Nim in Production

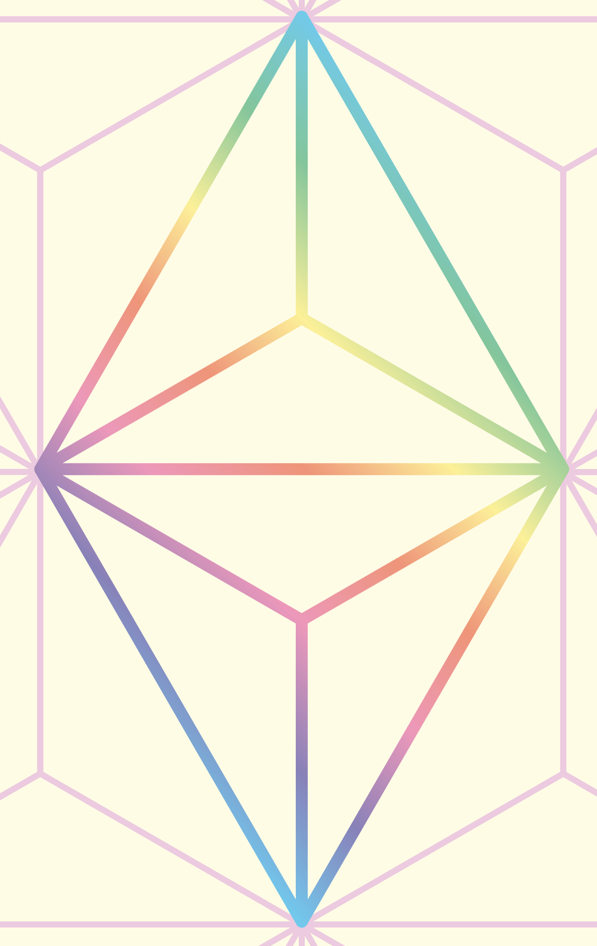
"With great power comes great responsibility"

Mamy Ratsimbazafy

Ethereum 2 / Nimbus developer @ Status.im

m_ratsim

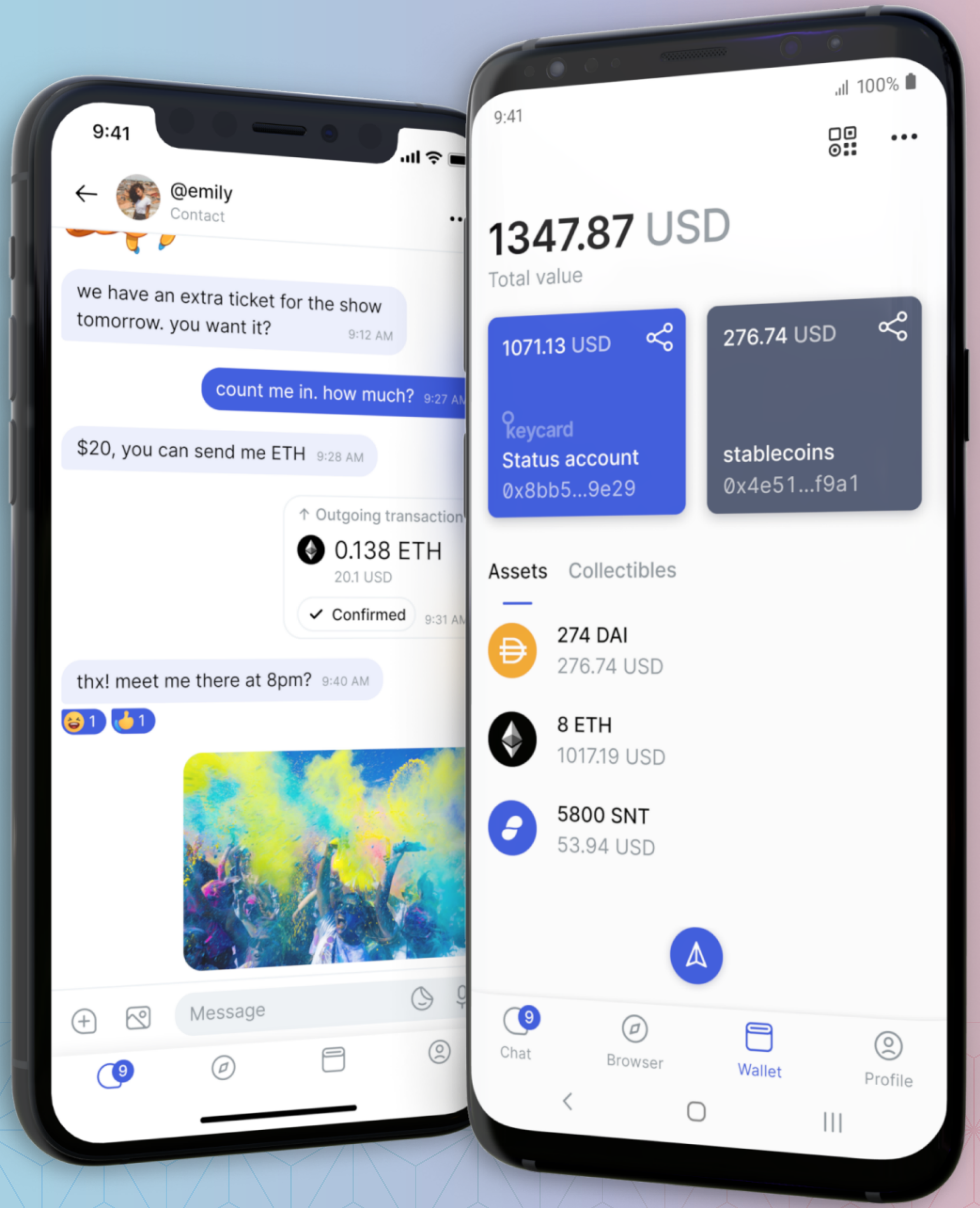
mratsim



Status focuses on

https://status.im

Products



Developer tools

Infrastructure and protocol research

Addressing the decentralized trinity

- Decentralized messaging
- Decentralized consensus
- Decentralized storage

Providing the foundations of a global P2P economy

All open-source and transparent, including meetings and financials

Who uses Nim at Status?

- Nimbus Ethereum 1 (Blockchain client)
- Nimbus Ethereum 2 (Blockchain client)
- LibP2P (P2P network stack)
- Waku (P2P messaging protocol)
- Status Desktop (Messenger)

Over 20 people, half were Nim devs before.

Other half comes from C, C++, Go.

Context & Stakes

- Blockchain & cryptocurrency
- CS domains: P2P networking, cryptography, databases, caches
- Moving fast
- High-profile hacking target
- Securing 5.4M ETH -> $12.1B as of June 19, 2021
- Media: high-profile on Crypto-Twitter and Crypto-Reddit
- High demand for documentation, audits, reproducible builds

Hiring & Ramp-up

C, C++, Go, Rust developers

Have an easy transition into Nim

1 month

To get productive for a domain expert. Learning a domain (P2P networking or blockchain) takes longer. Both Nim and a domain can be learned at the same time

Get software craftsmen

Driven and passionate about code.
Care about code quality.
Would develop on their free-time

Need at least 1 Nim “mentor”.

Lots of knowledge hidden in Nim bug reports and RFCs

Miscellaneous advantages

C & C++ interoperability

Can use any C library easily.

c2nim or nimterop for autogen with minimal change

Python research

Can port Python specs almost 1-to-1.

Nim implementation readable for researchers and auditors

Speed & Memory usage

https://github.com/jclapis/rp-pi-guide/blob/main/Docker.md

After having done lots of testing on my Raspberry Pi, I can confidently say that Nimbus has given me the best results. It's designed specifically for low-power systems like this; it uses the least amount of RAM by far (about 700 to 800 MB).

Need at least 1 Nim "mentor".

Lots of knowledge hidden in Nim bug reports and RFCs

Dealing with an immature ecosystem

- Standard library
- Compiler updates
- Nimble: no lockfiles or monorepo
- Third-party packages

- Fork (like Google's Abseil or Facebook's Folly)
- Pin to a compiler version
- Makefiles
- Clone and pin

Safety

- Auditor's handbook: https://nimbus.guide/auditors-book/
- Style guide:
  - https://github.com/status-im/nim-style-guide
  - https://status-im.github.io/nim-style-guide/
- Exceptions, Result types, Options, {.push raises: [].}
- Restricted safe subsets of Nim
- Fuzzing and sanitisers

Developing in async teams vs as individuals

- Automation: environment script, CI for all platforms
- Branches: stable, unstable, testing
- Pinning dependencies and compiler is key
- C++ can be a time sink (GCC vs Clang for unions, flexible array members, casts)
- Weekly sync, quarterly in-person meet for strategic features
- Workflow: PRs + reviews + architecture docs
  - Bonus: user docs, auditors, release notes, new joiners, yourself 6 months from now

Let's BUIDL!

Nim in Production

"With great power comes great responsibility"

Mamy Ratsimbazafy

Ethereum 2 / Nimbus developer @ Status.im

m_ratsim

mratsim