Metrics for open source community health

Can it be done?

Benjamin Balder Bach (Twitter/Github: @benjaoming)

Disclaimer: This is a prototype of an idea

You can even convince me that it's a bad idea!

- Idea is incomplete
- Prototype is incomplete
- This talk is incomplete

2 types of "community health" tools:

- Pro-active
- Re-active

Pro-active health tools

Growing and maintaining a community:

- Conferences, sprints
- HappinessPackets.io and SayThanks.io
- Code of Conducts
- Documentation, tutorials

Re-active health tools

Something that identifies challenges or threats?

Tasks and challenges for a community

- Maintaining
- Writing (good) docs
- Triaging bugs
- Fixing bugs
- Reviewing code
- Etc...

What's most critical?

How are these inter-connected?

Will skip that for now...

Approach: What can actually be measured?

And what does it mean?

Metrics: Github API

Popularity:

- Stargazers
- Watchers
- Forks

Top 10 most starred...

Popular? Used? Important?

```
vinta/awesome-python (1/100)
rg3/youtube-dl (2/100)
toddmotto/public-apis (3/100)
pallets/flask (4/100)
tensorflow/models (5/100)
nvbn/thefuck (6/100)
jakubroztocil/httpie (7/100)
django/django (8/100)                         # !!!
josephmisiti/awesome-machine-learning (9/100)
requests/requests (10/100)
```

Popularity => community size

Yes, let's assume that!

Metric 1: Popularity

Formula

```
watchers + stars + forks
```

Metric 2: Contributors vs. popularity

Formula

`popularity / contributors`

Explanation

...

Recommendations

- This project's popularity has unyielded potential!
- Help build and maintain communication channels
- Delegate responsibility or ask for it

Metric 3: Open issues vs. popularity

Formula

`popularity / open_issues`

Explanation

Recommendations

- Triage issues
- Write an issue template
- Improve your documentation
- Encourage people to open pull requests

Some results...

Be part of the community of community health!

Comment and create metrics!

github.com/benjaoming/python-community-health