Five things Go taught me about Open Source

Velocity Berlin 2019
What makes an open source project popular?
Proposition 1

Velocity is not a programming language conference. You don’t want to hear a dork from Australia rabbit on about Go.
Proposition 2

Open source has influenced—at least in part—the career of everyone in this room.
“As of November 2019, my best estimate is between 1.15 and 1.96 million.”

—Russ Cox, https://research.swtch.com/gophercount
Was it a maniacal focus on simplicity?
Was it solid performance?
Was it modern approach to concurrency?
Stoic commitment to backwards compatibility?
Google’s money?
The distribution model
Single, static, binary

Dead simple cross-compilation
First impressions are really important
Make it as simple as humanly possible for someone to try your project.
The Gopher

Image: @egonelbre
Gophers reimagine their relationship with the language via their art
Contribution is more than just code
“The actual company values, as opposed to the nice-sounding values, are shown by who gets rewarded, promoted or let go.”

—Patty McCord
Use internal packages to reduce your public API surface

In the beginning, before the go tool, before Go 1.0, the Go distribution stored the standard library in a subdirectory called pkg/ and the commands which built upon it in cmd/. This wasn't so much a deliberate taxonomy but a by product of the original make based build system. In September 2014, the Go distribution dropped the pkg/ subdirectory, but then this tribal knowledge had set root in large Go projects and continues to this day.

I tend to view empty directories inside a Go project with suspicion. Often they are a hint that the module's author may be trying to create a taxonomy of packages rather than ensuring each package's name, and thus its enclosing directory, uniquely describes its purpose. While the symmetry with cmd/ for package main commands is appealing, a directory that exists only to hold other packages is a potential design smell.
There are many ways to contribute to an open source project, make sure you recognise and reward them equally.
A functioning job market
The image is a graphical representation of the Technology Adoption Curve (TAC), also known as the Gartner Hype Cycle. It illustrates the stages of a technology as it matures from introduction to obsolescence.

- **Technology Trigger**: This is the initial stage where the technology is first introduced to the market.
- **Peak of Inflated Expectations**: The technology is initially adopted rapidly, leading to exaggerated expectations.
- **Trough of Disillusionment**: The technology fails to meet the inflated expectations, leading to a significant drop in visibility.
- **Slope of Enlightenment**: The technology is revisited and its true value is realized, leading to renewed interest.
- **Plateau of Productivity**: The technology reaches a stable phase where it is adopted and utilized effectively.
Choosing to invest your career in a new technology implicitly means letting your current skills lapse.
Go got lucky
Make sure you write a hit song
Codes of conduct
Codes of conduct are just words on paper, but they're important words.
Go waited too long to adopt a code of conduct
If you plan to build a community around your project, have a code of conduct in place from the start.
Open governance and succession
The source code of Go may be open, but membership of its governing body is not.
It is not enough for a project to be open source, it must also embrace an open governance model.
First impressions matter

Contribution is more than code

Functional job market is key

Write a code of conduct, enforce it

Adopt open governance and plan for succession