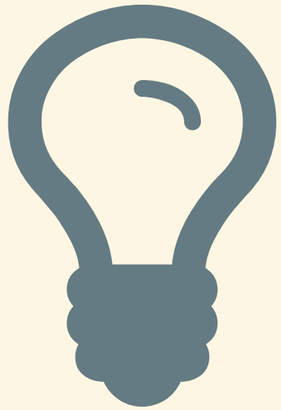
4. CONCEPTS IN CONTEXT

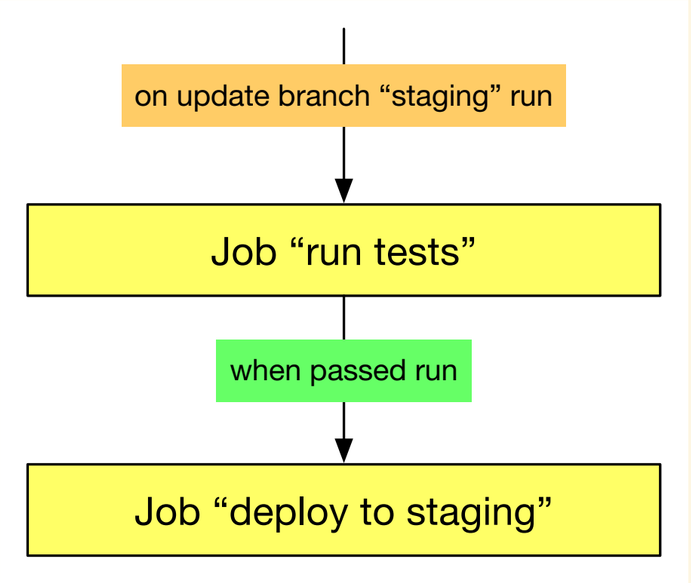


JOBS

EXAMPLES

- run test-suite
- perform static code checks
- build
- deploy

jobs can be **triggered** and can **depend on each other**



PROJECT CONFIGURATION

`cider-ci.yml` file in the project

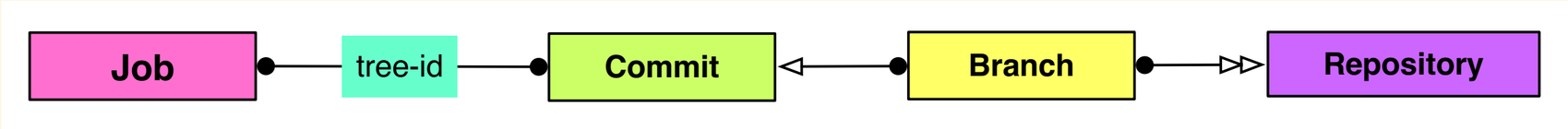
```
jobs:
  deploy_test:
    name: Deploy to test

    depends-on:
    - type: job
      job: integration-tests
      states: [passed]

    run-on:
    - type: branch
      include-match: ^master$

    # specify tasks etc
```

The source is the truth.

configuration: reproducible, reviews, audits ???

CIDER-CI AND THE SOURCE CODE

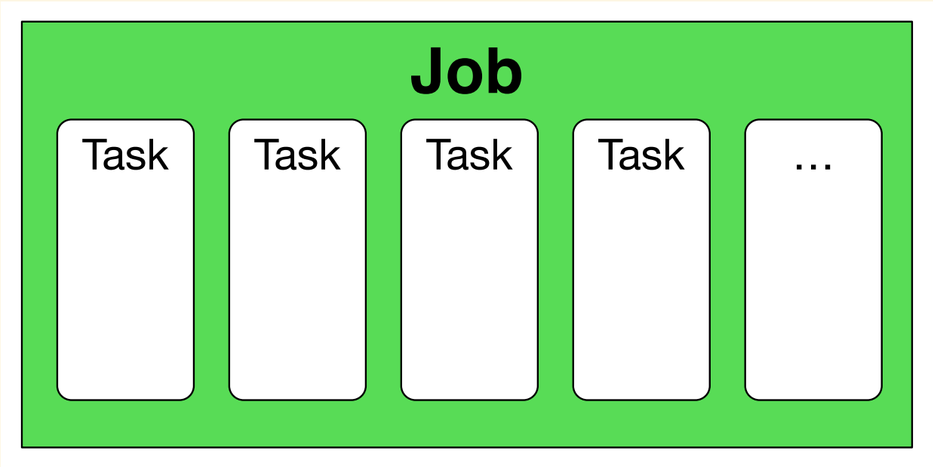
Cider-CI "knows" about commits, branches, submodules,…



`tree-id`: fingerprint of your source code

- **reproducibility**
- jobs can be **run at any time** (later)
- **binary search** for "bad" commits
- commit amends, squashing: **existing job remains valid**

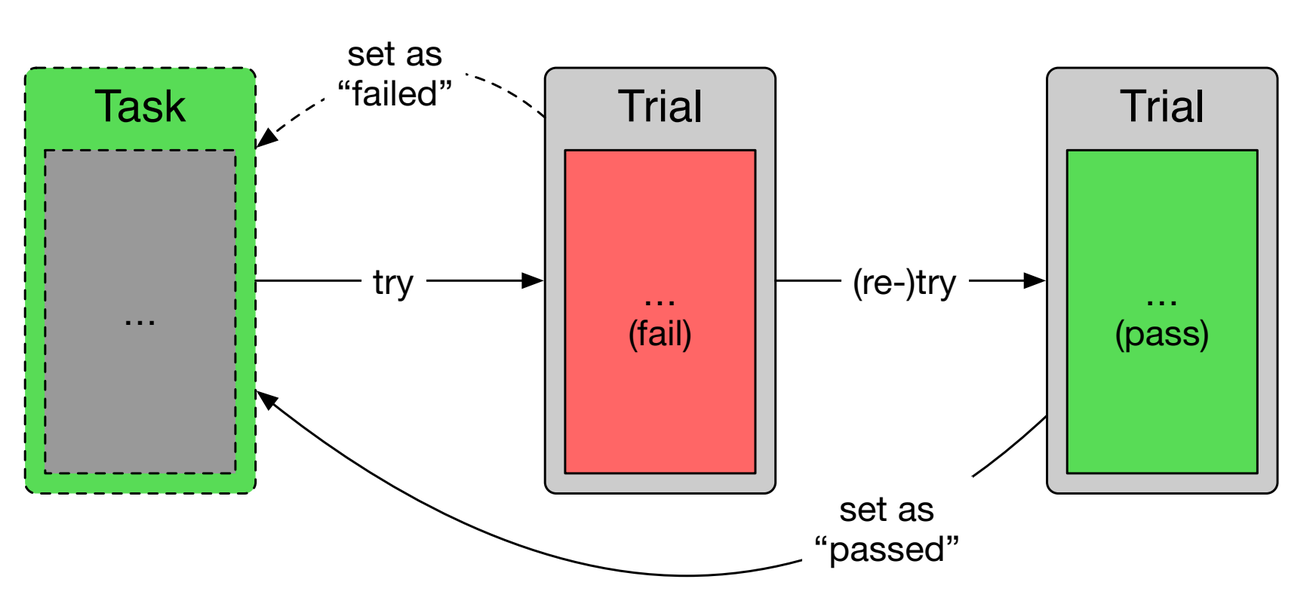
JOBS & TASKS



job: container and state aggregate for tasks

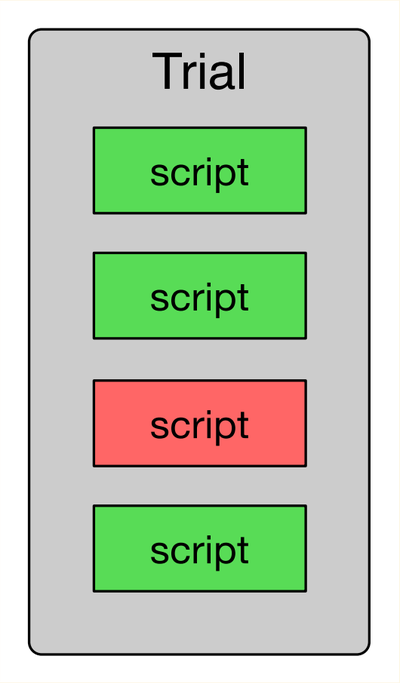
→ **parallelization**

TASKS & TRIALS



- **blueprint**
- container and state aggregate for trials

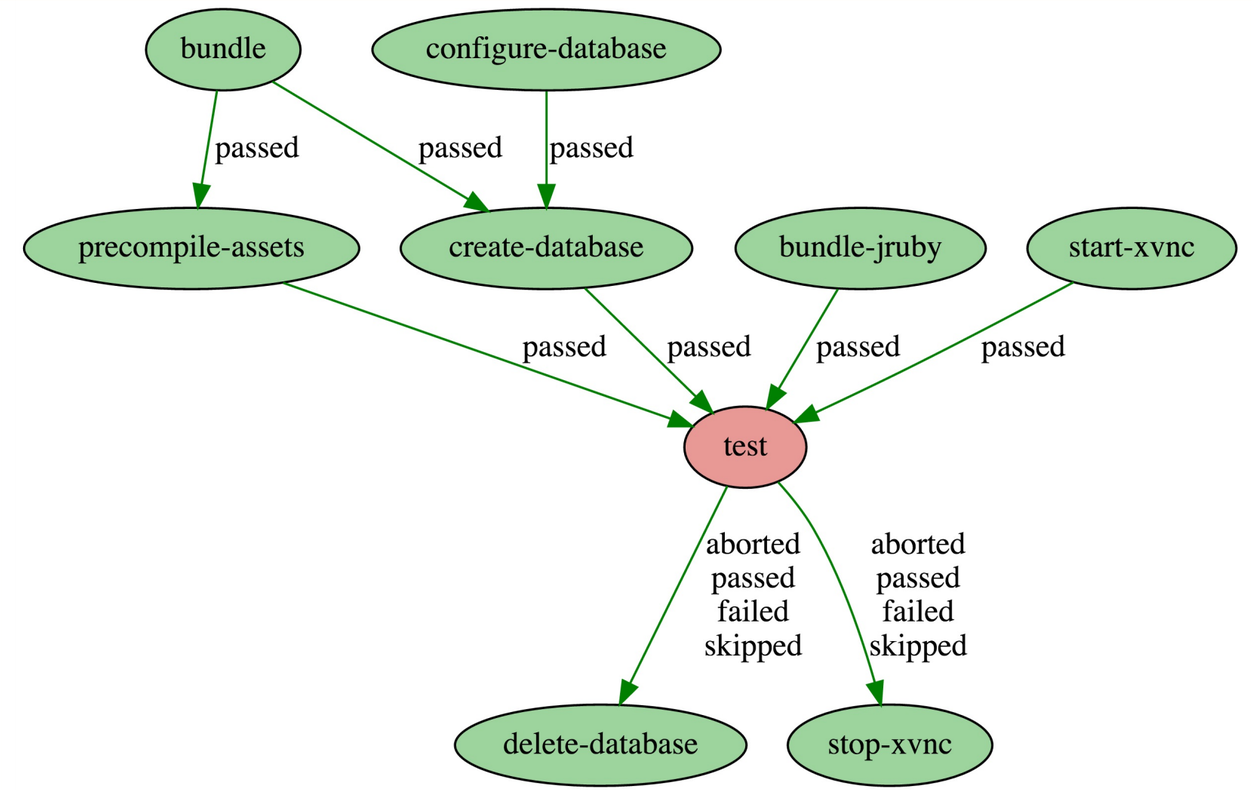
→ resilience



TRIAL & SCRIPTS

- actual unit of execution
- executed in the **same context**
- **depend** on each other

SCRIPT DEPENDENCIES



- traditional CI: one "build" ⇔ one script
- more modern: one main script + before and after "hooks"
- Cider-CI: **scripts with dependencies**