Interests completely unrelated to software engineering
  - Just a few fun pics, I promise!

What is my tech background?

- 30+ years as a software engineer
- Worked at two startups, multiple small companies, and 12 years at Boeing Defense (primarily AWACS)
- Taught at Univ of Washington Continuing Ed and Learning Tree (technical training company)
- Designed and coded network libraries, systems and infrastructure libraries, high availability systems, parallel architectures, and many back end systems
- Love coding in C++ and many other programming languages

By Senior Airman Roslyn Ward - usaf.mil, Public Domain, https://commons.wikimedia.org/w/index.php?curid=127132264

By Eugene Butler - https://cdn.jetphotos.com/full/2/88258_1305969734.jpg, GFDL 1.2

What are my open source projects?

- C++ networking and utility libraries: https://connectivecpp.github.io/
  - Multiple members from the Seattle area
  - Multiple repositories, quite a bit of stable, tested code (has been in hiatus for a couple of years, will become more active this year)
- Robotics, AI, other tech: https://github.com/robo-neurdology
  - A few ABQ local tech people are in the organization (me, Bill Holcombe, Lou Langholtz)
  - Not much (yet) on GitHub, lots of discussion in our Discord
  - We welcome new, interested people

What are some motivations for / benefits of an open source project?

- Modify an existing open source project to your own needs
- Create software or tech that you own
- Learn new and interesting tech
- Enhance your career
- Let freedom and creativity thrive in an environment without the usual business or organizational expectations and constraints
- Provide a benefit to your existing employment environment (e.g. greatly increase the testing and feedback on a particular app or library)

Did Bill H give me some inspirational quotes to use?

- Why yes, he did - glad you asked!
- “Imagine a thing you want to exist, that lots of people would find useful. Now imagine you decide to hire the best programmers in the world to help you make it. With an open-source project that's a real option and you don't have to pay them a 6 figure salary to do it!”
- https://blogs.vmware.com/opensource/2020/08/25/boost-your-career-through-open-source-contribution/
- “Working within open source communities has given me opportunities that I never would have thought possible. I’ve presented on various open source topics at conferences around the world, which has allowed me to meet all kinds of interesting people! The visibility and deep connections to other people that come from working in open source have given me more career options and allowed me to have more control over my career.”

  –Dawn Foster, Director of Open Source Community Strategy, VMware Open Source

What if you don’t have a particular project in mind?

- Consider functionality you’ve developed in a job or at school that would be useful to others, but be careful:
  - Your project must not contain proprietary functionality
  - Never copy and paste proprietary or non open source code - always develop code from scratch, as if in a “clean room”
- Expand the portability and supported platforms of an existing open source project
  - Support Windows, Linux, macOS, Raspberry Pi OS, etc as appropriate
  - Enhance for older or newer compilers or frameworks or GUIs
  - Port to a different hardware platform
- Design and implement a simple game or useful utility
- Fork an existing open source project to better suit your functionality needs
- Develop a project for new, unique, or small devices
  - IoT, embedded, “maker” projects

What can you do to spread the word?

- E-mail / Discord / Slack / etc to friends and co-workers
- Post on Reddit appropriate groups
- Create a presentation, whether local or for an online event
- Add it to your resume and LinkedIn page

How do we get started?

- Create an account on GitHub (as needed)
  - Other hosts are possible (e.g. GitLab, BitBucket)
- Create a repository
- Decide on which open source license to use (MIT, Boost, Microsoft, BSD, GNU GPL, etc), then choose / create the LICENSE.txt file
- Create a README.md file
- Start populating the repository!
- Additional guides:
  - https://www.freecodecamp.org/news/how-to-start-an-open-source-project-on-github-tips-from-building-my-trending-repo/
  - https://opensource.guide/starting-a-project/

What makes a good README file?

- An overview that answers the following questions:
  - What is the library or app's basic functionality?
  - Who is the intended audience?
  - What is a simple motivating example?
  - What gap is this library or app trying to fill?
  - What are similar libraries / apps and why is this one better?
  - What are the supported compilers / platforms / frameworks / language standards / build systems / etc?
- See GitHub's document about README files
- Additional details can be left for later sections

What are some open source examples?

- Catch2, a C++ unit test library: https://github.com/catchorg/Catch2
  - Basic functionality is explained along with advantages and comparisons to similar libraries
  - Motivating examples are provided
  - “Badges” indicate high level of continuous integration and provide info on supported platforms and tools
  - Rich set of unit tests, build support, and other indicators of quality
- Span-lite, a well presented small C++ library for `std::span`: https://github.com/martinmoene/span-lite
- Fmt, “printf” style formatting in C++: https://github.com/fmtlib/fmt

What quality factors are in good open source?

- Unit tests that cover a high percentage of the functionality and code
- Continuous integration (CI) to invoke unit tests, doc toolchains, etc
- Build system support (e.g. CMake for C++)
- Good documentation ("good" can be subjective, whether API docs or higher level tutorials)
- Quality inspection tools, typically as part of CI, e.g. sanity checkers, fuzz tools, static coverage tools

Can I show some gratuitous pictures?

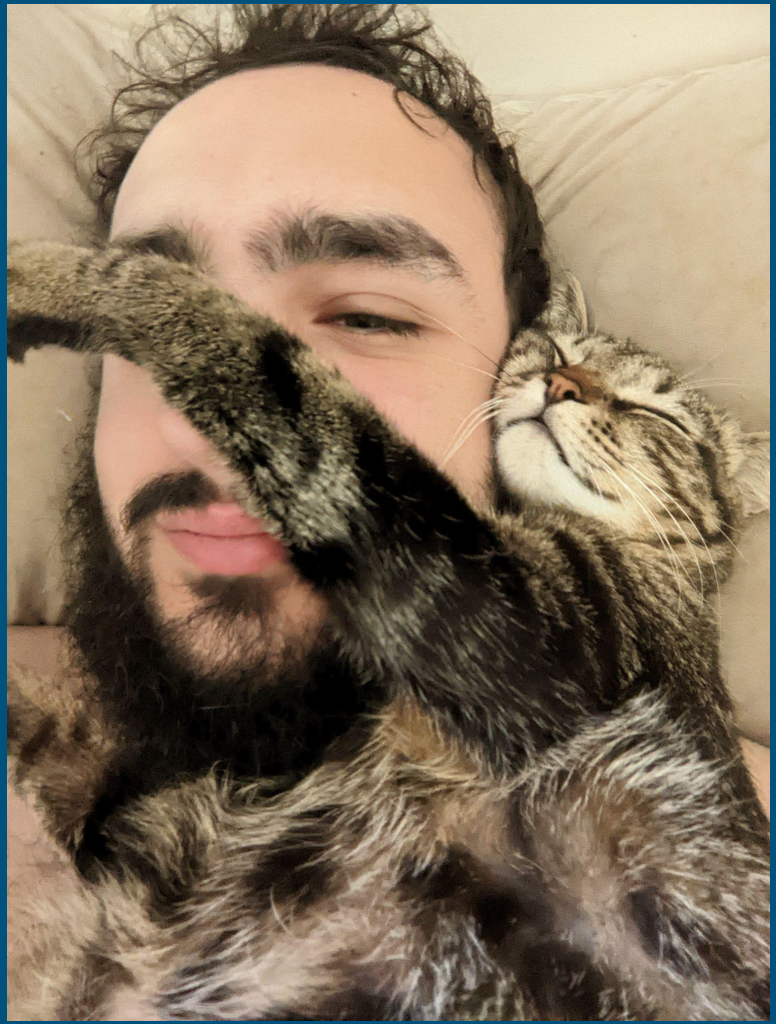
- Of course - you are a captive audience! (However, I promise to be quick.)
- Some of my other interests include:
  - Hiking - following pics were taken in both WA state (Lake Serene) and NM (Sandia Mtn La Luz trail, Ghost Ranch Chimney Rock trail)
  - Aviation, in particular soaring (gliding) - I'm majority owner of Merlin Aviation, Inc, based in Moriarty, NM, https://www.merlinaviation.net/
  - Railroading, I'm a member of New Mexico Heritage Rail, restoring the Santa Fe 2926 steam engine
  - Showing pics of our son and our grandcat named Stoney
- (All gratuitous pics are owned / copyrighted by me or my aviation company)


10
N503S


10
10
N503S


N5035 #10


2926
2926



What are some software engineering best practices to use in your project?

- Use git for version control and branching or an equivalent tool
- Use continuous integration (CI)
- Implement automated tests
- Define a branching and release strategy
- All of the above are good practices, whether open source or not, and show mature development processes

What is continuous integration and why is it a good thing?

- Continuous integration (CI) is where code (or documentation) changes trigger automated builds and tests
  - Builds are triggered for multiple platforms or compilers as appropriate
  - Unit tests are run, results become available on hosting system
  - GitHub supports a rich set of CI workflow commands named GitHub Actions
  - CI automatically runs on GitHub (or appropriate host), no manual starts required

Why do I emphasize test driven development?

- Automated tests take time and effort up front, but pay for themselves many times over
  - CI will run unit tests, every committed change will provide results
  - Regressions are typically caught much sooner in the dev cycle
- Designing test functionality improves interfaces and correctness
  - “Turns around” the thinking on production code APIs - you write explicit, detailed code as a user instead of the API implementor
  - Requires thinking about input constraints, boundary conditions, and other “design by contract” aspects (invariants, pre-conditions, post-conditions)

What is a branching strategy?

- A branching strategy provides coordination among multiple developers, particularly for parallel development
- The strategy also controls and manages releases, providing 100% reproducible builds
- Multiple types of branching strategy
  - Main / release and integration branches at a minimum
  - Different requirements may drive different types of branches
    - Bug fix branches
    - Experimental or new feature branches
    - Multiple platform branches
- Good discussion at https://www.flagship.io/git-branching-strategies/

What are some boundaries and constraints to consider for an open source project?

- Decide which operating systems, platforms, and / or frameworks are supported, and document it
- Define clearly what functionality is provided and what is not
  - An open source project is not an offer to provide unlimited work for free just because “feature X would be really great to have in your project!”
  - Requests for new functionality or significant changes may be a basis for consulting work or other monetization
- Have a supportive culture for your collaborators and contributors

What are some other considerations for an open source project?

- On GitHub, if multiple developers on your project, consider creating an organization (versus personal repositories)
- Be responsive to bug reports, whether they are true bugs or not
- Evaluate how much time and motivation you can put into your project
- Open source projects provide continuing benefits
  - To you and your career and knowledge base
  - To your users
  - It’s not just “something to do” after code camp or degree to “get a job”

What questions or discussions do you have?

- Don’t be shy, I’m not
- Contact info: cliffg2004@gmail.com, 206-915-4382