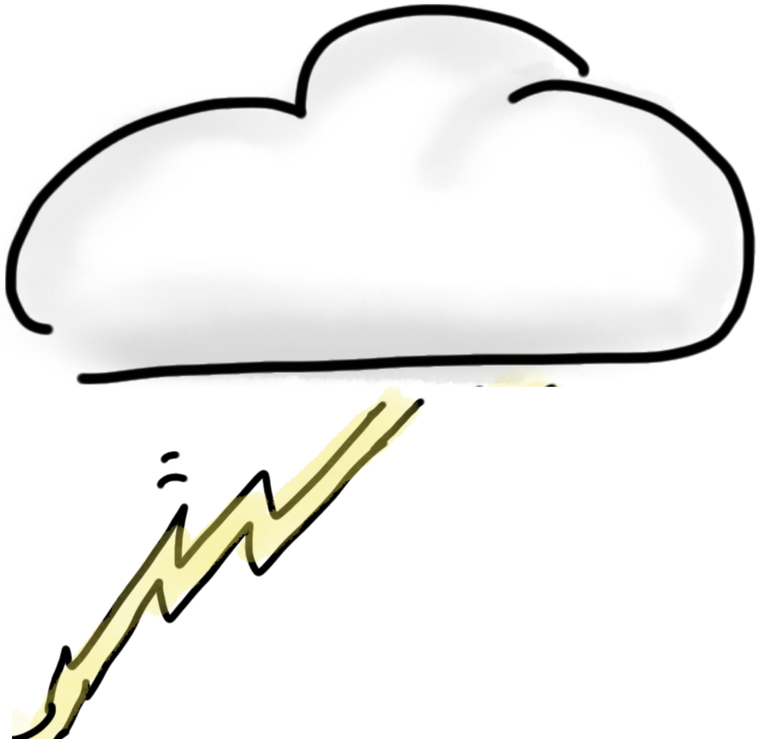
the adventurer’s guide to **breaking production**

Holly Cummins
@holly_cummins



me: “innovation leader” at IBM

me: “innovation leader” at IBM

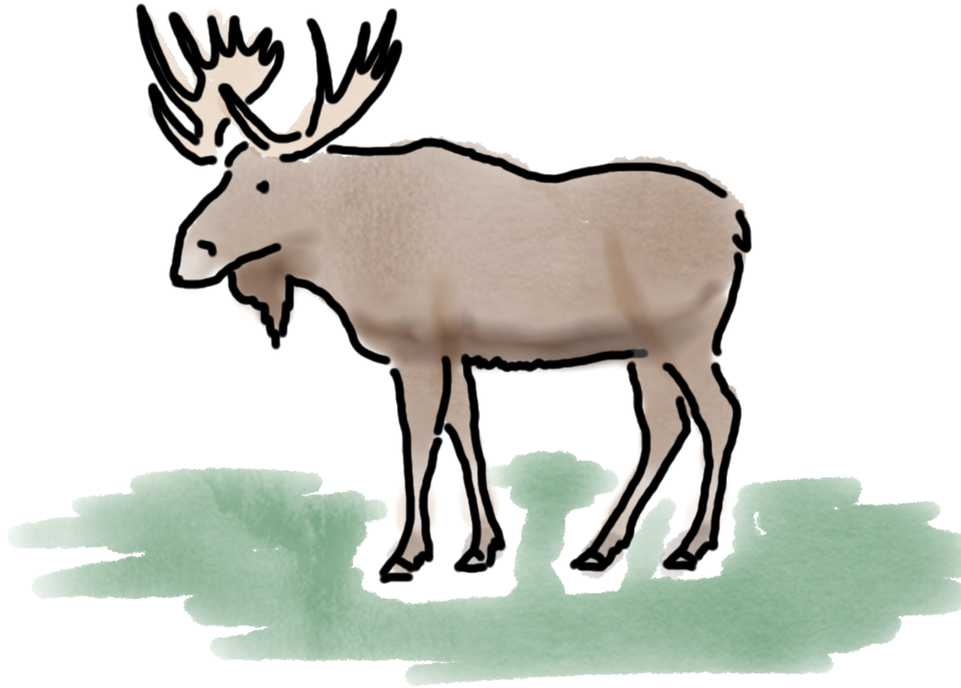
translation: getting into trouble with technology.

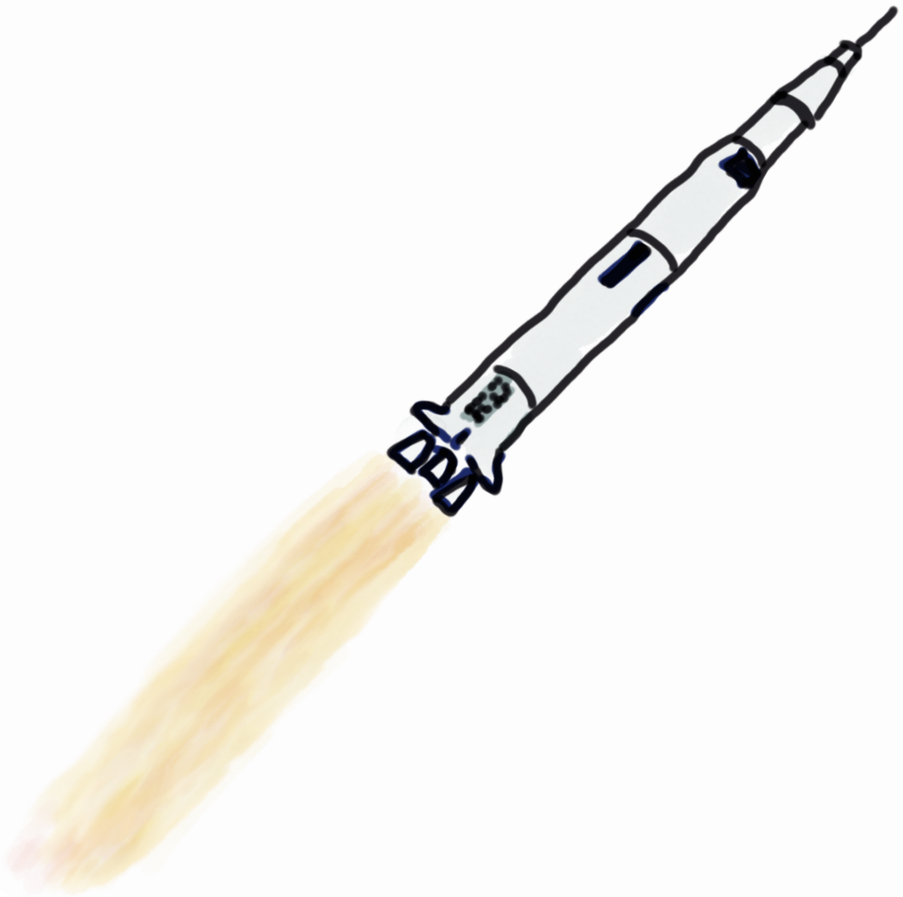
me: “innovation leader” at IBM

translation: getting into trouble with technology.

... for 20 years

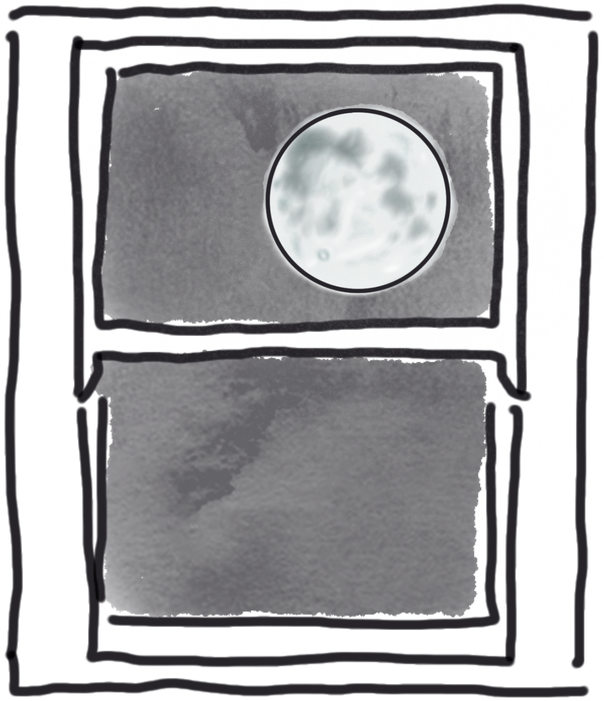
how often do we get to
go someplace truly new?





see things we've never seen before?

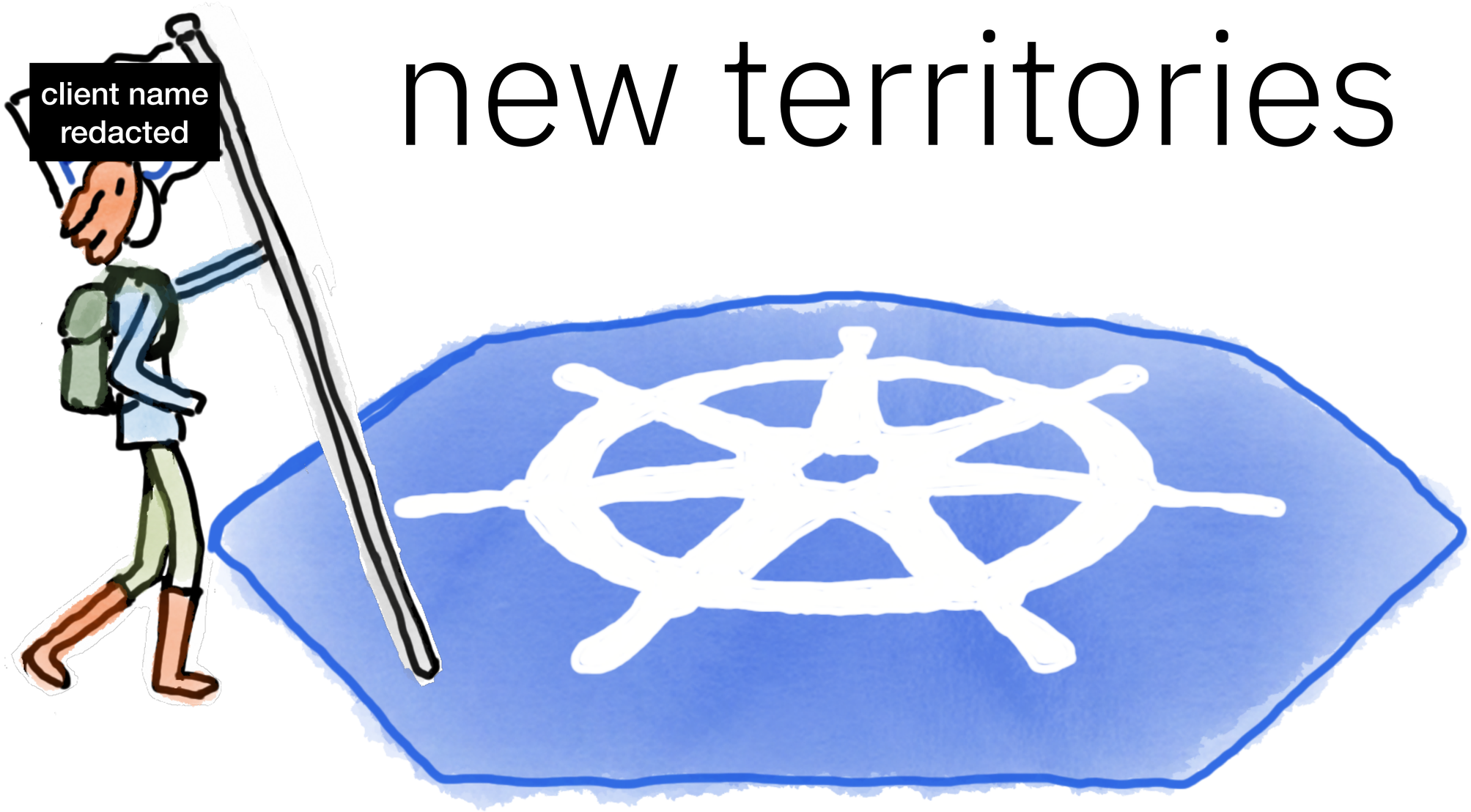
in software we do it all the time



ancient landscapes

legacy environments

new territories

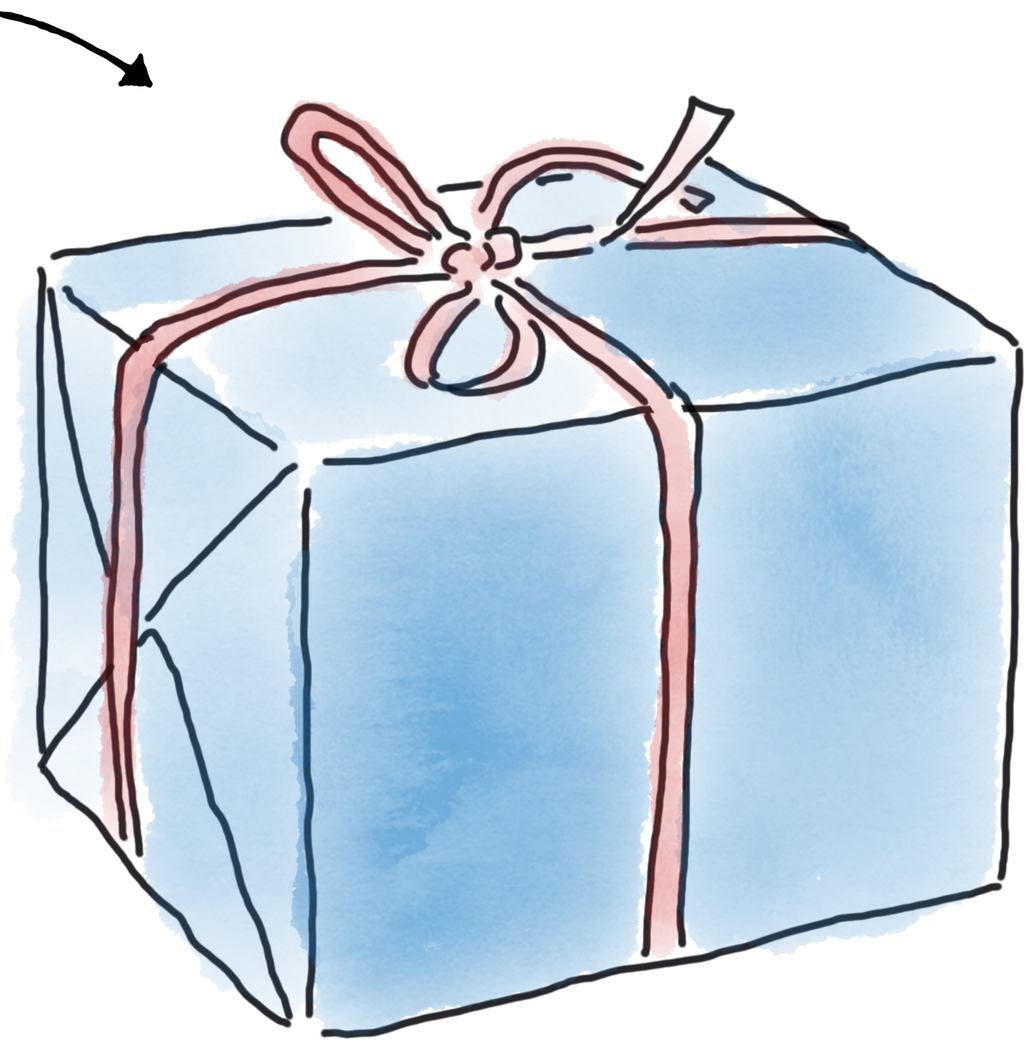


what could possibly go wrong?

um ... what problem were we trying to solve?

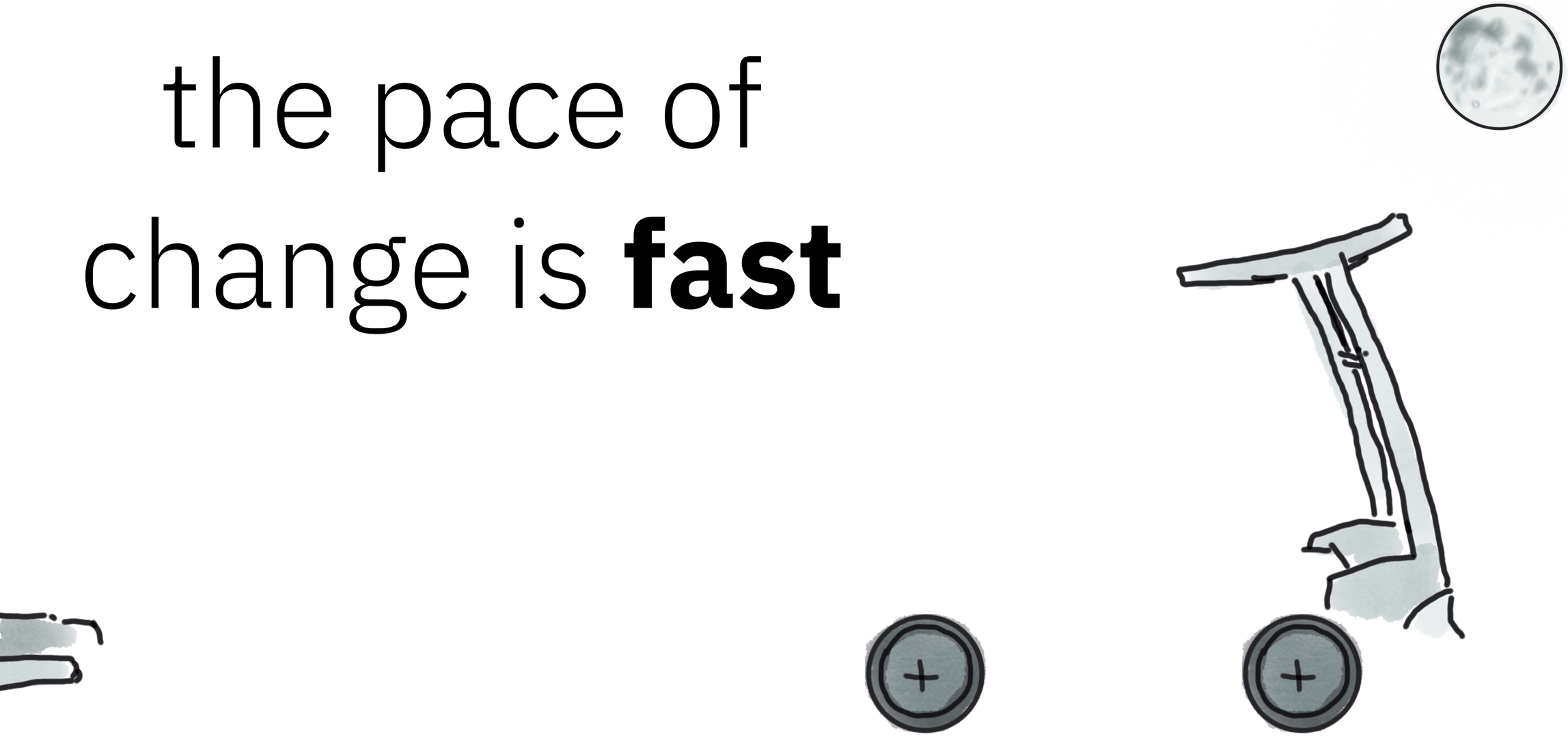

George Mallory
Died on Everest...
30 years before Edmund Hillary
because it’s there
new for the sake of it

surprise inside

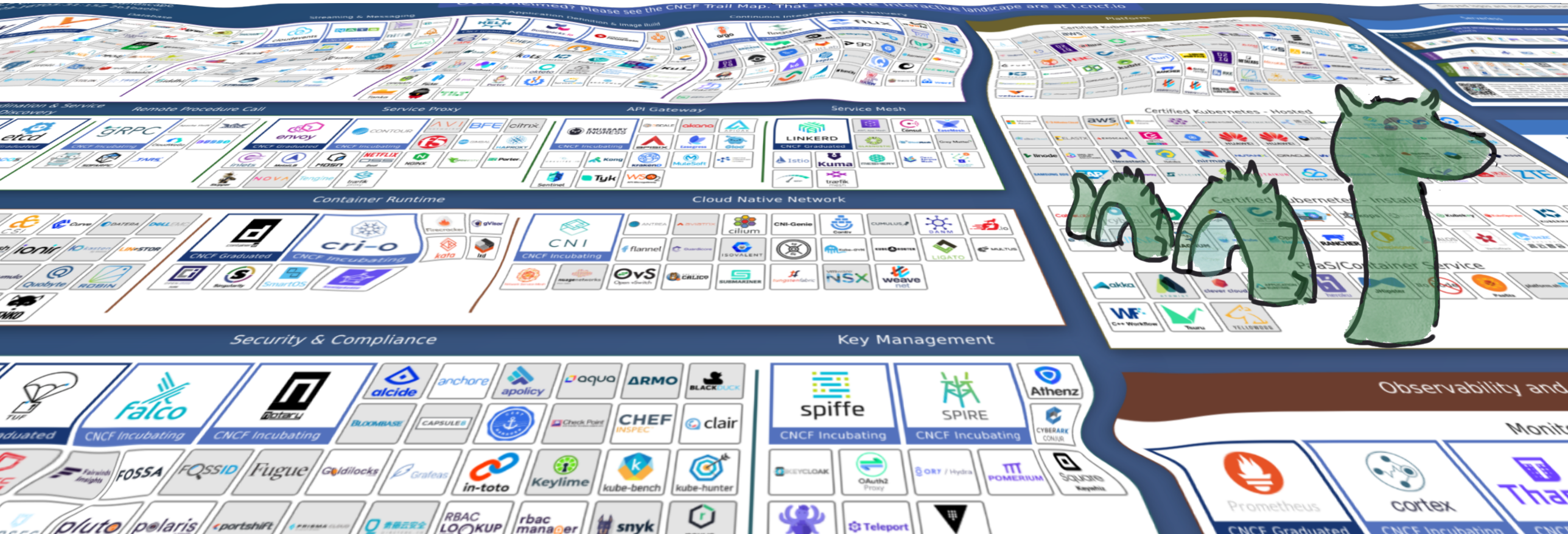


the pace of change is **fast**

the pace of change is **fast**

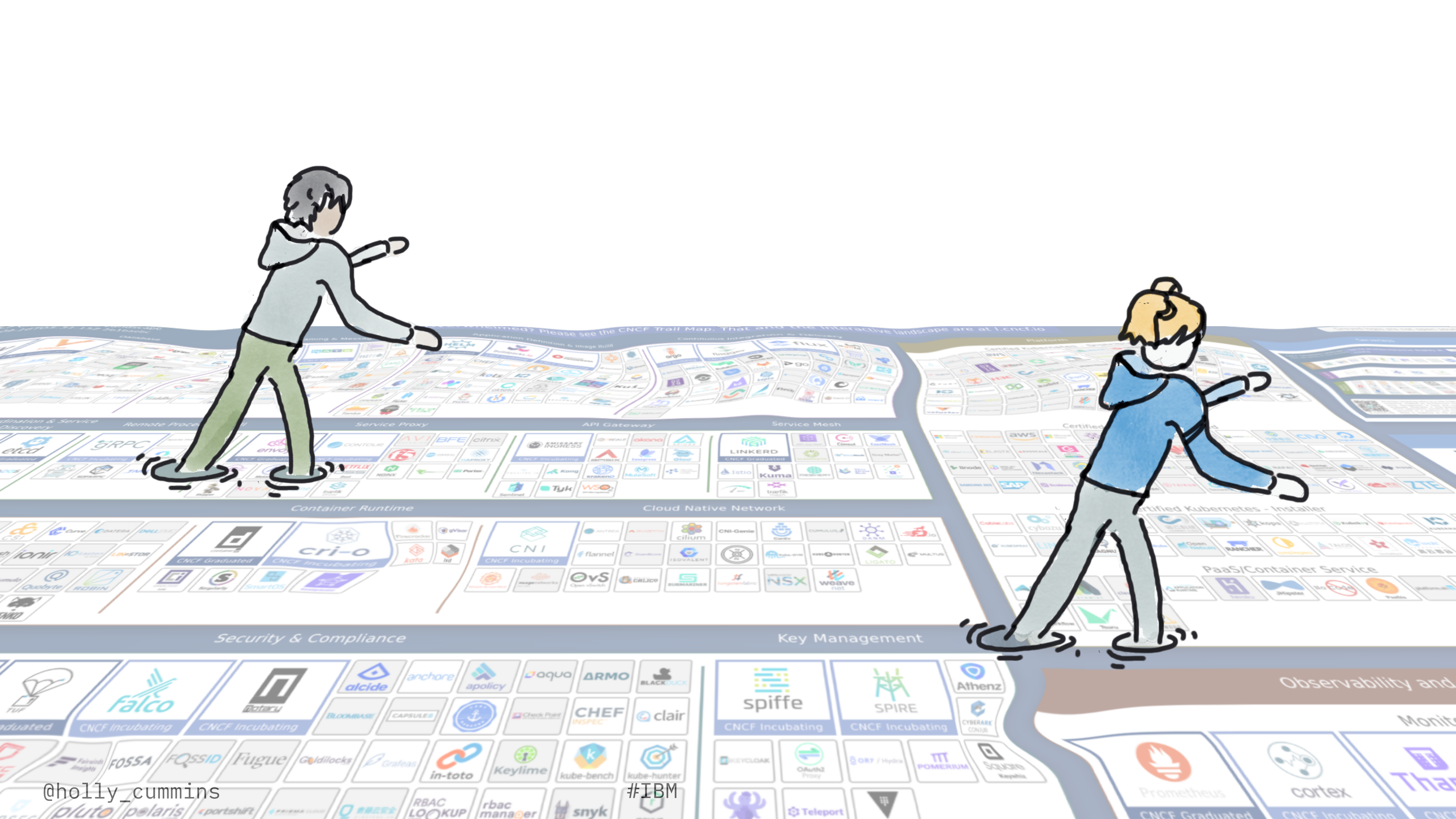


the landscape is complicated



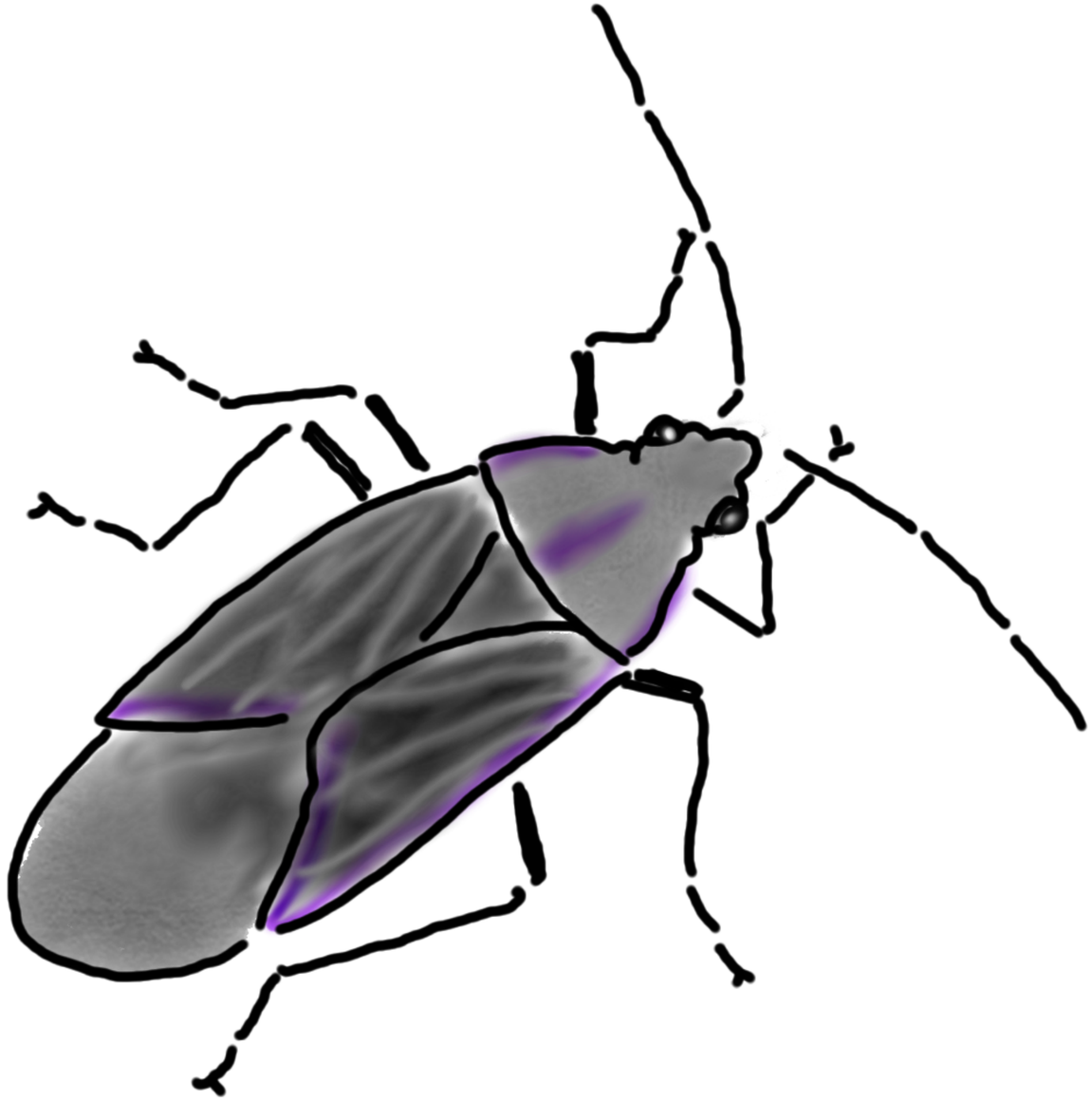
the old stuff hasn’t gone away

stuff slows us down


Service Proxy
API Gateway
Service Mesh
Container Runtime
Cloud Native Network
Security & Compliance
Key Management
CNCF Graduated
CNCF Incubating
PaaS/Container Service
Observability and
@holly_cummins
#IBM


Cisco Webex Meetings
Meeting Info
Show menu bar
Layout
all your colleagues

and ...

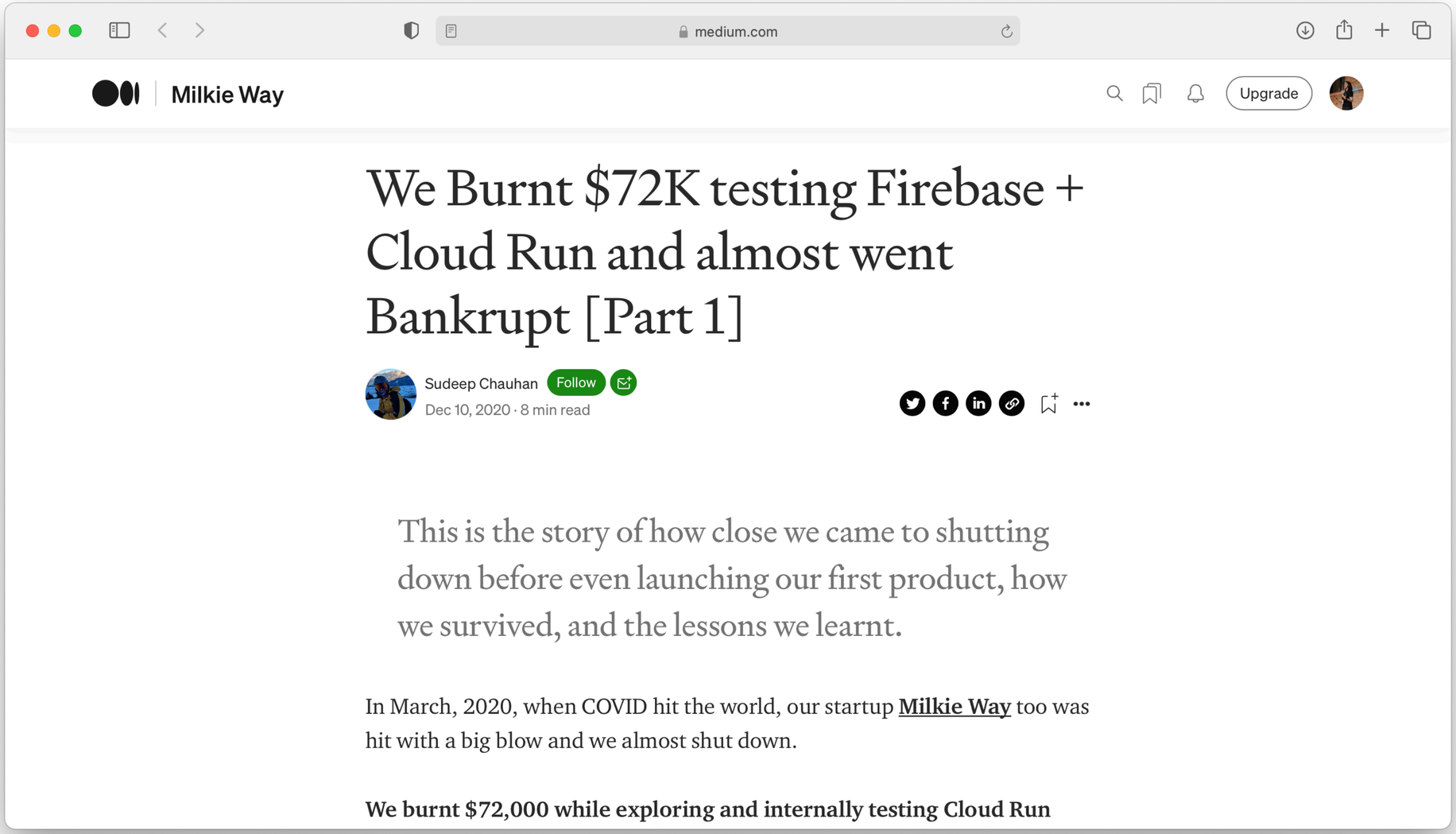


ok but what’s the worst that could go wrong?

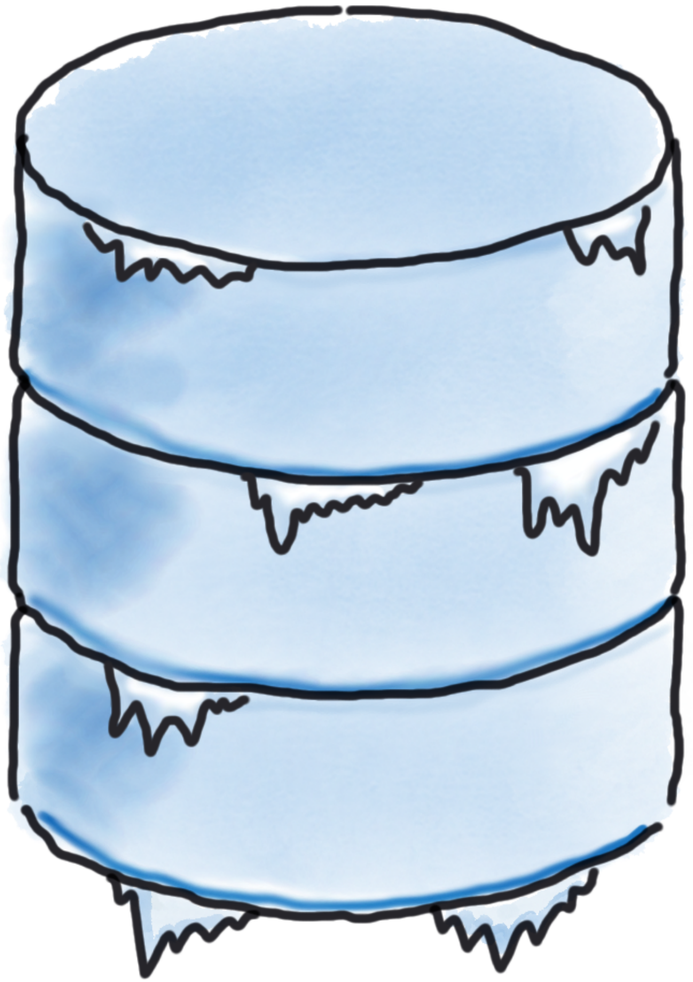
$460 million loss

$460 million loss

in 45 minutes


medium.com
Milkie Way
Upgrade
We Burnt $72K testing Firebase + Cloud Run and almost went Bankrupt [Part 1]
Sudeep Chauhan
Follow
Dec 10, 2020 · 8 min read
This is the story of how close we came to shutting down before even launching our first product, how we survived, and the lessons we learnt.
In March, 2020, when COVID hit the world, our startup Milkie Way too was hit with a big blow and we almost shut down.
We burnt $72,000 while exploring and internally testing Cloud Run

the million-dollar frozen database



Hey boss, I created a Kubernetes cluster.

true story, unfortunately

Holly

Hey boss, I created a Kubernetes cluster.

I forgot it for 2 months.

Holly

true story, unfortunately

true story, unfortunately

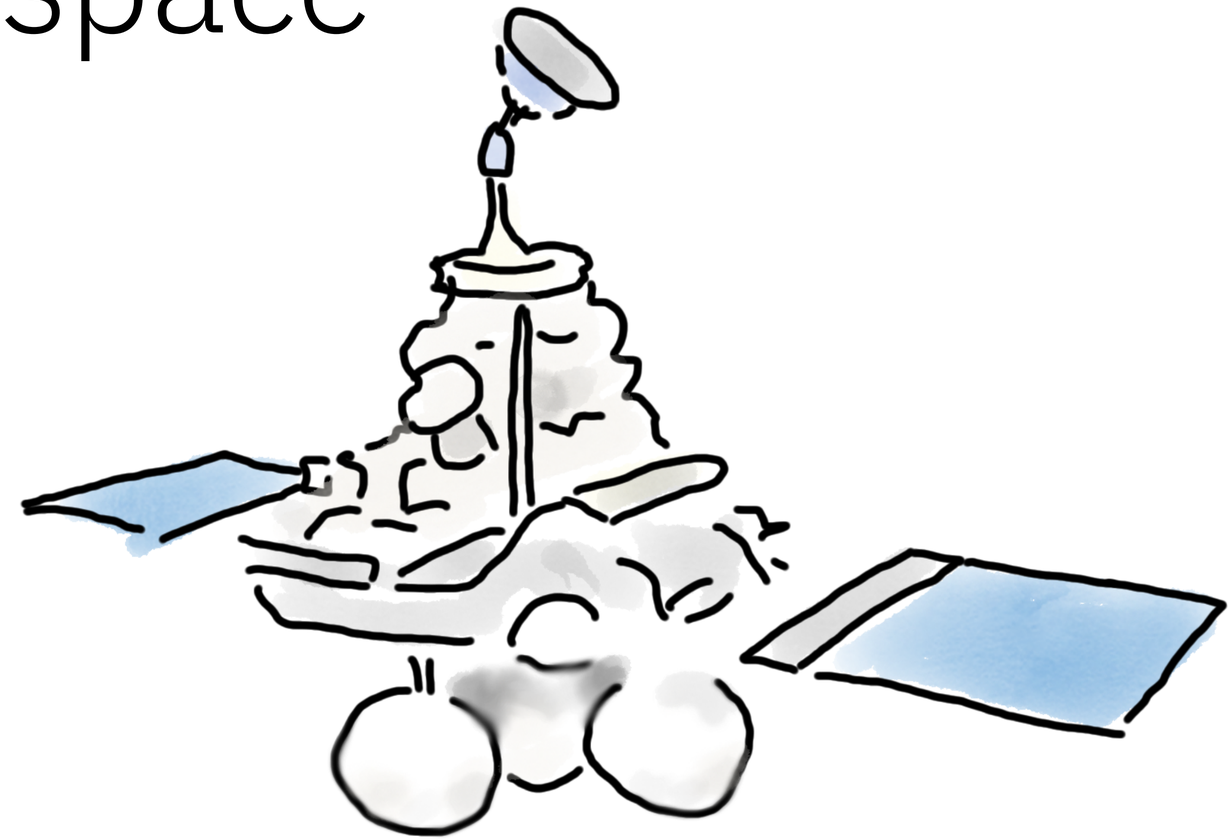
Hey boss, I created a Kubernetes cluster.

I forgot it for 2 months.

... and it’s £1000 a month.

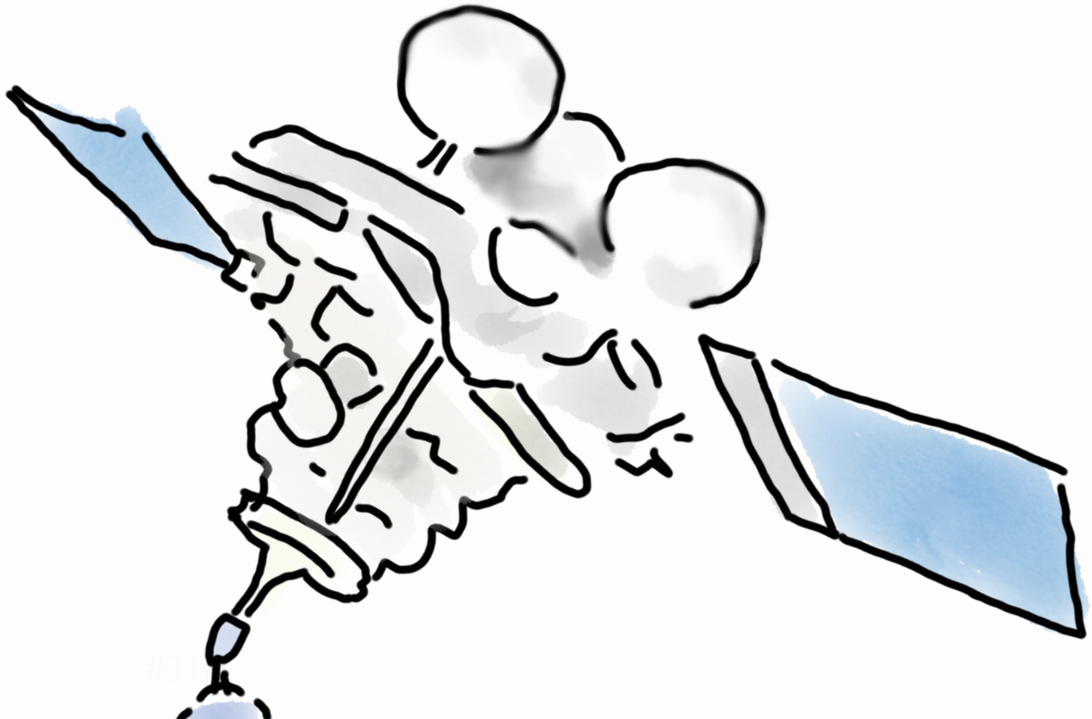
Holly

bugs, but in space



Phobos 1

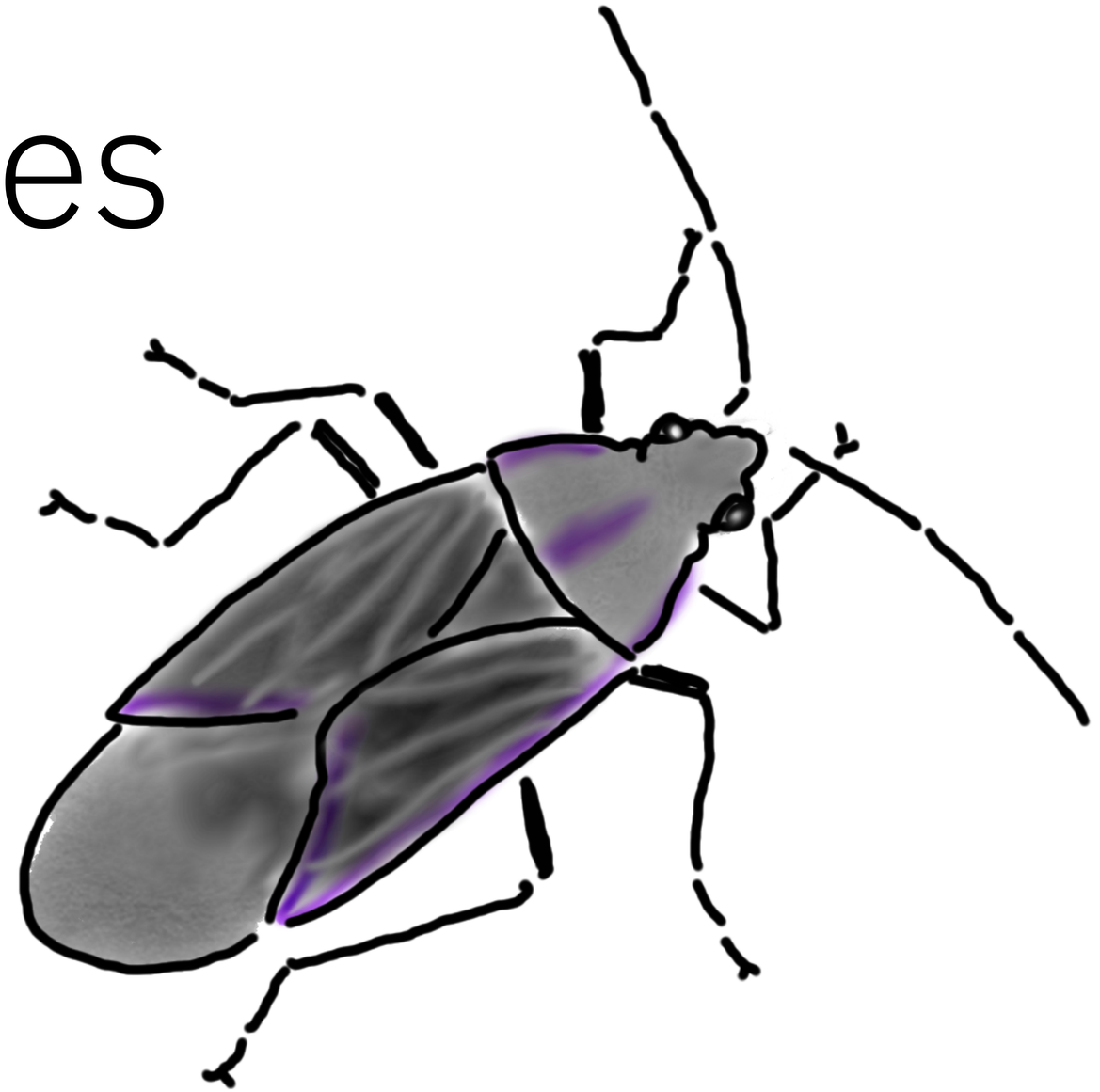
“we couldn’t get the automated checks to work, so we bypassed them”



“the space probe is bricked.”

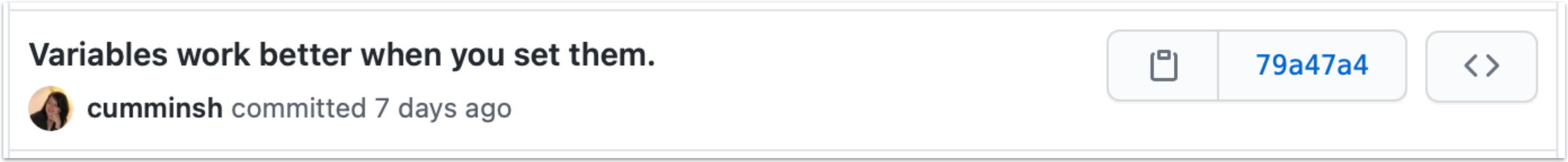
@holly_cummins

what causes bugs?



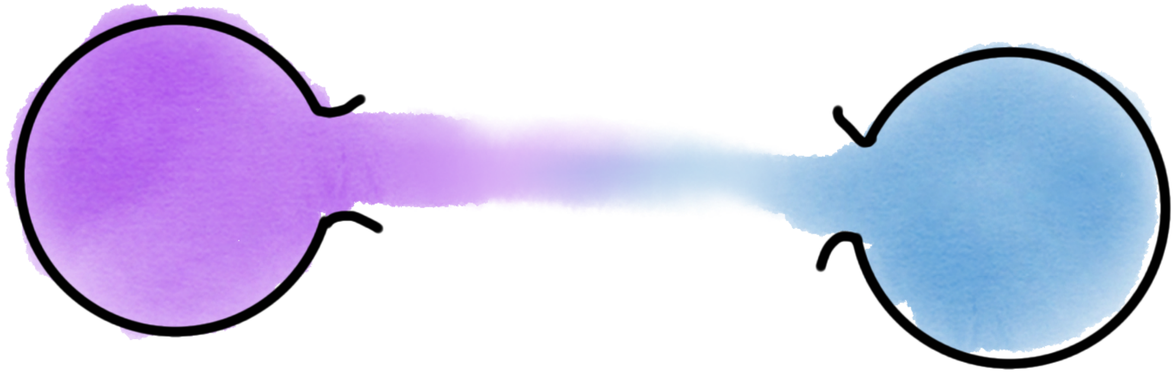
other people

us


Variables work better when you set them.
cumminsh committed 7 days ago
79a47a4

interactions

“every time we change one microservice, another breaks”



distributed != decoupled

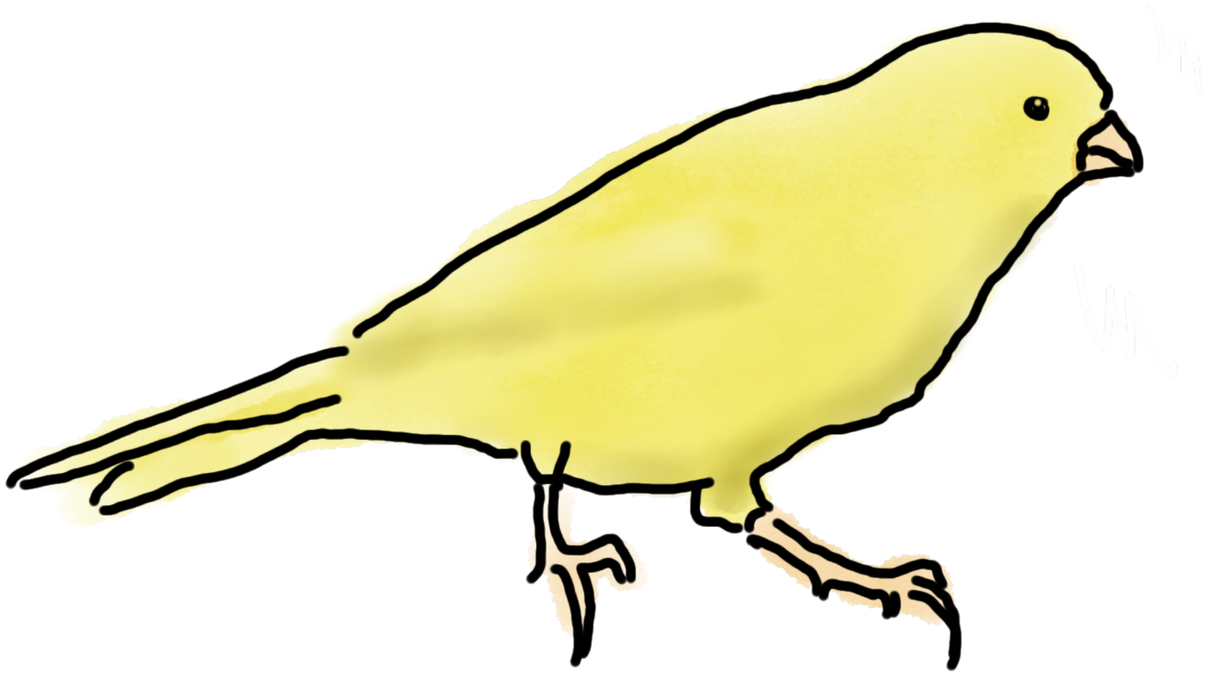
managing bugs

breaking production isn’t the worst thing

as long as it’s a small break

as long as it’s a tiny break

limit blast radius



canary deploys

breaking production isn’t the worst thing

the important thing is how fast you can **unbreak** production

recoverability

unrecoverable

unbreak

unbreak

diagnose

unbreak

diagnose

deploy

unbreak

diagnose

deploy

observability

unbreak

diagnose

deploy

observability

devops

unbreak

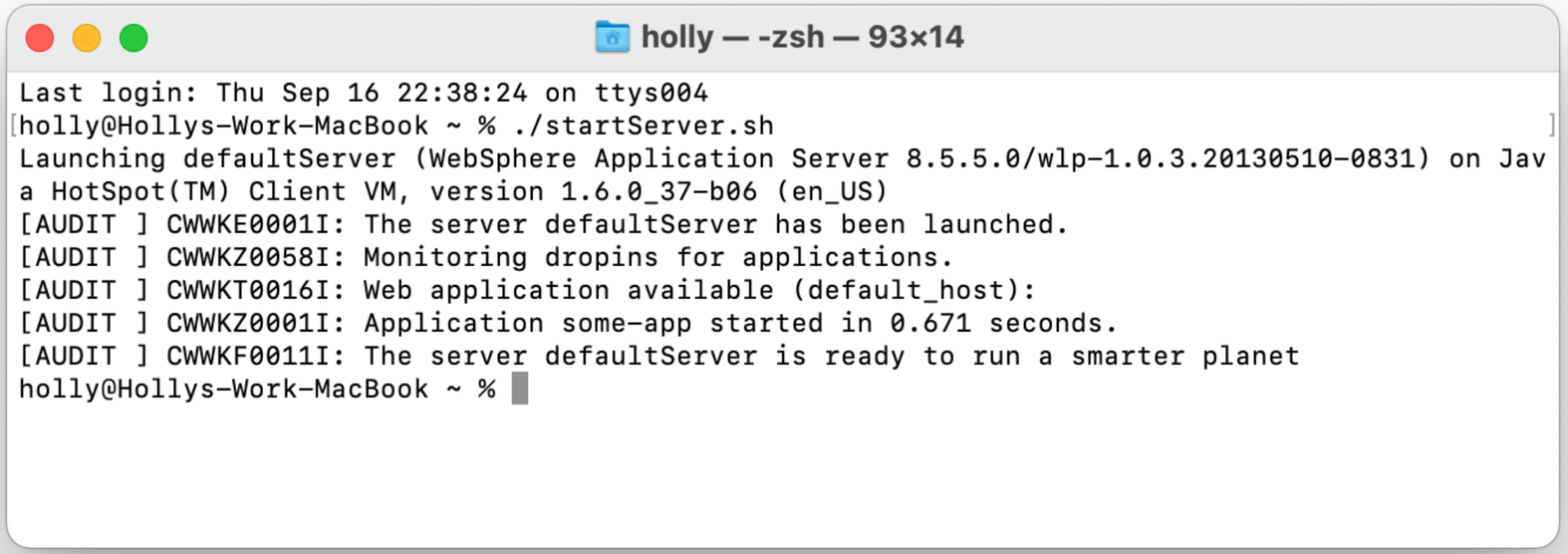
diagnose

deploy

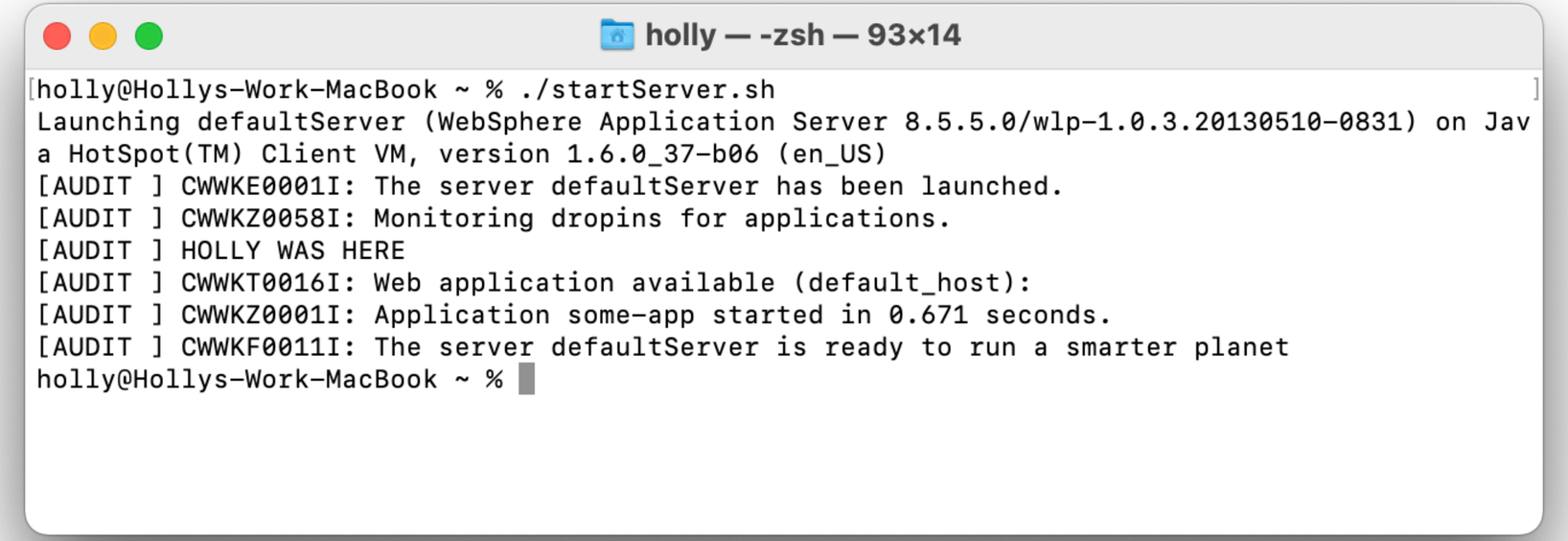
observability

devops

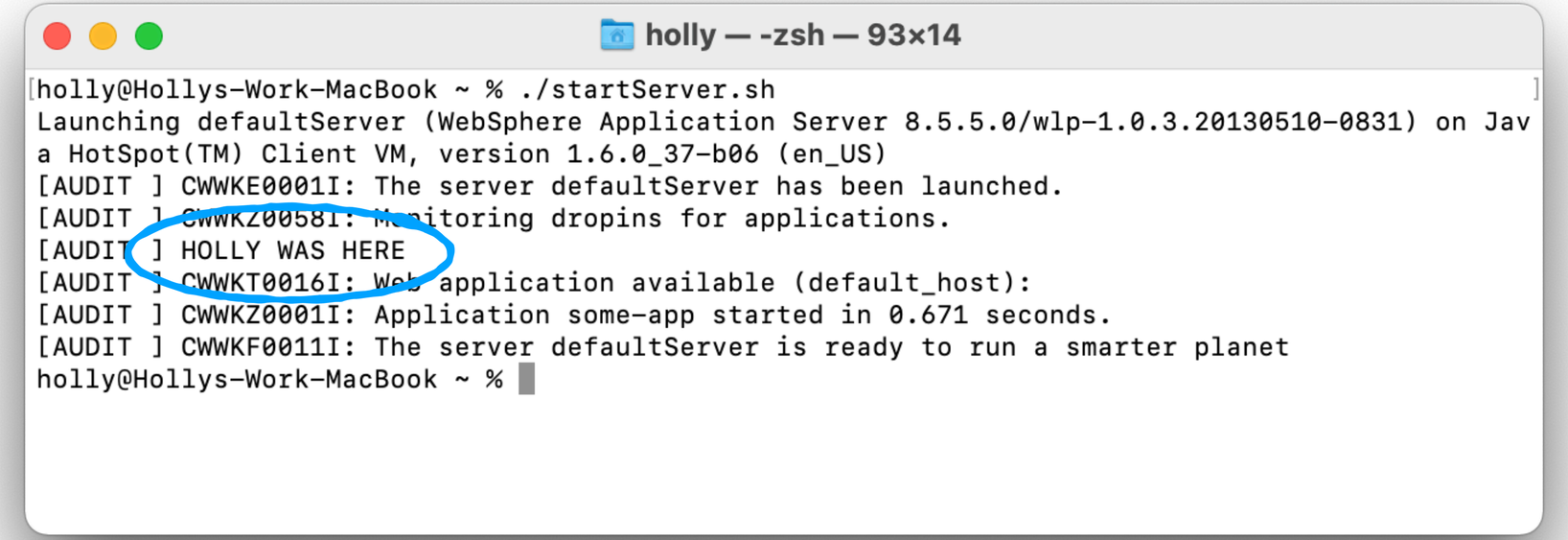
my most embarrassing break of production



my most embarrassing break of production



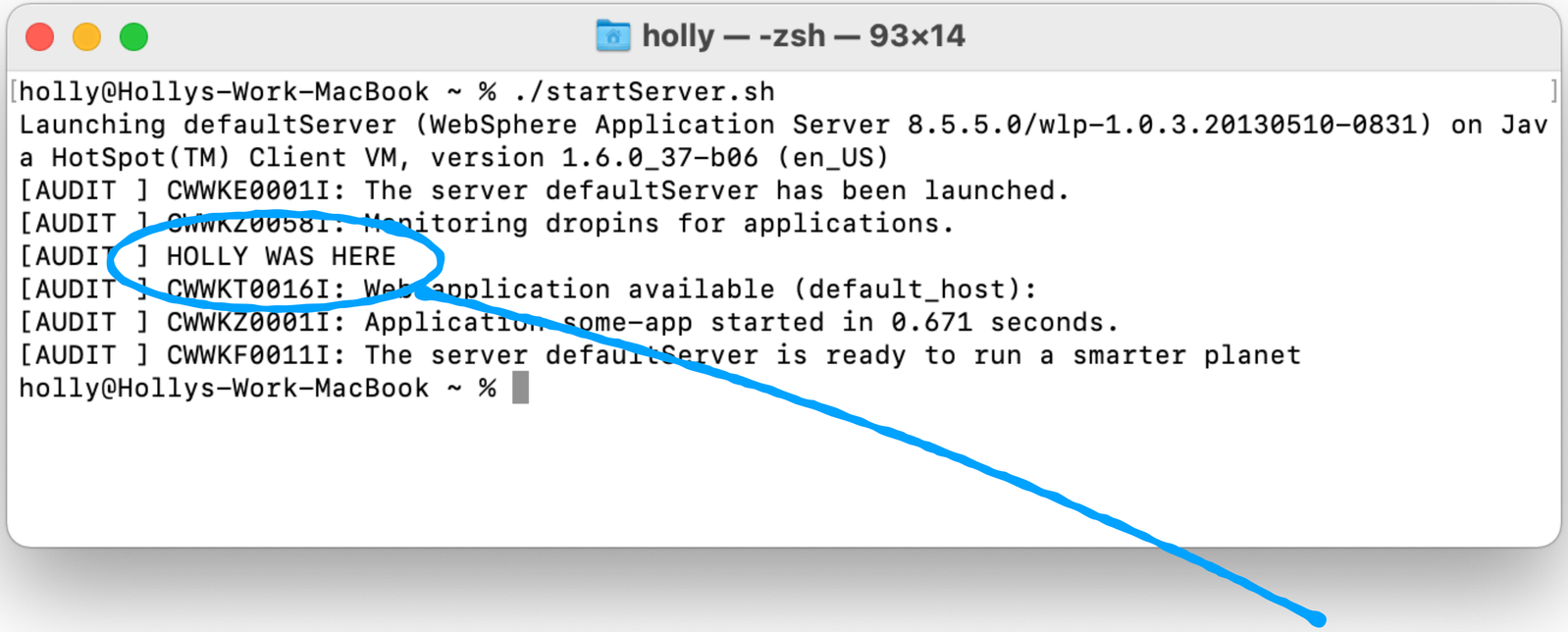
my most embarrassing break of production



most problems are harder to diagnose

observability

observability



what you don't have to do
... if you have observability

unbreak

diagnose

deploy

observability

devops

make

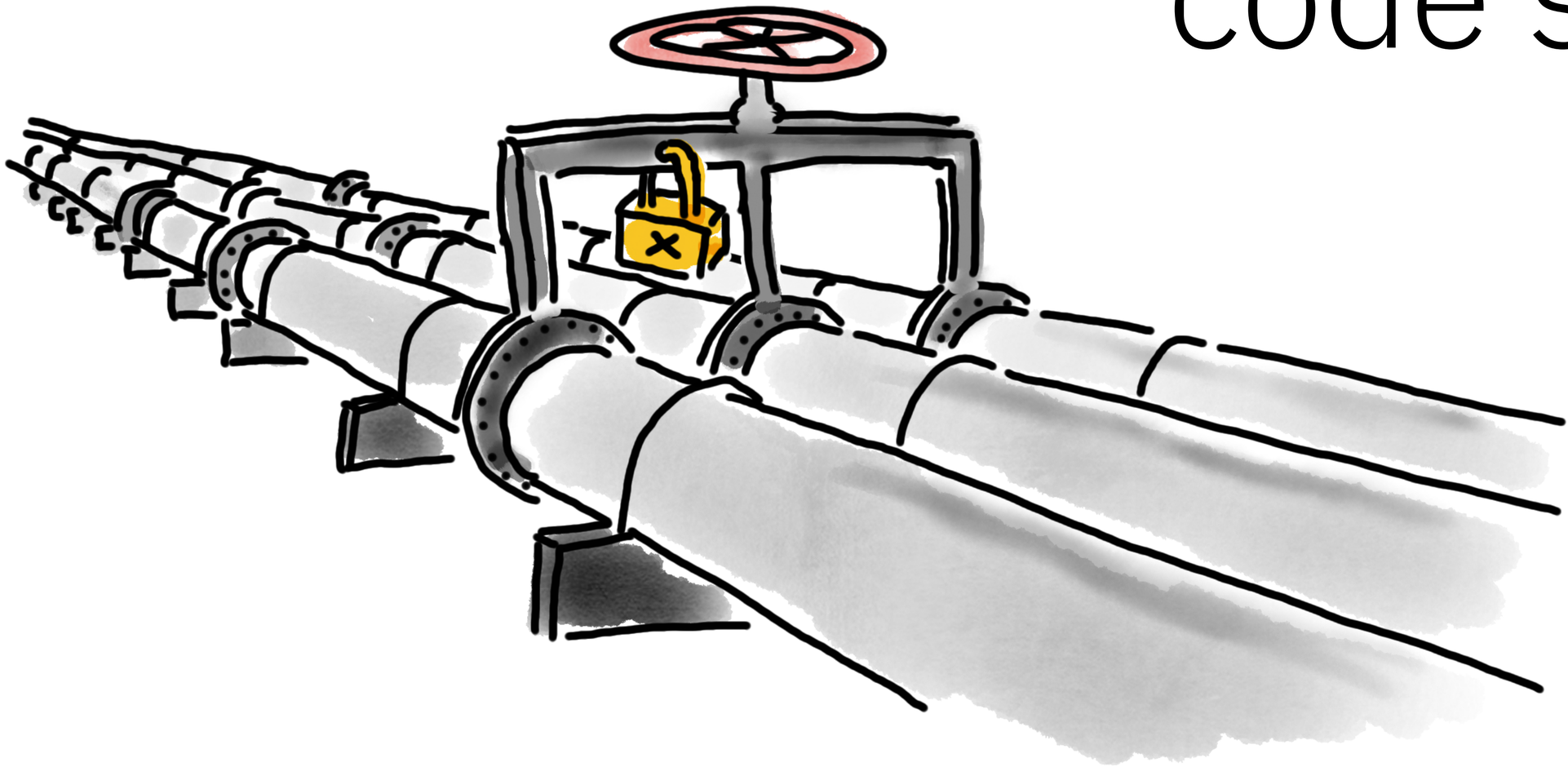
releases

deeply **boring**

make releases deeply **boring**

so you can do them **all the time**

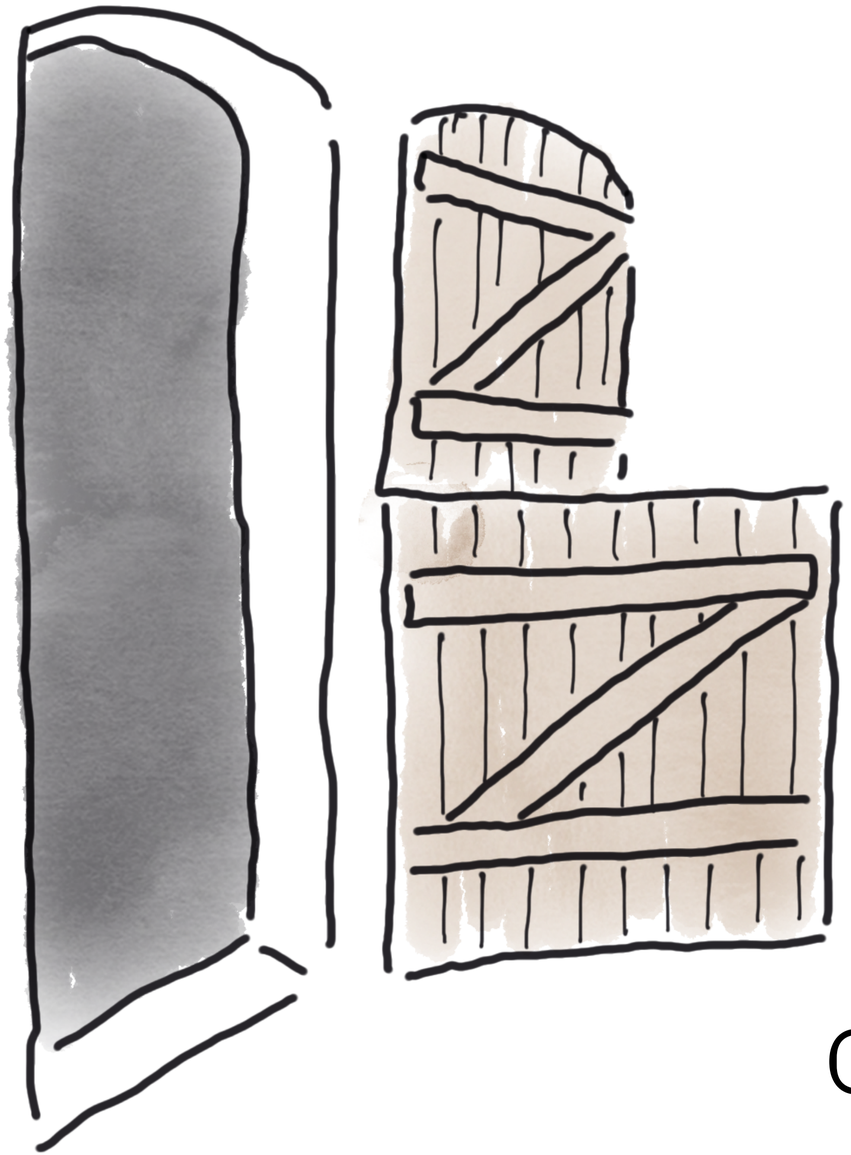
CI/CD first
code second



GitOps

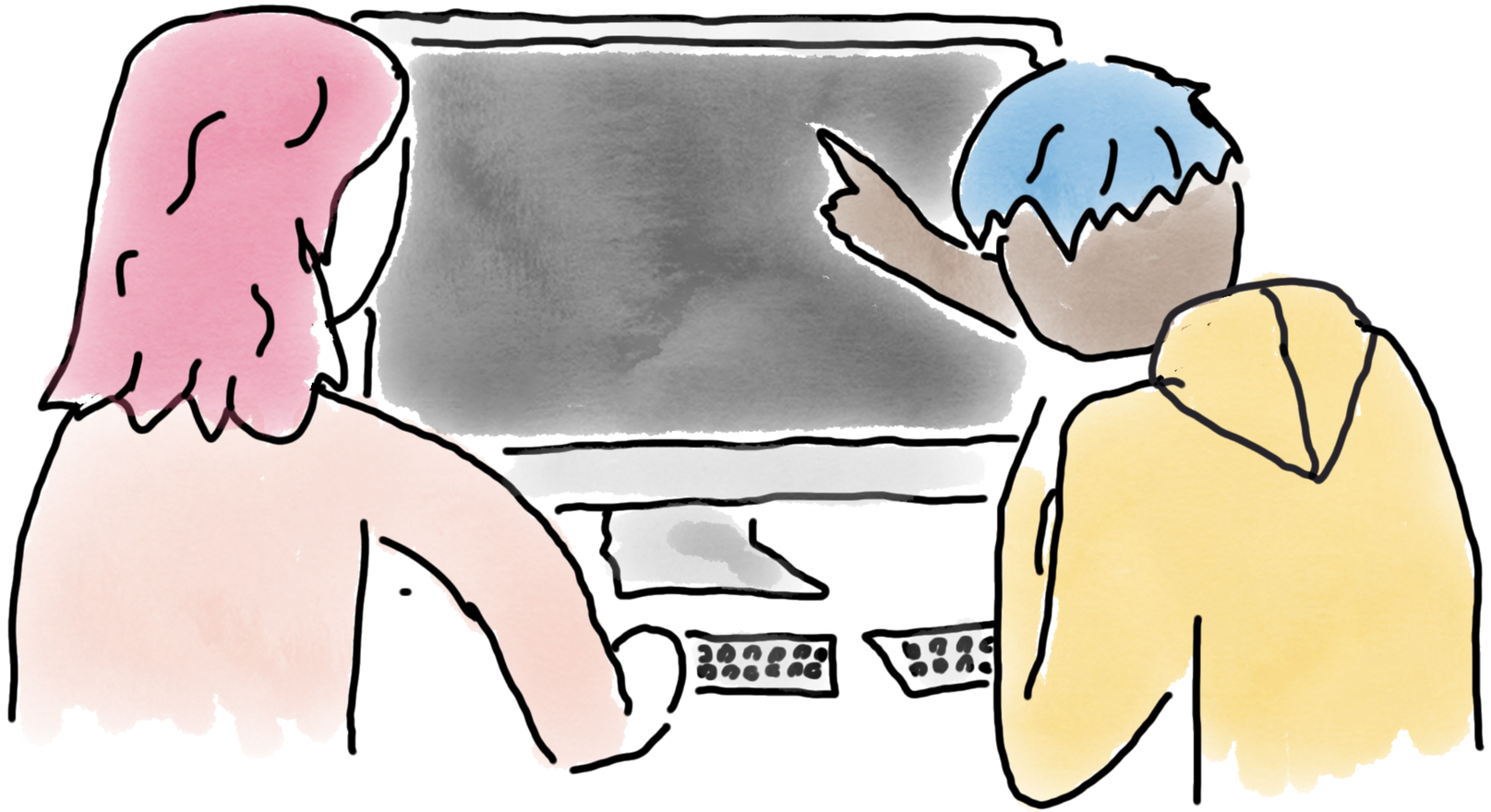
GitOps

infrastructure as code



ok but preventing problems?

pair programming



test-driven development (TDD)

if you care about it, automate it

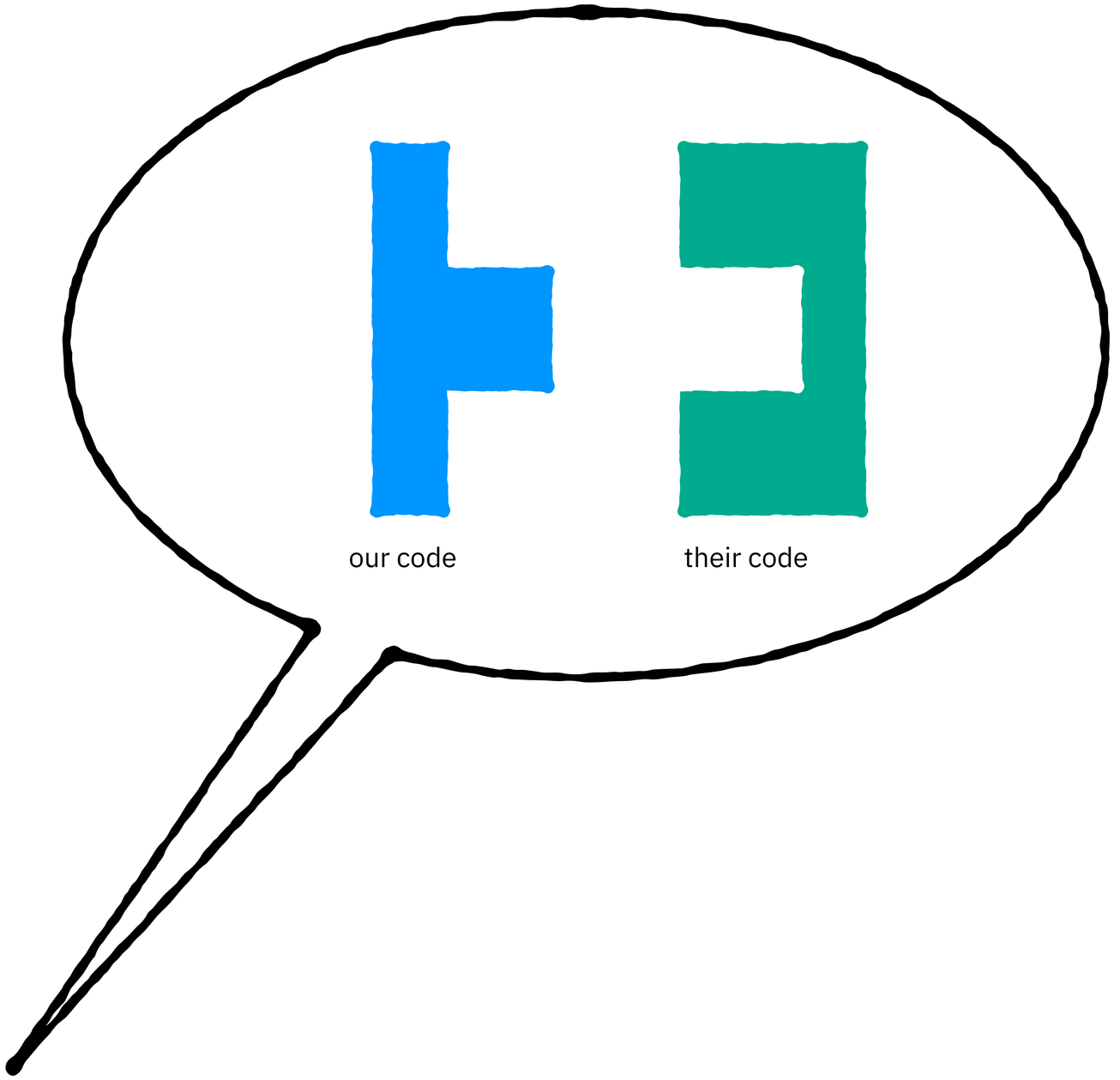
integrate early and often

integrate early and often

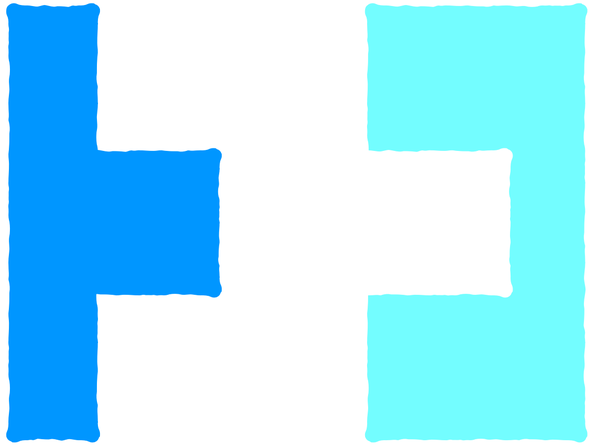
many times a day

contract test your interactions

the problem with mocks



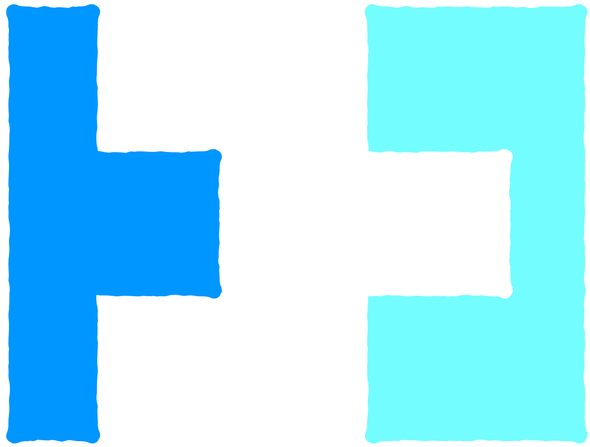
the problem with mocks



our code

our mock

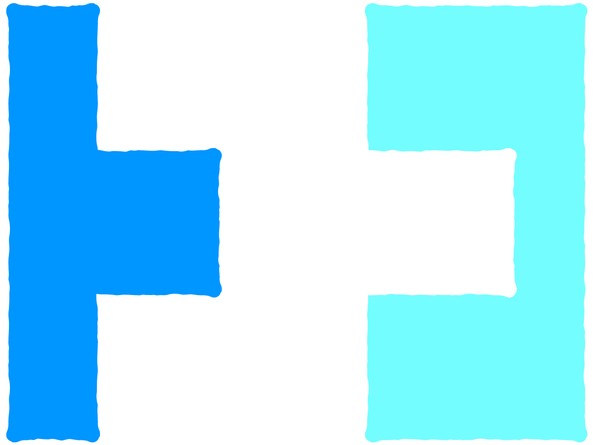
the problem with mocks



our code | our mock

tests ✔

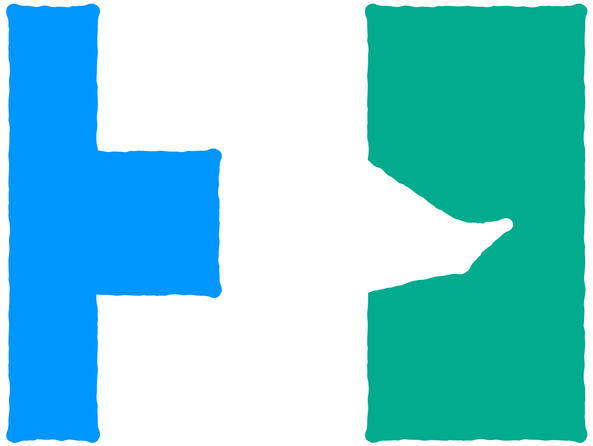
the problem with mocks



our code

our mock

tests ✔



our code

their actual code

the problem with mocks

our code

our mock

tests ✔

reality ✘

our code

their actual code

the problem with mocks

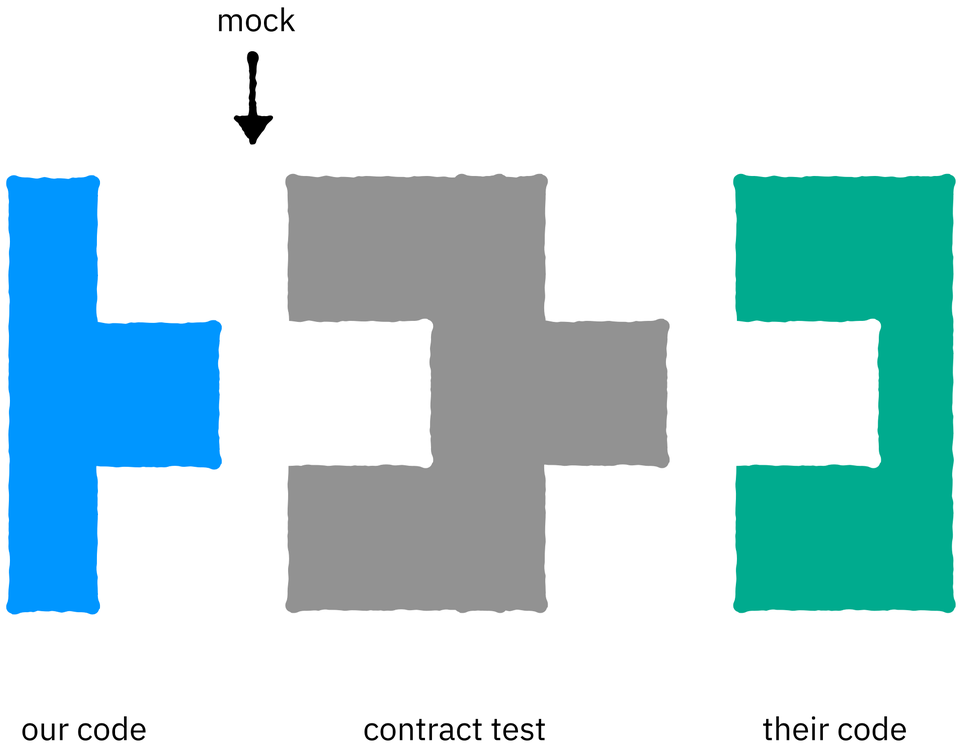
our code

their code

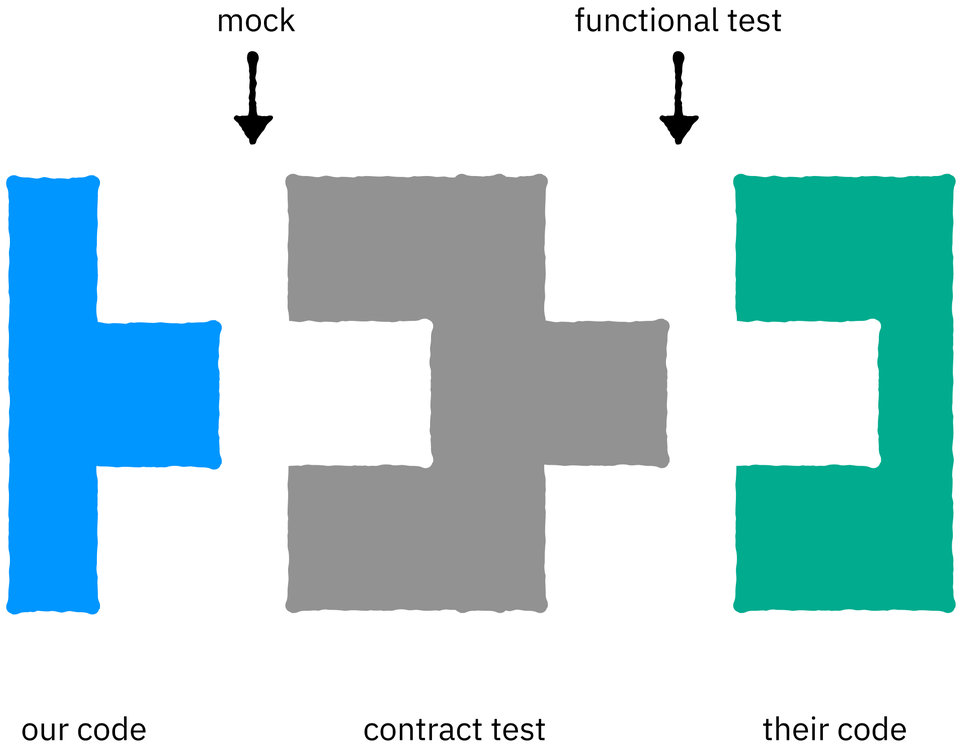
the problem with mocks

our code contract test their code

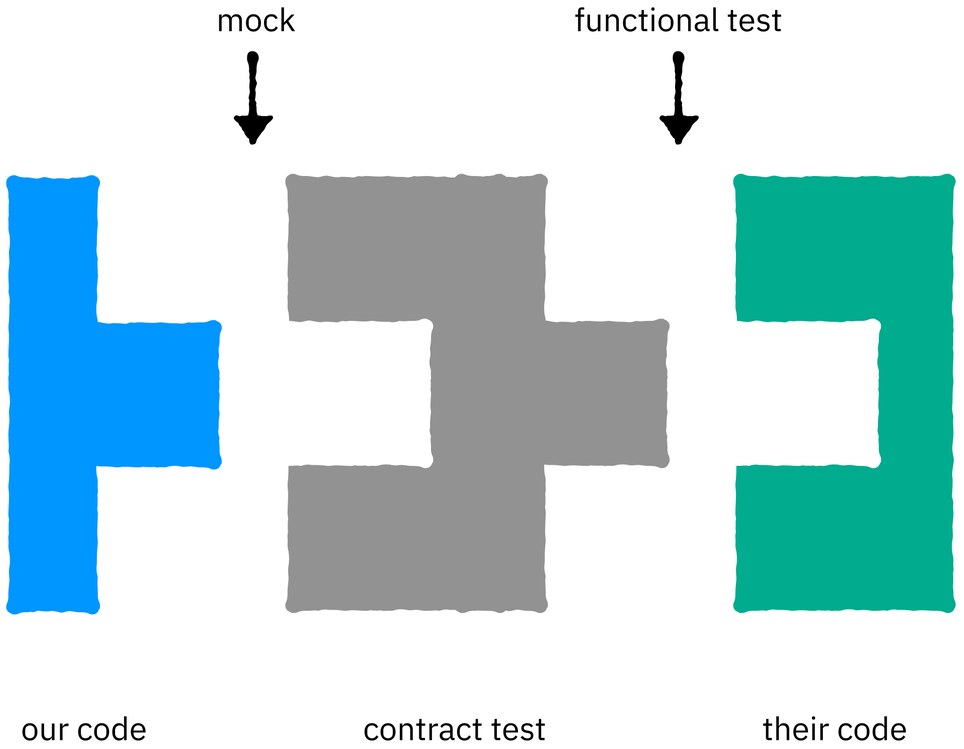
the problem with mocks



the problem with mocks

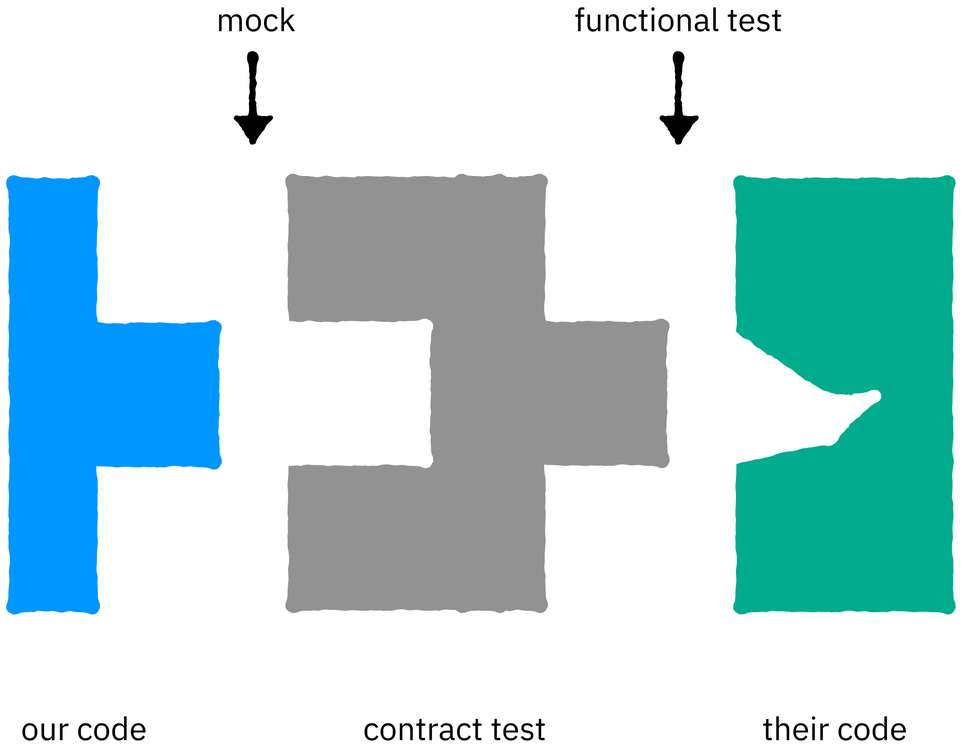


the problem with mocks

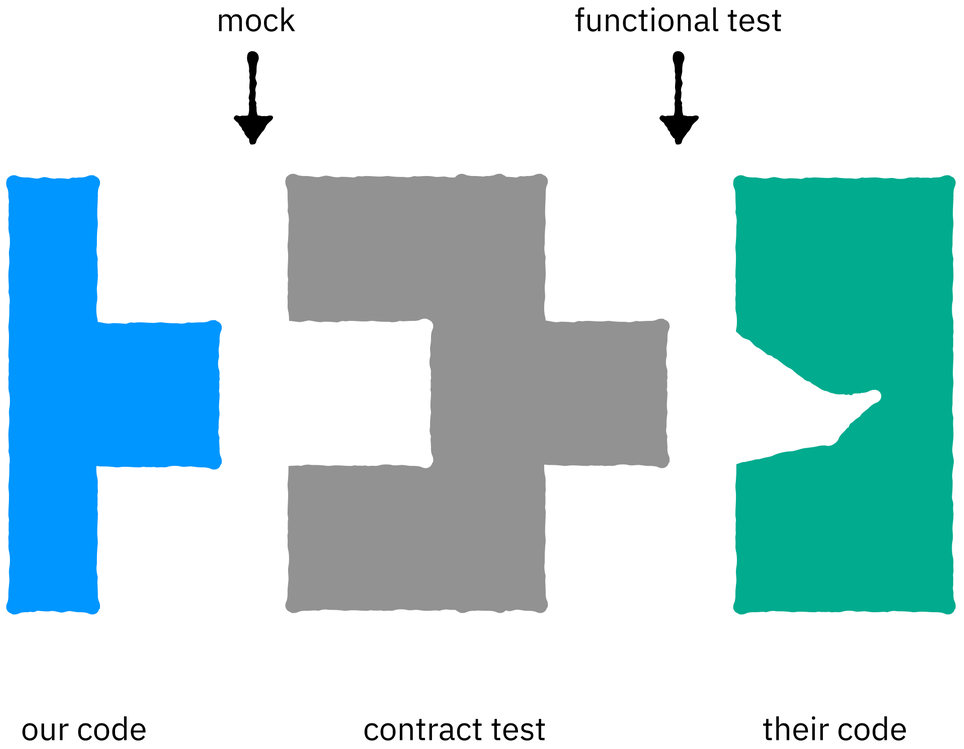


our tests ✔
their tests ✔
reality ✔

the problem with mocks

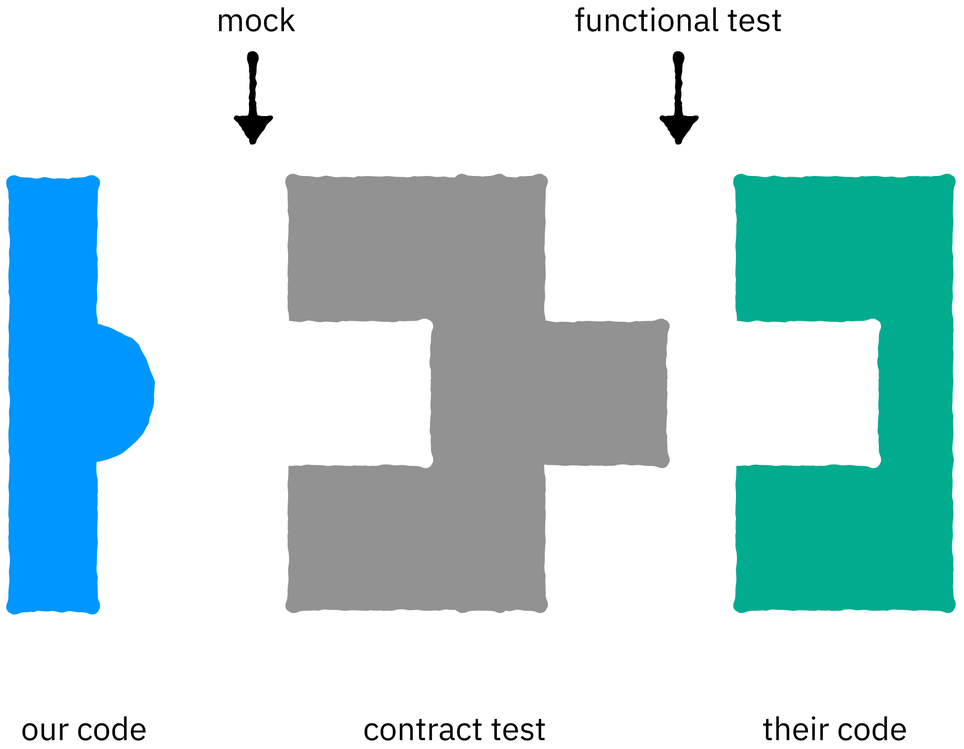


the problem with mocks

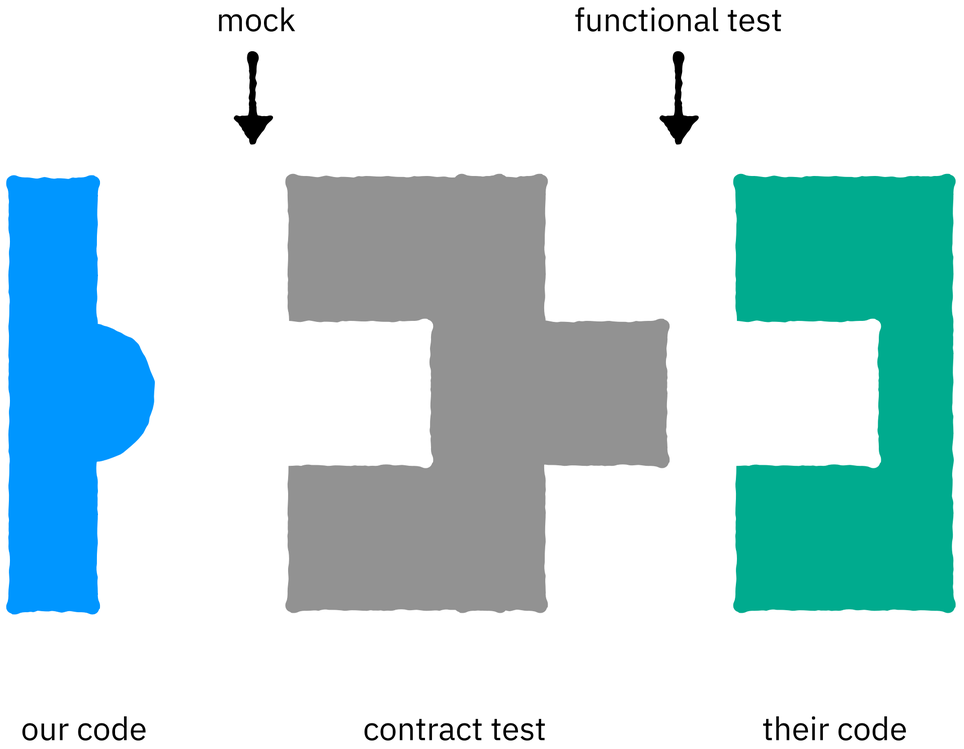


our tests ✔
their tests ✘
reality ✘

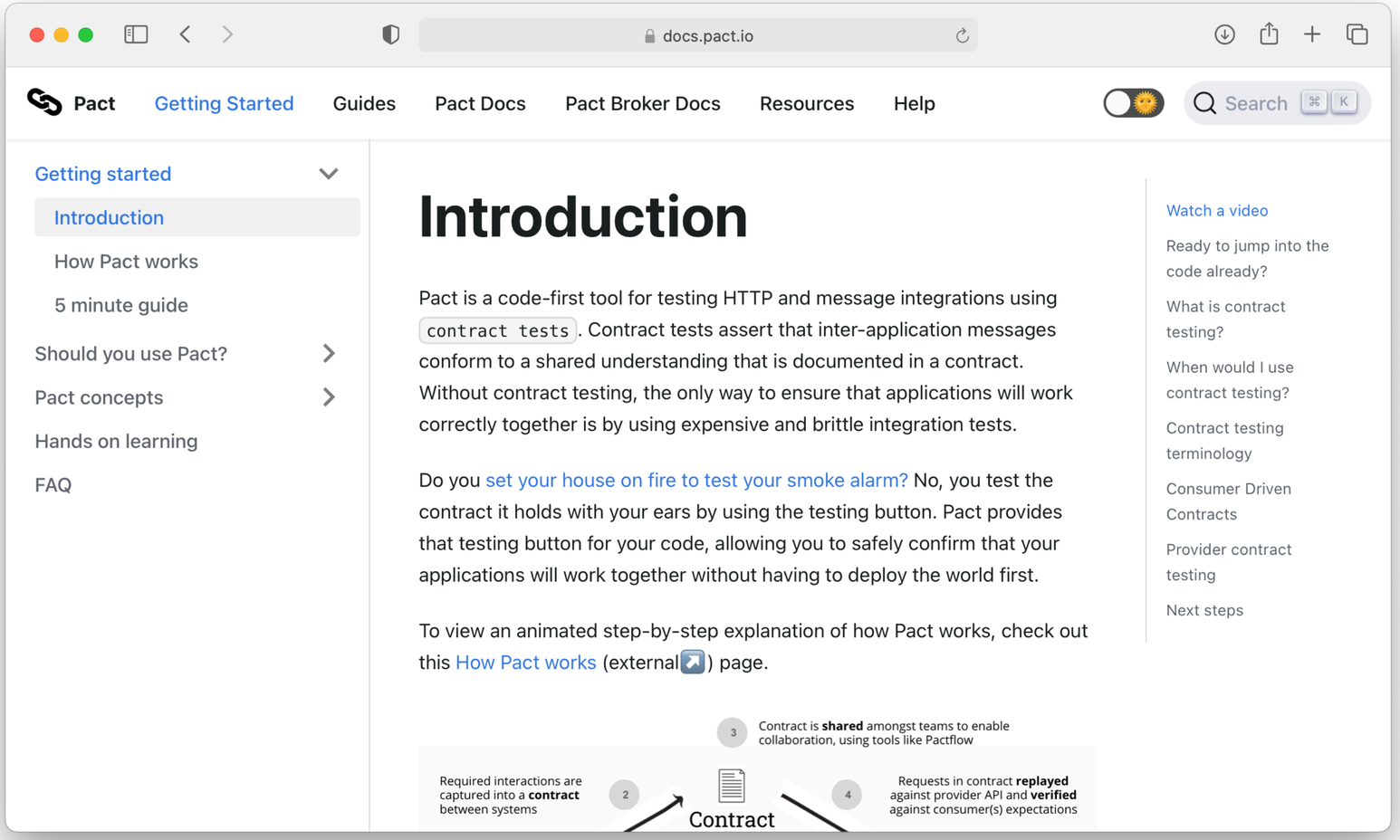
the problem with mocks



the problem with mocks



our tests ✘
their tests ✔
reality ✘


docs.pact.io
Pact
Getting Started
Guides
Pact Docs
Pact Broker Docs
Resources
Help
Search ⌘ K
Getting started
Introduction
How Pact works
5 minute guide
Should you use Pact?
Pact concepts
Hands on learning
FAQ
Introduction
Pact is a code-first tool for testing HTTP and message integrations using contract tests. Contract tests assert that inter-application messages conform to a shared understanding that is documented in a contract. Without contract testing, the only way to ensure that applications will work correctly together is by using expensive and brittle integration tests.
Do you set your house on fire to test your smoke alarm? No, you test the contract it holds with your ears by using the testing button. Pact provides that testing button for your code, allowing you to safely confirm that your applications will work together without having to deploy the world first.
To view an animated step-by-step explanation of how Pact works, check out this How Pact works (external) page.
Watch a video
Ready to jump into the code already?
What is contract testing?
When would I use contract testing?
Contract testing terminology
Consumer Driven Contracts
Provider contract testing
Next steps
3
Contract is shared amongst teams to enable collaboration, using tools like Pactflow
Required interactions are captured into a contract between systems
2
Contract
4
Requests in contract replayed against provider API and verified against consumer(s) expectations

demo

2014

Ant

Java 7
OSGi
WebSphere

locally deployed

2021

Tekton

Kubernetes
OpenShift
Node.js
React.js

on public cloud

my stack

your ability to learn is a key professional asset

teach people the stuff you're learning

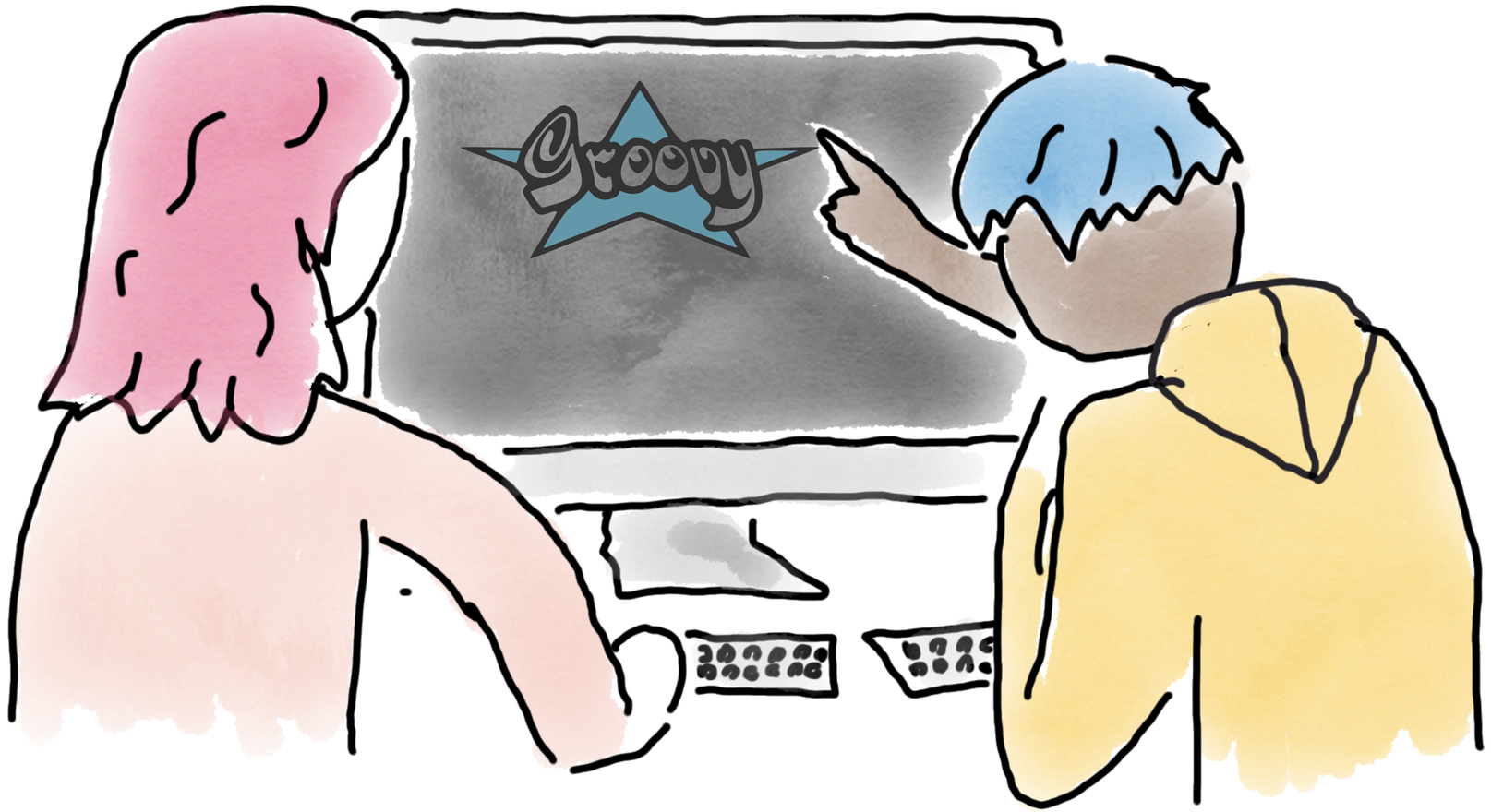

Groovy

2008: a developer had a lot of fun with Groovy.

2008: a developer had a lot of fun with Groovy.

2009: he left the company; the others who had to maintain his code had less fun.

pair programming



the value of discomfort

TDD (test driven development)

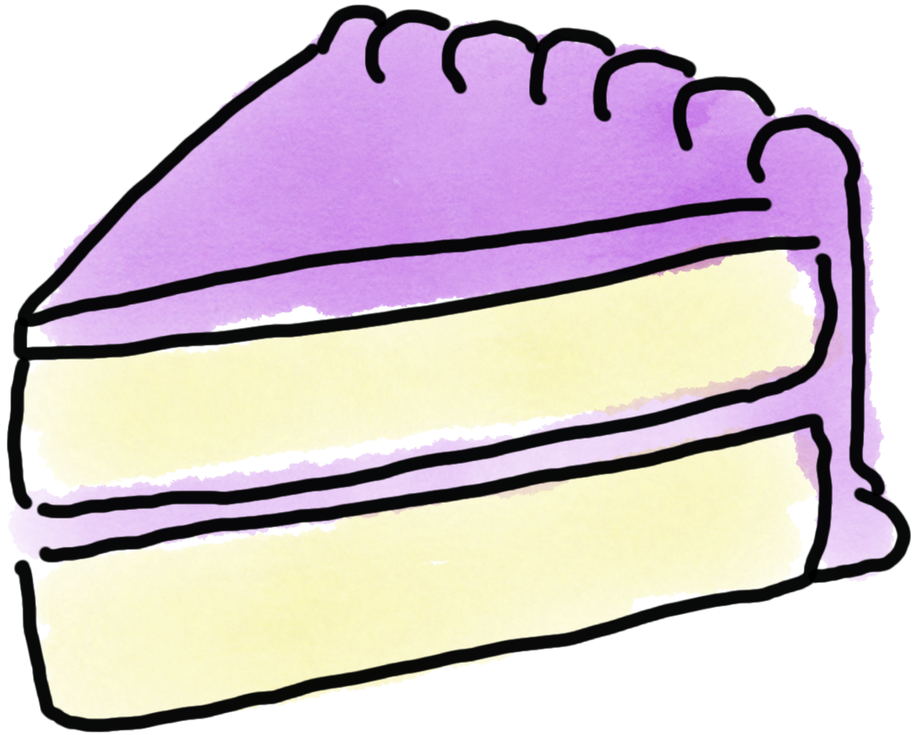
TDD (test driven development)

BDD (behaviour driven development)

TDD (test driven development)

BDD (behaviour driven development)

CDD (cake driven development)

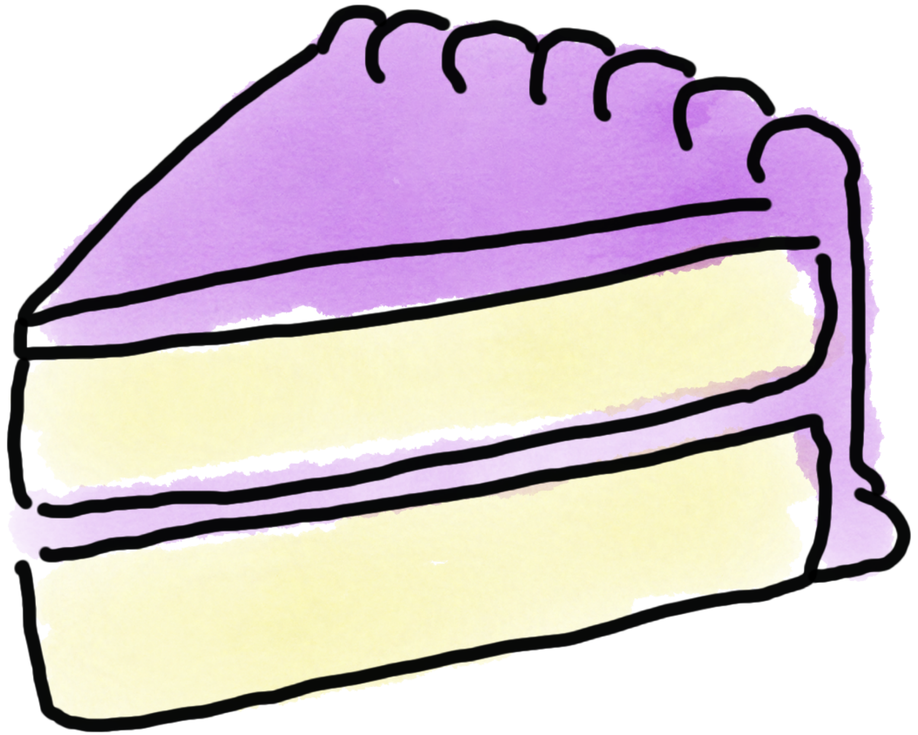


TDD (test driven development)

BDD (behaviour driven development)

CDD (cake driven development)

PDD (pain driven development)



harness discomfort to drive innovation

learning comes from failure

success comes from learning

thank you!

(and have fun at the rest of the event)

Holly Cummins
@holly_cummins