Scaling Open Source Projects from 0-1000 Commits

Barak Michener
Let us know what you think

Click ‘engage’ to rate session.
Rate 12 sessions to get the supercool GOTO reward

Join the conversation #gotoldn
~100 Open Source repos on Github
#gifee

Google’s Infrastructure For Everyone Else
http://github.com/google/cayley
0 commits
$ git init .
Skeleton

- You may want one for your org
- Standard choice of licenses, Make/BUILD files
- README.md
- CONTRIBUTING.md
A note on Licenses:

PICK ONE.

http://choosealicense.com/

Only ~20% of Github repos have a license. Don’t be “that project”.
Code Style

- Mandate by fiat for your project/org
  - Avoid bikeshedding
- Big plug for Go here, `gofmt` is awesome
- PEP8 for Python fans
- Always an option: Adopt (or fork) from Google
  - [https://github.com/google/styleguide](https://github.com/google/styleguide)
- Mention this in CONTRIBUTING.md
The past 200 commits

- Commit messages may have been messy
  - more work
  - remove cruft, add test
  - Don’t do this in general, but we all get started and rebasing is fine, until now.
- Getting things working
- Super fast iteration
What if it doesn’t work out?

- That’s okay!
  - Experimentation should be more encouraged
  - “Fail fast”
  - Rebasing early is okay too
    - Your teammates may hate you a little, but only a little.
Prepping for Announce

- Mailing list
- IRC Channel?
- ...And two important things
From “Hierarchy of Developer Needs” by +JuliaFerraioli
ROADMAP.md

(Pull requests accepted.)
Announce!

The sooner you’re working in the public sphere, the better.
The past 200 commits

- Real commit messages, PRs, code review
  - subpackage: Add new feature doing X. Fixes #23.

- Still early, things subject to change.
  - Don’t be afraid to experiment, and break a few things
First external contributors!

- Treat them like gold.
- If you have to answer the same question three times, write a three paragraph answer. Start your wiki/FAQ.md/Documentation file.
Optional: Benchmarks

- They don’t have to be perfect
- Having them means measurement, and numbers help quantify decisions
- Shout-out for Go again
$ benchx boltnoproto boltwproto

<table>
<thead>
<tr>
<th>benchmark</th>
<th>old ns/op</th>
<th>new ns/op</th>
<th>delta</th>
<th>mult</th>
</tr>
</thead>
<tbody>
<tr>
<td>BenchmarkNamePredicate</td>
<td>536192</td>
<td>519102</td>
<td>-3.19%</td>
<td>-1.03x</td>
</tr>
<tr>
<td>BenchmarkLargeSetsNoIntersection</td>
<td>25406759</td>
<td>16517821</td>
<td>-34.99%</td>
<td>-1.54x</td>
</tr>
<tr>
<td>BenchmarkVeryLargeSetsSmallIntersection</td>
<td>55738157</td>
<td>38773174</td>
<td>-30.44%</td>
<td>-1.44x</td>
</tr>
<tr>
<td>BenchmarkHelplessContainsChecker</td>
<td>72689377</td>
<td>50568672</td>
<td>-30.43%</td>
<td>-1.44x</td>
</tr>
<tr>
<td>BenchmarkHelplessNotContainsFilms</td>
<td>125709712</td>
<td>82282494</td>
<td>-34.55%</td>
<td>-1.53x</td>
</tr>
<tr>
<td>BenchmarkHelplessNotContainsActors</td>
<td>101334859</td>
<td>67082261</td>
<td>-33.80%</td>
<td>-1.51x</td>
</tr>
<tr>
<td>BenchmarkNetAndSpeed</td>
<td>5530942</td>
<td>4847584</td>
<td>-12.36%</td>
<td>-1.14x</td>
</tr>
<tr>
<td>BenchmarkKeanuAndNet</td>
<td>4399839</td>
<td>4081278</td>
<td>-7.24%</td>
<td>-1.08x</td>
</tr>
<tr>
<td>BenchmarkKeanuAndSpeed</td>
<td>5144942</td>
<td>4554785</td>
<td>-11.47%</td>
<td>-1.13x</td>
</tr>
<tr>
<td>BenchmarkKeanuOther</td>
<td>16995806</td>
<td>12897262</td>
<td>-24.12%</td>
<td>-1.32x</td>
</tr>
<tr>
<td>BenchmarkKeanuBullockOther</td>
<td>26998991</td>
<td>19858152</td>
<td>-26.45%</td>
<td>-1.36x</td>
</tr>
</tbody>
</table>
600 commits
The past 200 commits

- A couple of early releases
- Semantic versioning
- Support!
If you give a mouse a binary...

...he’s going to put it into production.
Support is great

- Means people have a need for your project
- Finds bugs

However,

- Sometimes a burden on small teams
WTFM

- You’re the person who knows the most about your project
- Sourcing writers is hard, and you won’t have many at this stage
- Roll up your sleeves and write the docs
- Pay attention to UX as well
- Closing bugs with a pleasant “Hi, thanks for your question; we’ve committed an answer here based on it” helps everyone!
800 commits
The past 200 commits

- Prep for 1.0
- Stabilize APIs
- Major refactors tricky but possible
1000 commits
Where do we go from here?

- Release cadence
- stable/master
  - 2.0.x on stable, 2.1alpha/rc on master
  - 2.1.x on stable, 2.2alpha/rc on master
- ???
- Profit!
Please remember to rate this session.

Thank you!

Join the conversation gotoldn
Thanks !

@barakmich
@coreoslinux