Containerize Everything

Thinking in TEUs(*)

(*) Twenty-foot Equivalent Unit

Previously

- Most things in docker but not everything
  - docker-compose.yml
- Frequently a “bare metal” PostgreSQL
  - Needed during a build
  - May be configured oddly
  - “bleeds through”, i.e. not “the same for everyone”
- Many manual steps to “get going”
  - Easy to get sidetracked into debugging your environment

Changes

- docker-compose
  - ./prime-router/docker-compose.yml
  - ./prime-router/docker-compose.build.yml
- gradle | ./gradlew
- ./prime-router/build.sh
- ./prime-router/cleanslate.sh

docker-compose

- ./prime-router/docker-compose.yml
  - The ‘same’ you know and love
  - Creates/attaches itself to a (docker) internal network called ‘prime-router_build’
  - Uses this network for container-to-container communications within your composed environment
- ./prime-router/docker-compose.build.yml
  - The minimal set of (correctly configured) containers that you need to do a build
    - Really, only PostgreSQL – get rid of your bare metal PostgreSQL
    - But configured especially for us
  - docker-compose --file docker-compose.build.yml up --detach
  - (also) Creates/attaches to that ‘prime-router_build’ network, this is your runtime DB
- Warnings and ‘errors’ about orphan artifacts
  - Due to splitting over multiple docker-compose files; ignore them

docker-compose

- PostgreSQL
  - docker-compose --file docker-compose.build.yml up --detach
  - docker-compose --file docker-compose.build.yml down
- PRIME Router
  - docker-compose --file docker-compose.yml up --detach
  - docker-compose –file docker-compose.yml down
- Are things running?
  - docker container ls [-a]: list my containers; -a includes stopped containers if there are any
- Logs:
  - docker logs prime-router_prime_dev_1 [--follow]
  - docker logs prime-router_postgresql_1 [--follow]
- Attaching
  - docker exec -it prime-router_prime_dev_1 bash: bash into that running container

gradle | ./gradlew

- Does not change
- Keep using it the way you are using it

./prime-router/build.sh

- “These are (most likely) not the droids you are looking for”
  - Only intended for the CI/CD pipeline
  - If you want to build *as if* you are the CI/CD pipeline
  - e.g.: the pipeline fails but your box doesn’t
- The build happens *inside* a (controlled) container
- The build artifacts are located *outside* the container
- Benefits
  - Repeatability: The box can be recreated over and over and over again
  - Reproducibility: anyone can build in the same environment as the pipeline
  - Supply Chain Management: we know what’s “on the box” as opposed to what is in GitHub’s image
- Requires elevation (i.e. will prompt you for sudo password)
  - Because: id(container($USER)) != id(local($USER)) and both stat.st_uid and stat.st_mode store just integers
  - To play nice with ‘local’ gradle invocations, does a chown -R $YOU & chmod -R a+rw on anything a build touches
  -

./prime-router/cleanslate.sh

- First *Port-of-Call* if your environment gets messed up
- Does what it says on the tin: wipes the slate clean
  - Cleaning up containers is easy
  - Cleaning up apps running on bare metal is … riskier
- Requires that you run things in containers and not on bare metal
  - I'm (lovingly) looking at you, PostgreSQL
- "Destroys" everything you tell it to destroy, and then brings it up again while satisfying all dependencies
- Two Functions
  - Onboarding: sets up everything, including containers, vault, a first build of bits and container, instantiate the router, front to back
  - Wipe your slate clean: recover from a bad state, pretend I'm a new developer
- ./cleanslate.sh --help will show you the way
  - There's also ./cleanslate.sh --instructions for post-run instructions
- ./prime-router/cleanslate.sh.log: diagnostics
- Repeatable but takes a bit to run because it does what it says it will do