12 FACTOR APPLICATION DEVELOPMENT

Frank P Moley III

ABOUT ME

- Software Architect with 15+ experience in all phases of SDLC

- Focus on web services, security, privacy, and education

- Content author for Lynda/LinkedIn

- Internet Architect at Garmin

- @fpmoles & github.com/fpmoles

WELCOME TO THE "CLOUD"

- Most impactful change in recent history

- Global availability

- Highly scalable and available

- Limited human resources

MONOLITHIC APPS

- Extended deployment times

- Long break/fix cycles (see above)

- Difficult to scale

- Difficult to isolate changes

- Ops sees container, not application

CLOUD NATIVE APP

- Horizontal Scalability

- Rapid delivery

- Ops friendly

- Better supports agile development

12 FACTORS

- The 12 Factor methodology is an observed pattern

- Very opinionated, note I don't agree with all the opinions

- Developed out of work at Heroku by Adam Wiggins ->
  Documented at 12factor.net

1. CODEBASE

- Single root commit per application

- Single codebase per application

- Must use VCS -> flavor doesn't matter

- Branching model should support development needs

- Supports CI-CD and agile development

2. DEPENDENCIES

- All dependencies encapsulated in app

- NO FILE SYSTEM dependencies

- Remove version change breaks

- Dependencies declared in manifests

- Improves portability

3. CONFIG

- What is config?

- Must be externalized

- Provided to the application at runtime

- Allows you to deploy to different data centers and environments

- Can increase complexity for legacy systems

4. BACKING SERVICES

- What are backing services?

- Treat them as local resources

- Backing services binding

- Supports ops and portability

5. BUILD, RELEASE, RUN

- Distinct phases of build and deploy

- Not CD but CD helps

- Build takes changes from SCM

- Release binds the build with a config

- Run stage is just taking the release and executing

6. PROCESSES

- Each application is run as its own process

- Not run in another process!!!

- Memory considerations

- No Sticky Sessions!!!!

- Supports ops and dev communication

7. PORT BINDING

- Each application is bound to a communication port at runtime

- Framework support

- Local running vs Container running

8. CONCURRENCY

- Ever tried to scale a monolith?

- Horizontal concurrency

- Consider your backing services

9. DISPOSABILITY

- Processes can be started and stoped at will

- Crashes happen

- Clean up on shutdown, control crashes

- Startup time is important

10. DEV/PROD PARITY

- Anyone ever have a bug that only exists in prod?

- Keep it all the same

- May need to reduce the number of environments

11. LOGS

- Logs are streams

- No file system so standard out only

- Consider log aggregators

- Timestamps and trace ids

12. ADMIN PROCESSES

- 1st class citizens

- Triggered by events

- Act like every other 12 factor app

WHY

- You don't have to rely on Cloud providers

- Can be run in on traditional VMs

- Allows you to support growth

QUESTIONS

- @fpmoles

- linkedin.com/frankpmoley

- github.com/fpmoles