EFFECTIVE INTEGRATION TESTING

Andrew Rembrandt

@andrewrembrandt.bsky.social
https://rembrandt.dev/contact

SocratesCH '24

WHY & WHAT

- We've nailed effective unit testing
  - IT not understood, effective (or implemented 😞)
- ! End-to-End (E2E) tests
  - 1 app/service under test, with
    - 2nd & 3rd party components typically running in docker[-compose]

WHY & WHAT

- Benefits are often more significant that unit tests
- Immediate feedback when these are broken 🥳
    - SQL statements or ORM mapper (e.g. Hibernate entities)
    - HTTP parameters, [de]serialization, TCP/security setup
    - PubSub/Message-bus setup or messages
    - Component layer integration within the app/service

WHAT

- [Spring-]Slices or complete app/service?
  - Optimise for performance, e.g
    - @DataJpaTest, @WebMvcTest, & @Import / @TestConfiguration, common base class
  - Use core framework support
- Build tool support
  - Maven failsafe & skipITs parameter

WHY & WHAT

- Use 3rd party libraries & tools
  - TestContainers/docker-compose for internal & 3rd party external services (DB, pubsub, etc)
  - Bash scripts

COMMON MISCONCEPTIONS / MYTHS

- ITs aren't for frontend
- ITs are too slow to run in CI or Pull/Merge Requests
    - Can easily be made to run fast
        - Parallel
        - tmpfs for 3rd party containers
        - test slices
- ITs aren't 'pure' like unit tests

ESSENTIALS

- Like unit tests, ITs *must always* run in CI
  - PR/MRs will break them & neglect them if not
- At least one IT in every project that devs can copy/build upon
- Must be in the same codebase as the app/service code
  - Ideally alongside unit tests (=> Reuse test utils)
- Written in the same language as the app/service code

PEOPLE COMPLAIN, IT'S TOO HARD

- Less is more
- Easier than unit tests (!mocking)
- ChatGPT / AI code generation
- Functional coverage with less

CONTAINERS AND TOOLS

- Bash ~~shell~~ scripts
  - Loop till services have started
- Multiple java processes
  - Can easily debug 2nd party services
- nginx reverse proxy
  - Useful for frontend code having backend services available at the correct urls / paths

OPTIMISATION / SPEED

- Critical to run fast
- DB / 3rd party tools
  - Persist state with DB backups, or docker volumes
- Run in parallel
  - Ensure idempotency or non conflicting data between tests
    - Insert unique users / grouping elements for each test

EXAMPLE CODE

- Java (WIP)

  https://github.com/andrewrembrandt/effective-integration-testing-talk/

- Go

  TBP

QUESTIONS