YEARLY PROJECT

By @raptazure - Haoran Liu

Source Code of the Slide

I. CHOOSE YOUR FIELDS

- Machine Learning
- Embedded Systems
- Web Applications
- Computer Graphics
- Operating Systems
- Distributed Systems
- Databases and Storage System
- Programming Language Theory
- ...

MACHINE LEARNING

- Video: Andrew Ng Machine Learning
- Frameworks: tensorflow, pytorch
- Computer vision: OpenCV(cv2), YOLOv3
- Not only Python (Maybe JS or Rust?)

EMBEDDED SYSTEMS

- Raspberry Pi: https://www.raspberrypi.org/
- Linux and Open Source
- The Rust Programming Language
  - Rust is blazingly fast and memory-efficient: with no runtime or garbage collector, it can power performance-critical services, run on embedded devices, and easily integrate with other languages.
  - Rust's rich type system and ownership model guarantee memory-safety and thread-safety — enabling you to eliminate many classes of bugs at compile-time.
  - Rust has great documentation, a friendly compiler with useful error messages, and top-notch tooling — an integrated package manager and build tool, smart multi-editor support with auto-completion and type inspections, an auto-formatter, and more.
  - The Embedded Rust Book | Chinese version

WEB APPLICATIONS

- Goals are important: what do people need?
- Front-end:
  - HTML, CSS, JavaScript → TypeScript
  - Framework/Library: Vue, React, Angular
  - Reactive, Functional: rxjs, elm, OCaml...
- Back-end:
  - RESTful API design, MVC
  - Frameworks: Node.js(express, nest), Java(Spring), Python(flask, django), Ruby(rails), elixir(phoenix)...
  - Databases: MySQL, PostgreSQL, MongoDB...
  - Container: Docker, Kubernetes...
- Mobile Apps:
  - Native: Java(Kotlin) for Android, Objective-C(Swift) for IOS
  - Hybrid: React Native, Flutter (go back to front-end)
- Desktop Apps: Cross-platform solution - QT, Electron...

COMPUTER GRAPHICS

- Rust implementation of “Ray Tracing in One Weekend”
- Games: Unity, UE4...

OPERATING SYSTEMS

- Tutorial for rCore OS step by step (2nd edition)
- Writing an OS in Rust - Philipp Oppermann's blog

DISTRIBUTED STORAGE SYSTEM

- MIT 6.824: Distributed Systems
- PingCAP Talent Plan: TiDB and TiKV

PROGRAMMING LANGUAGE THEORY

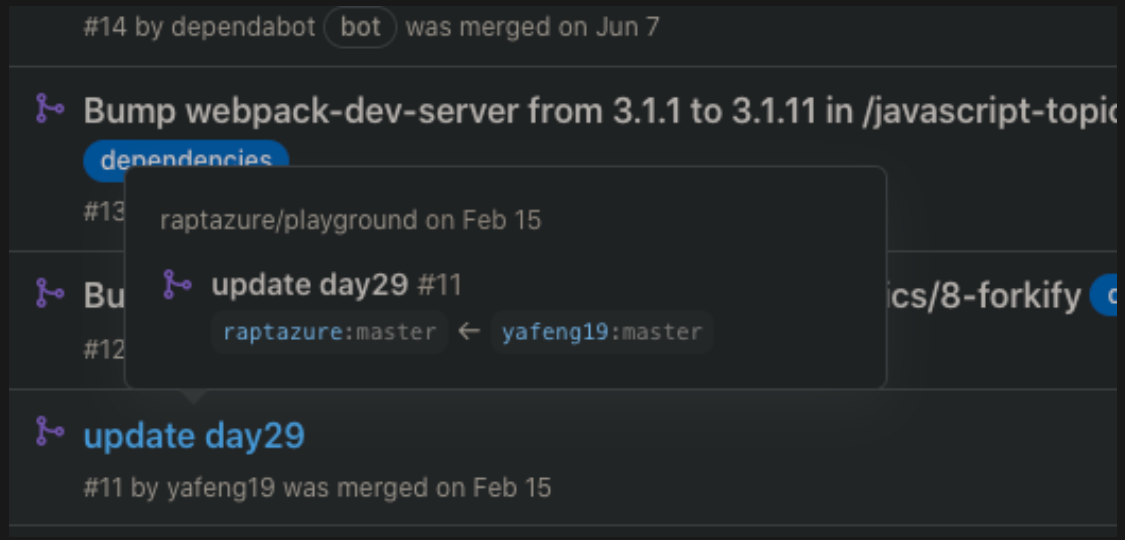
- SICP (MIT 6.001) and Computer Magic
- Functional programming: Haskell, Elixir, λ-calculus...
- Formal Proof: proof assistant → Agda, Arend...
- Type Theory: HoTT, CuTT, Dependent type → Agda, Idris...
- Implement a lisp/scheme interpreter using Haskell...
- Write a C compiler using Rust → ref: rcc
- Design your own programming language

II. ORGANIZE YOUR TEAM

- Open Source Workflow
- GitHub Project

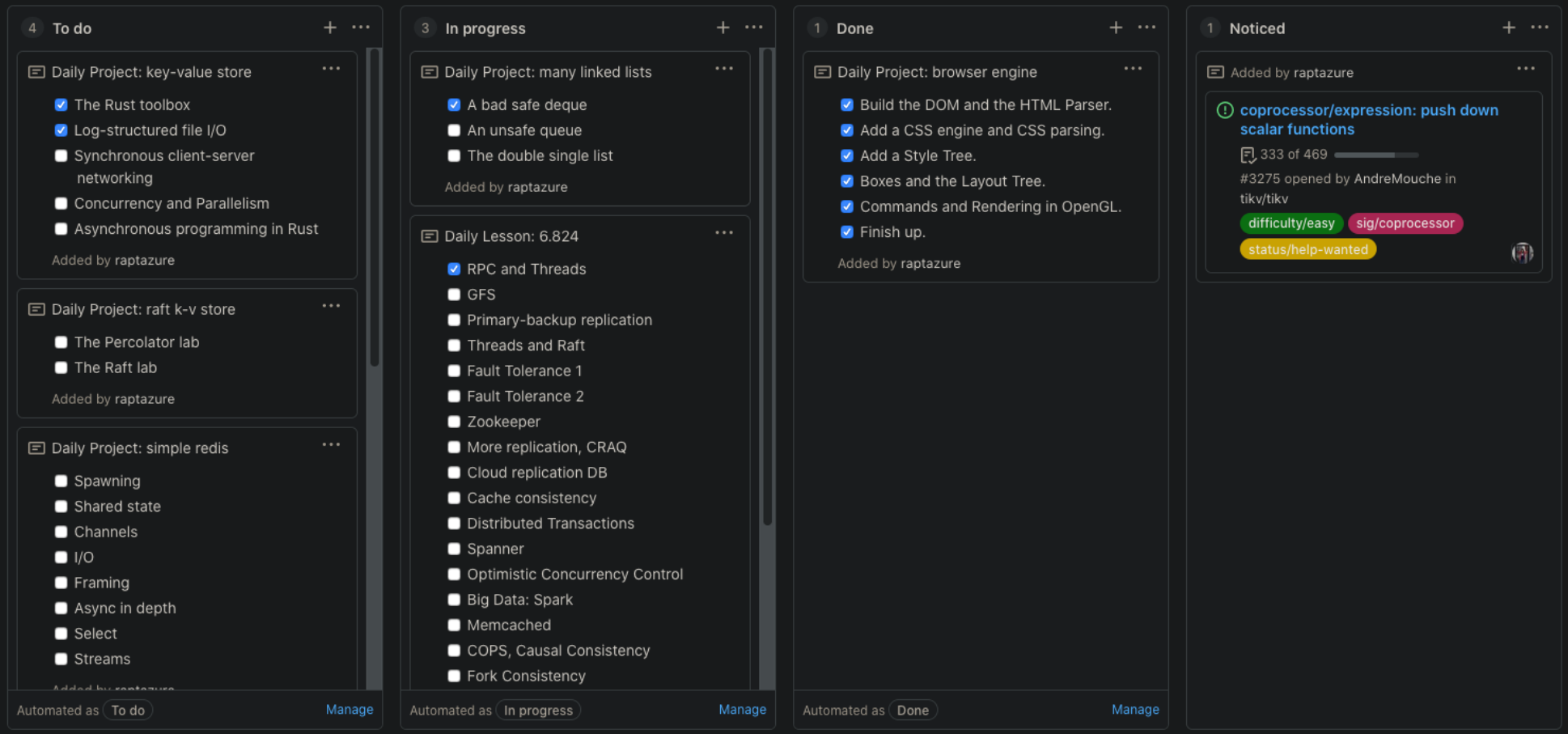
OPEN SOURCE WORKFLOW

- Push your code to GitHub (using git)
- Pull Requests / Code Review
- Merge PR and work together
- Assistant: QQ Group



GITHUB PROJECT

- Manage your progress
- Assign tasks to your team members

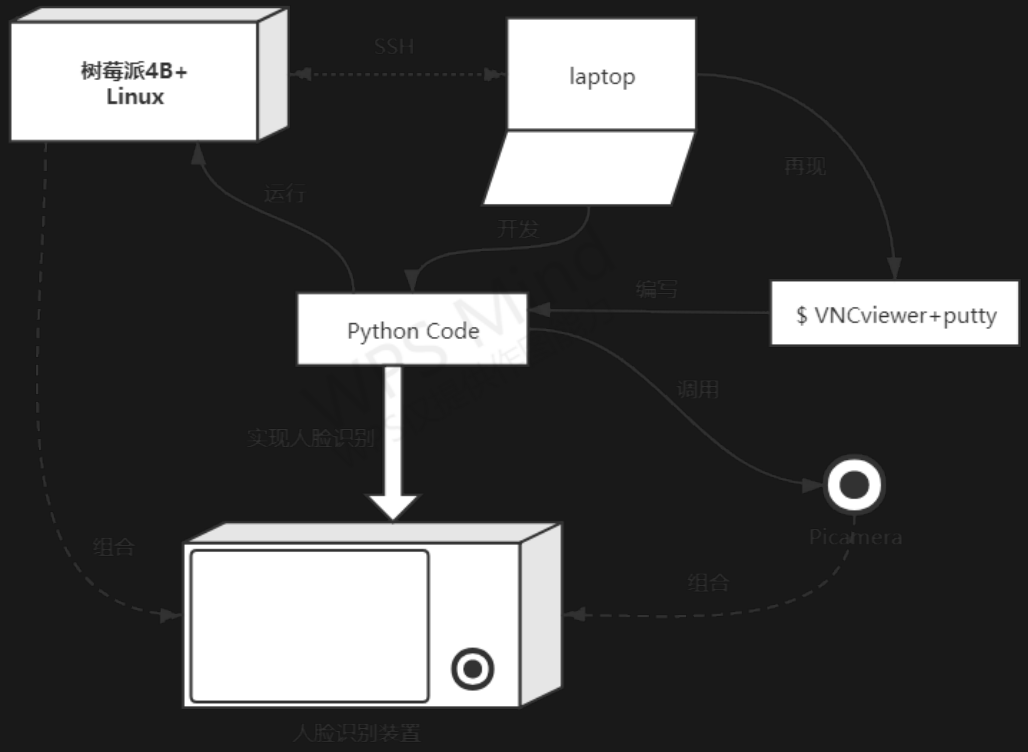


III. PRESENT YOUR IDEAS

- Creation and Presention

CREATION AND PRESENTION

- Everything creative: from your thoughts to the implementation...
- Show it to others: module graph & flowchart



THANK YOU!

- QQ: 1051276278
- https://github.com/raptazure