

Life outside the Chocolate Factory

Why employee productivity is a reliability issue

@msuriar

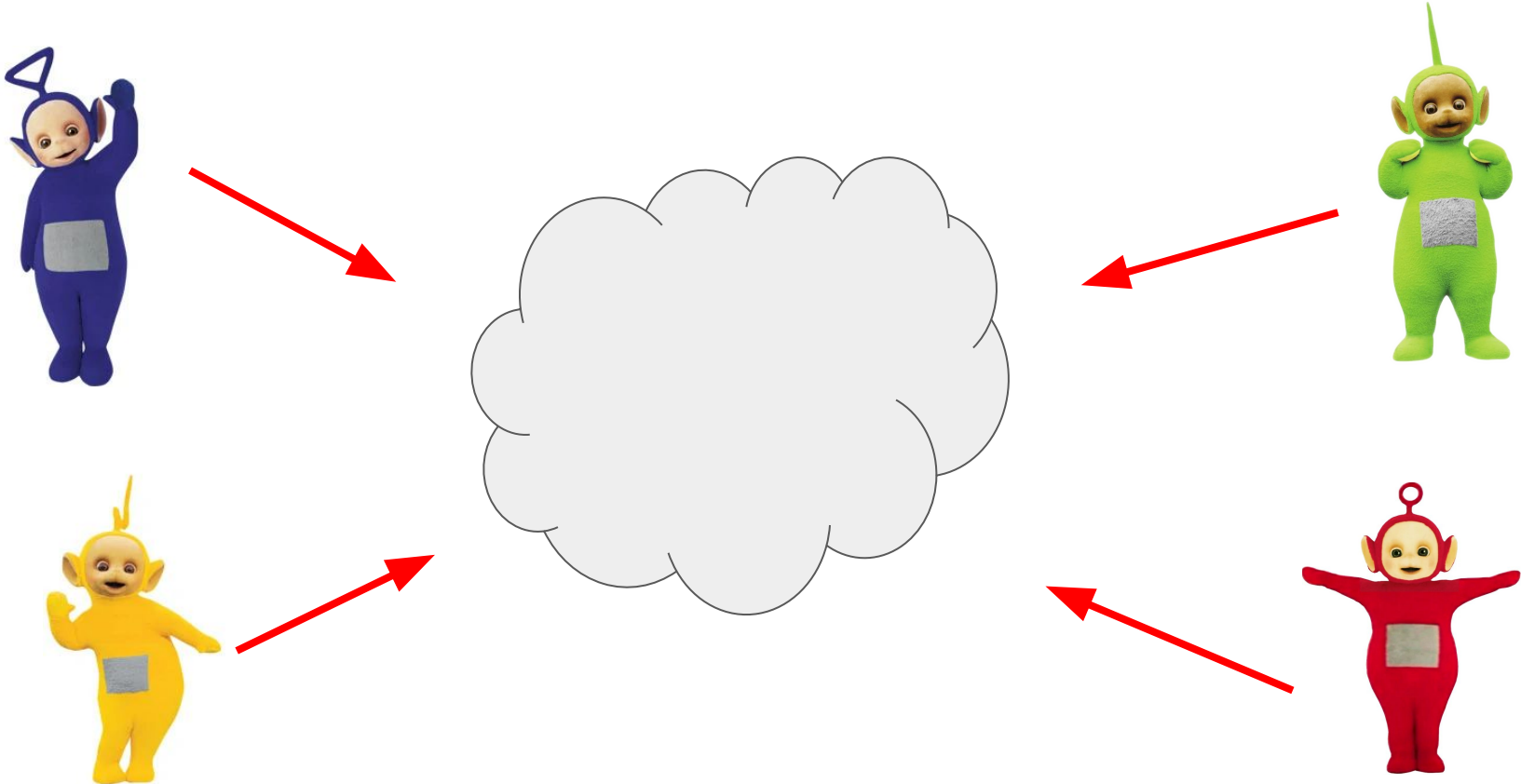
@EmilStolarsky



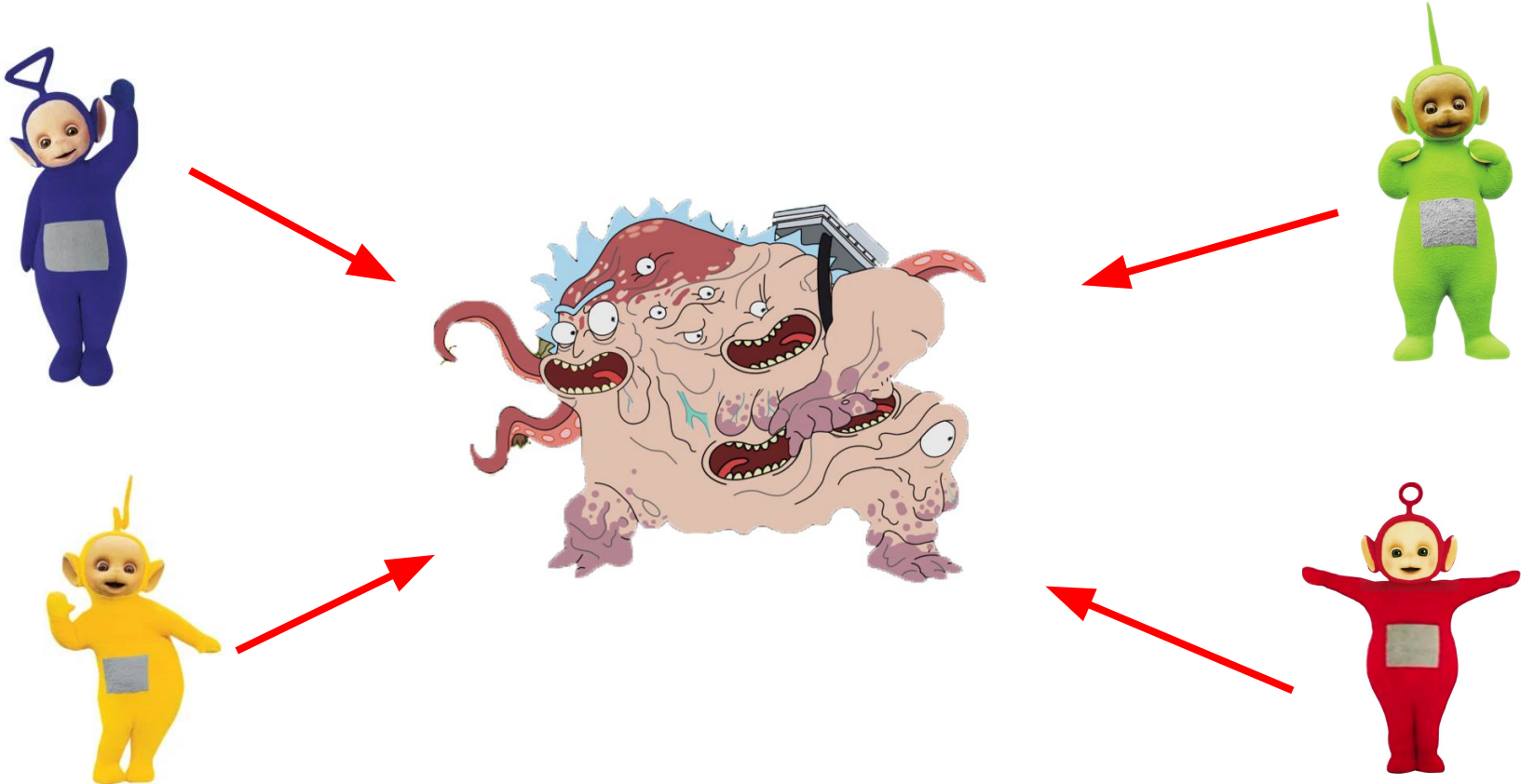
Disclaimer



Disclaimer



Disclaimer





Onboarding



1st day





Getting Started

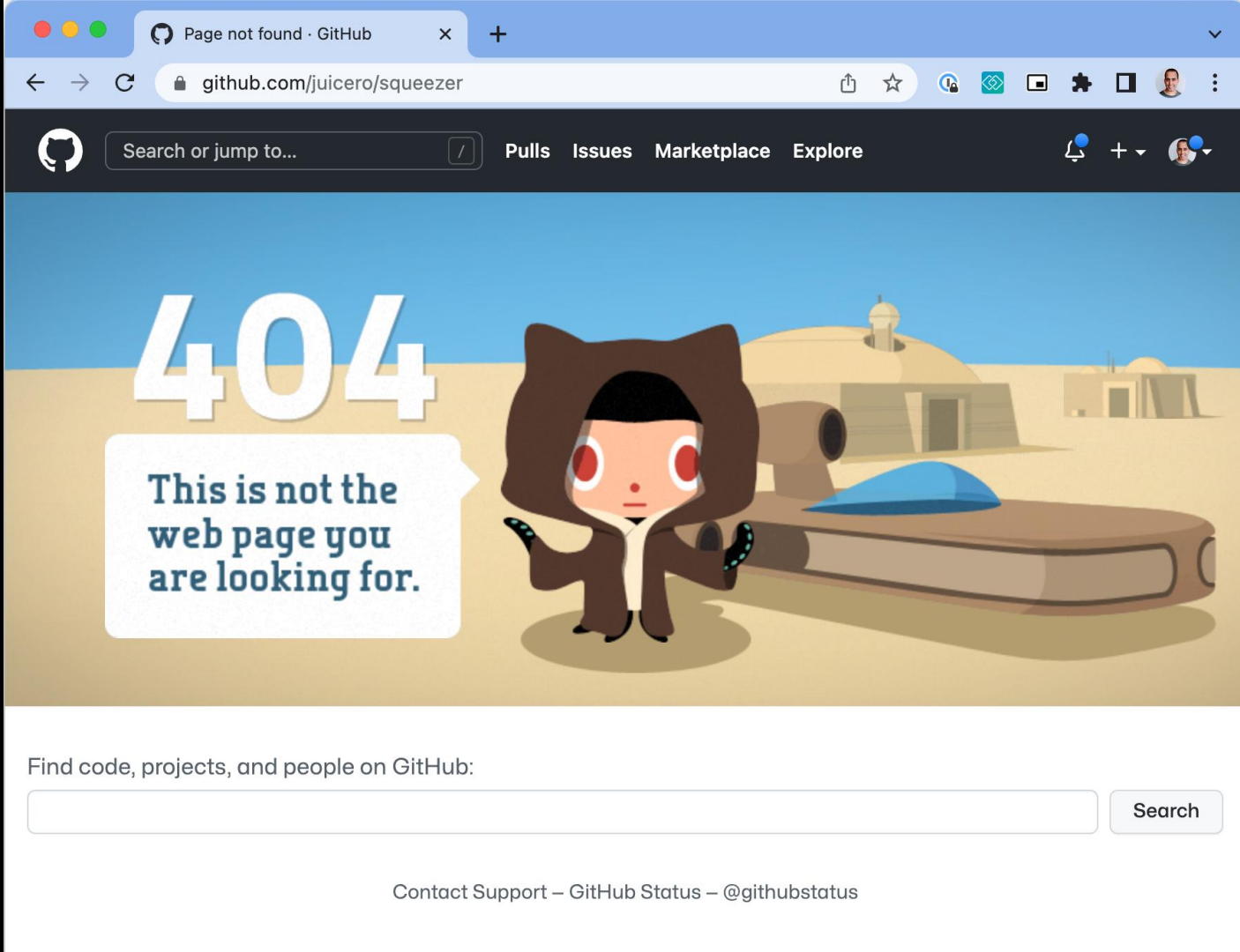
Welcome to Juicero!

This microservices monorepo runs on Kubernetes, OpenShift, and OpenStack - just like all the fruits in a salad! To get started, go ahead and clone our repo:

```
| git clone git@github.com:juicero/squeezer.git
```

Table of Contents

- [Getting Started](#)
- [Our Mission](#)
- [Testing](#)





Getting Started

Welcome to Juicero!

This microservices monorepo runs on Kubernetes, OpenShift, and OpenStack - just like all the fruits in a salad! To get started, go ahead and clone our repo:

```
| git clone git@github.com:juicero/squeezer.git
```

Table of Contents

- [Getting Started](#)
- [Our Mission](#)
- [Testing](#)

Getting Started

Welcome to Juicero!

This microservices monorepo runs on Kubernetes, OpenShift, and OpenStack - just like all the fruits in a salad! To get started, go ahead and clone our repo:

```
| git clone git@github.com:juicero/squeezer.git
```

Table of Contents

- [Getting Started](#)
- [Our Mission](#)
- [Testing](#)

- [Our Mission](#)

- Testing

Getting Started

Table of Contents

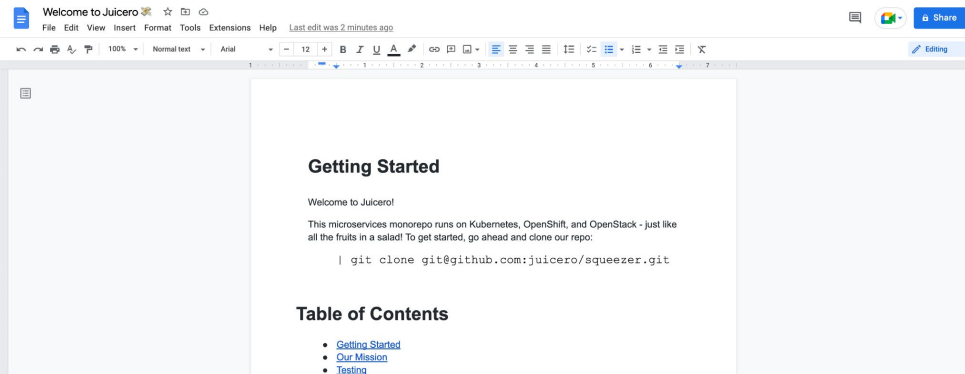
Costs

World
Images

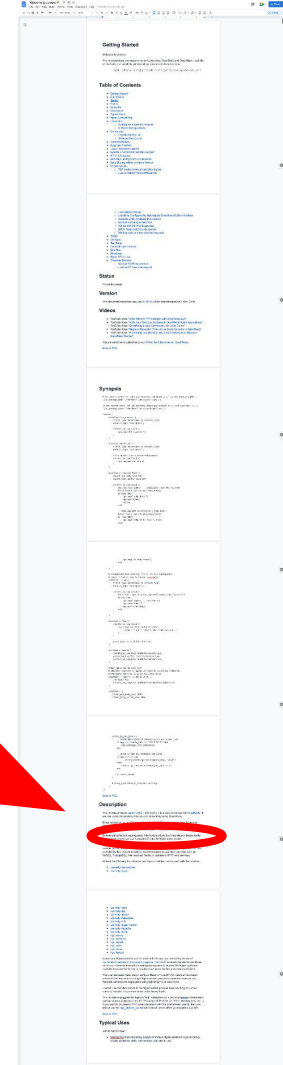
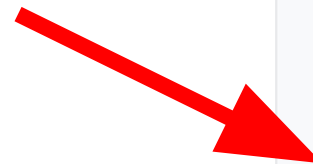
Syn

Desa

Type:

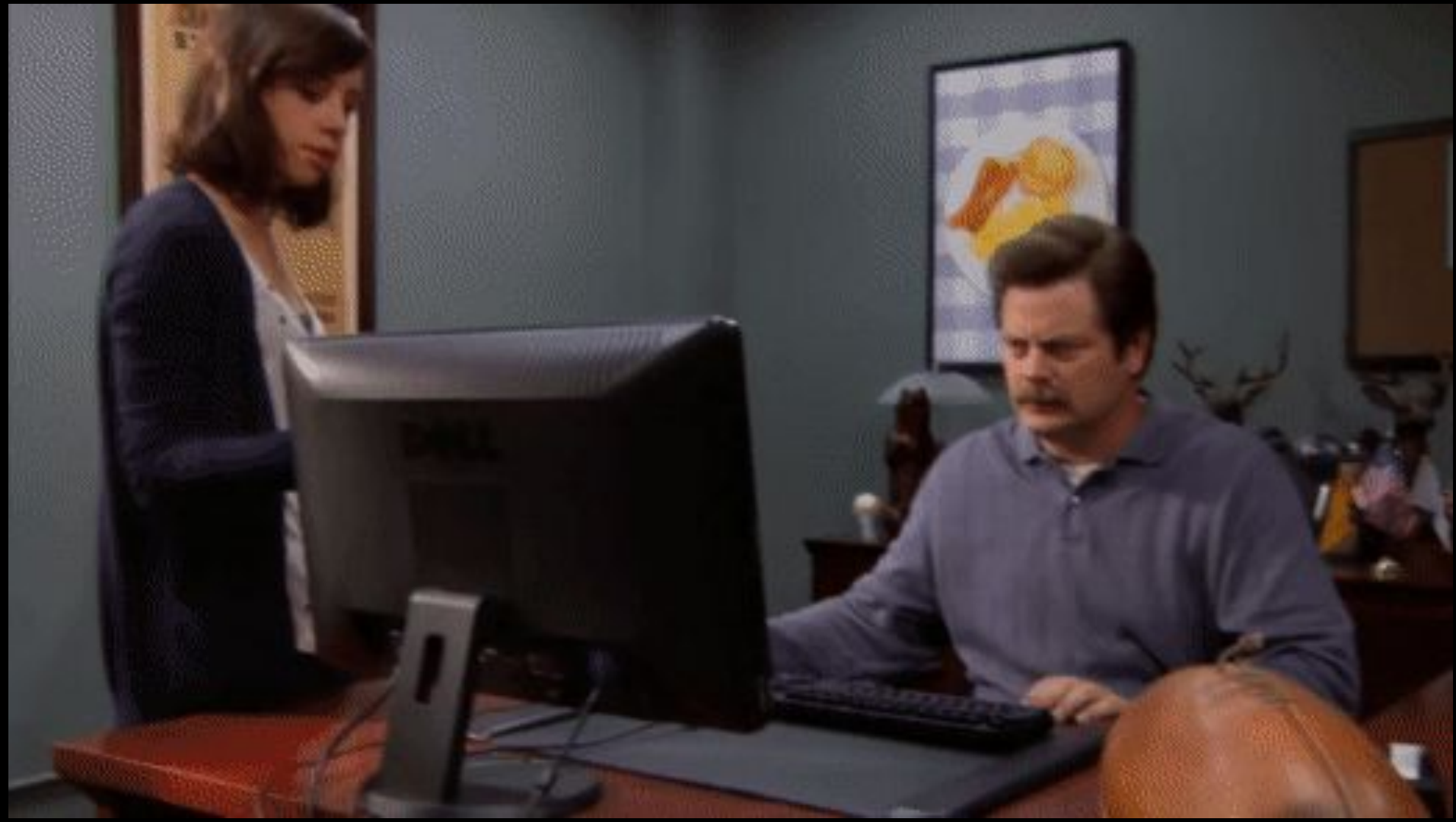


“Make sure to ask IT to be added to the Github Organization before cloning 😊”







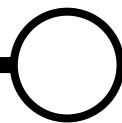
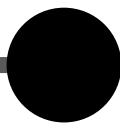






Onboarding

Deploy

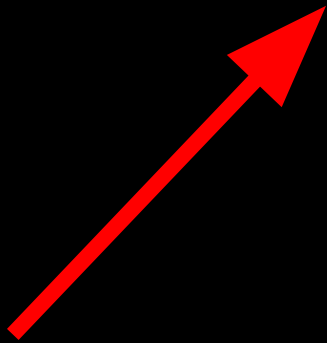


1st day

1st week

INFO: 2022/11/10 12:01:06 main.go:26: Starting the application...

INFO: 2022/11/10 12:01:07 main.go:27: Enabling juice DRM




```
package main

import (
    "log"
    "os"
)

var (
    InfoLogger    *log.Logger
)

func init() {
    file, err := os.OpenFile("logs.txt", os.O_APPEND|os.O_CREATE|os.O_WRONLY, 0666)
    if err != nil {
        log.Fatal(err)
    }

    InfoLogger = log.New(file, "INFO: ", log.Ldate|log.Ltime|log.Lshortfile)
}

func main() {
    InfoLogger.Println("Starting the application...")
    InfoLogger.Println("Enablng juice DRM")

    // TODO: figure out viable business model
}
```



```
package main

import (
    "log"
    "os"
)

var (
    InfoLogger    *log.Logger
)

func init() {
    file, err := os.OpenFile("logs.txt", os.O_APPEND|os.O_CREATE|os.O_WRONLY, 0666)
    if err != nil {
        log.Fatal(err)
    }

    InfoLogger = log.New(file, "INFO: ", log.Ldate|log.Ltime|log.Lshortfile)
}



func main() {
    InfoLogger.Println("Starting the application...")
    InfoLogger.Println("Enabling juice DRM")

    // TODO: figure out viable business model
}
```



All checks have passed

2 successful checks

- | | | | |
|---|---|--|-------------------------|
| ✓ |  | ci/circleci: setup — Your tests passed on CircleCI! | Details |
| ✓ |  | ci/circleci: build — Your tests passed on CircleCI! | Details |

[Pull requests](#)[Issues](#)[Marketplace](#)[Explore](#)

[juicero](#) / [squeezer](#) Private

[Code](#)[Issues](#)[Pull requests](#) 1[Actions](#)[Projects](#)[Wiki](#)[Security](#)[Insights](#)[Settings](#)

Fix spelling error #42

[Open](#)

msuriar wants to merge 1 commit into [main](#) from [msuriar/fix-log-line](#)

[Pull requests](#)[Issues](#)[Marketplace](#)[Explore](#)

[juicero](#) / [squeezer](#) Private

[Code](#)[Issues](#)[Pull requests](#) 1[Actions](#)[Projects](#)[Wiki](#)[Security](#)[Insights](#)[Settings](#)

Fix spelling error #42

[Open](#)

msuriar wants to merge 1 commit into [main](#) from [msuriar/fix-log-line](#)

All checks have passed

1 failed and 2 successful checks



ci/circleci: setup — Your tests passed on CircleCI!

[Details](#)

ci/circleci: build — Your tests passed on CircleCI!

[Details](#)

ci/circleci: test — Your tests failed on CircleCI!

[Details](#)


```

endif

ifeq (${WODBY_GROUP_ID},)
    WODBY_GROUP_ID := 1000
endif

ifeq (${BASE_IMAGE_STABILITY_TAG},)
    BASE_IMAGE_TAG := $(ALPINE_VER)
else
    BASE_IMAGE_TAG := $(ALPINE_VER)-$(BASE_IMAGE_STABILITY_TAG)
endif

REGISTRY ?= docker.io
REPO = ${REGISTRY}/wodby/juicero
NAME = juicero-$(JUICERO_MINOR_VER)

ifneq (${STABILITY_TAG},)
    ifneq (${TAG},latest)
        override TAG := ${TAG}-$(STABILITY_TAG)
    endif
endif

.PHONY: build buildx-build buildx-push buildx-build-amd64 test push shell run start stop logs
clean release

default: build

# Make sure to download personal aws_config.yml from AWS on your first commit
build:
    docker build -t ${REPO}:${TAG} \
        -v "${PWD}"/aws_config.yml:/app:z \
        --build-arg BASE_IMAGE_TAG=$(BASE_IMAGE_TAG) \
        --build-arg JUICERO_VER=$(JUICERO_VER) \
        --build-arg WODBY_GROUP_ID=$(WODBY_GROUP_ID) \
        --build-arg WODBY_USER_ID=$(WODBY_USER_ID) ./

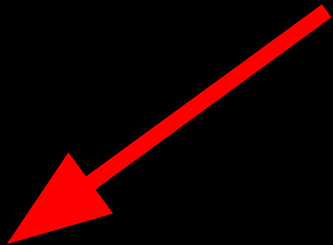
# --load doesn't work with multiple platforms https://github.com/docker/buildx/issues/59
# we need to save cache to run tests first.
buildx-build-amd64:
    docker buildx build --platform linux/amd64 \
        --build-arg BASE_IMAGE_TAG=$(BASE_IMAGE_TAG) \
        --build-arg JUICERO_VER=$(JUICERO_VER) \
        --build-arg WODBY_GROUP_ID=$(WODBY_GROUP_ID) \
        --build-arg WODBY_USER_ID=$(WODBY_USER_ID) \
        --load \
        -t ${REPO}:${TAG} ./

buildx-build:
    docker buildx build --platform ${PLATFORM} \
        --build-arg BASE_IMAGE_TAG=$(BASE_IMAGE_TAG) \
        --build-arg JUICERO_VER=$(JUICERO_VER) \
        --build-arg WODBY_GROUP_ID=$(WODBY_GROUP_ID) \
        --build-arg WODBY_USER_ID=$(WODBY_USER_ID) \
        -t ${REPO}:${TAG} ./

```

Makefile

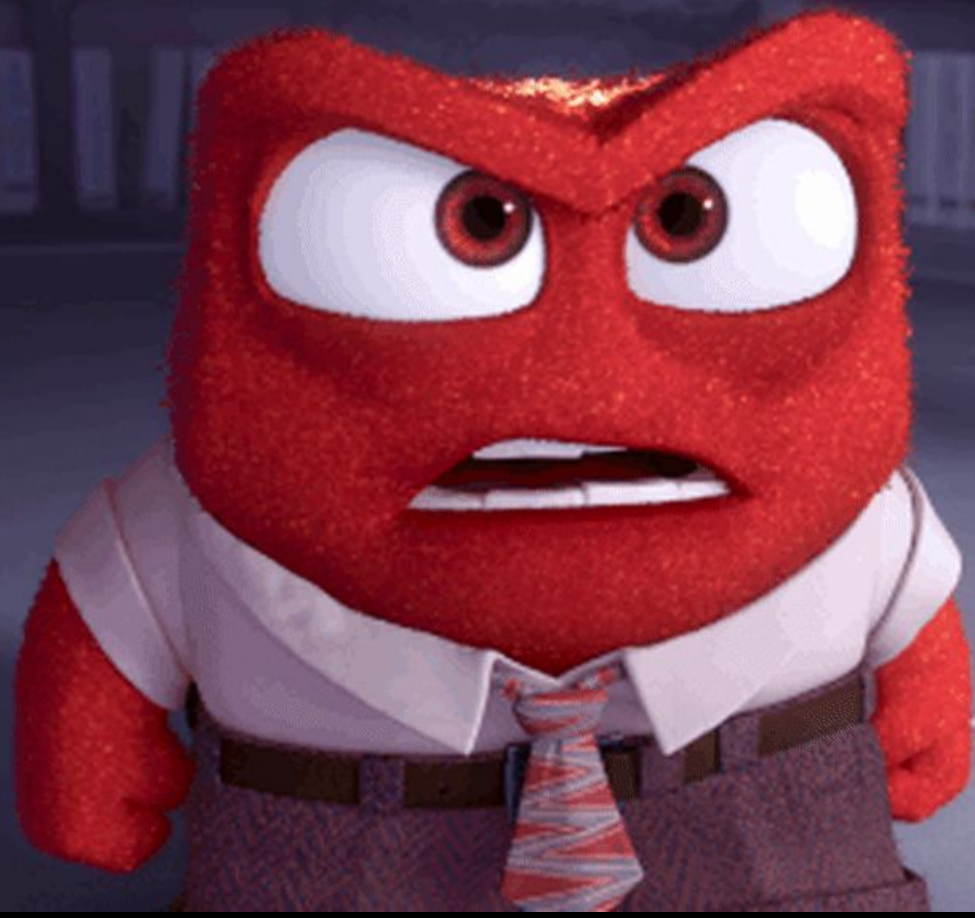
WHAT



```
# Make sure to download personal aws_config.yml from AWS on your first
```

```
docker build -t ${REPO}:${TAG} \  
  -v "$(pwd)"/aws_config.yml:/app:z \  
  --build-arg BASE_IMAGE_TAG=${BASE_IMAGE_TAG} \  
  --build-arg JUICERO_VER=${JUICERO_VER} \  
  --build-arg WODBY_GROUP_ID=${WODBY_GROUP_ID} \  
  --build-arg WODBY_USER_ID=${WODBY_USER_ID} ./. 
```

Makefile





[Pull requests](#)[Issues](#)[Marketplace](#)[Explore](#)

[juicero](#) / [squeezer](#) Private

[Code](#)[Issues](#)[Pull requests](#) 1[Actions](#)[Projects](#)[Wiki](#)[Security](#)[Insights](#)[Settings](#)

Fix spelling error #42

[Open](#)

msuriar wants to merge 1 commit into [main](#) from [msuriar/fix-log-line](#)

All checks have passed

1 failed and 2 successful checks



ci/circleci: setup — Your tests passed on CircleCI!

[Details](#)

ci/circleci: build — Your tests passed on CircleCI!

[Details](#)

ci/circleci: test — Your tests failed on CircleCI!

[Details](#)

[Pull requests](#)[Issues](#)[Marketplace](#)[Explore](#)

[juicero](#) / [squeezer](#) Private

[Code](#)[Issues](#)[Pull requests](#) 1[Actions](#)[Projects](#)[Wiki](#)[Security](#)[Insights](#)[Settings](#)

JIRA-00000 Fix spelling error #2

[Open](#)

msuriar wants to merge 1 commit into [main](#) from [msuriar/fix-log-line](#)

[Pull requests](#)[Issues](#)[Marketplace](#)[Explore](#)

[juicero](#) / [squeezer](#) Private

[Code](#)[Issues](#)[Pull requests](#) 1[Actions](#)[Projects](#)[Wiki](#)[Security](#)[Insights](#)[Settings](#)

JIRA-00000 Fix spelling error #2

[Open](#)

msuriar wants to merge 1 commit into [main](#) from [msuriar/fix-log-line](#)

All checks have passed

3 successful checks



ci/circleci: setup — Your tests passed on CircleCI!

[Details](#)

ci/circleci: build — Your tests passed on CircleCI!

[Details](#)

ci/circleci: test — Your tests passed on CircleCI!

[Details](#)

[Pull requests](#)[Issues](#)[Marketplace](#)[Explore](#)

[juicero](#) / [squeezer](#) Private

[Code](#)[Issues](#)[Pull requests](#) 1[Actions](#)[Projects](#)[Wiki](#)[Security](#)[Insights](#)[Settings](#)

JIRA-00000 Fix spelling error #2

[Open](#)

msuriar wants to merge 1 commit into [main](#) from [msuriar/fix-log-line](#)



Merging is blocked

The base branch restricts merging to authorized users. [Learn more about protected branches.](#)

[Merge pull request](#) You're not [authorized](#) to merge this pull request.



Discussion

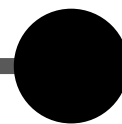
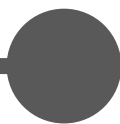
- Permissions-Permissions-Permissions
 - How many times will someone need to ask to be added to a group or granted certain permissions?
- 2nd Order Effects
 - Multiple systems splinter any ability to improve systems as a whole
 - Imagine having to navigate Active Directory & then Okta (and AWS IAM and Github and ... and ..)
 - Or, if you don't have to imagine, curl up into a ball and rock back and forth.



Onboarding

Deploy

Ownership



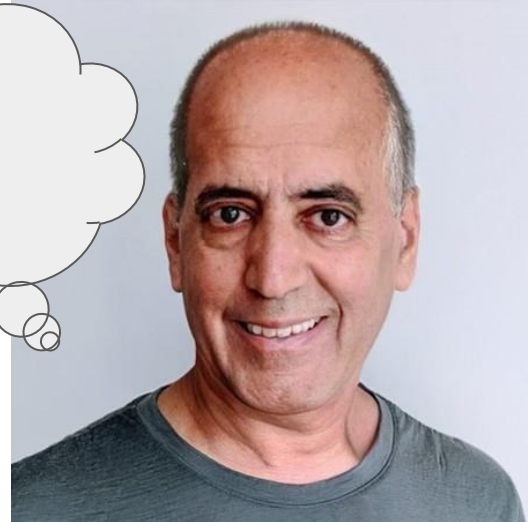
1st day

1st week

1st month



I fixed a
DNS bug



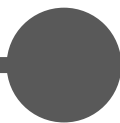
Onboarding

Deploy

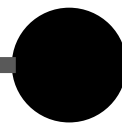
Ownership



1st day



1st week



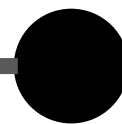
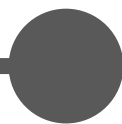
1st month



Onboarding

Deploy

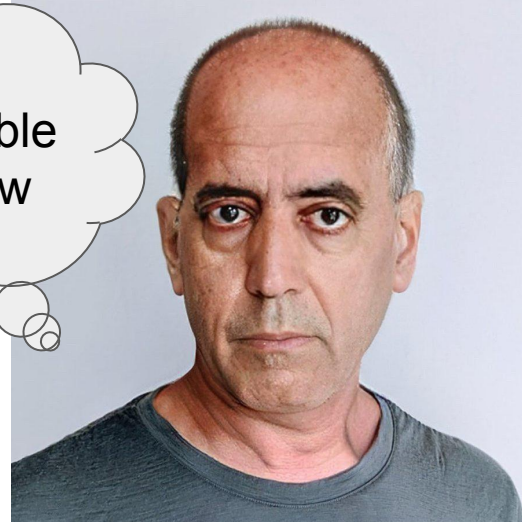
Ownership



1st day

1st week

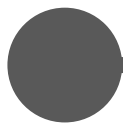
1st month



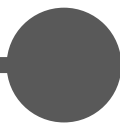
Onboarding

Deploy

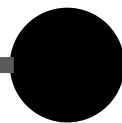
Ownership



1st day



1st week



1st month



Juicero:

(noun) /joos/ār/rō



Juicero:



(noun) /joos/ār/rō



Projects / iOS App

Components

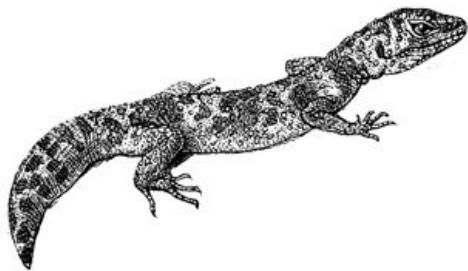
Create component

Component ▾	Description ▾	Component lead ▾	Default assignee	Issues ▾	
Error logging	Mandatory for all	 Maxwell Mei	Component lead	1 issue	...
API	Collection of all APIs	 Kelly Drozd	Project lead	4 issues	...



YOU NEED TO
LEAVE NOW

Sure, that'll cure burnout



Leaving FAANG for a Startup

The Definitive Guide

Conclusion

O RLY?

@msuriar, @EmilStolarsky

Who are we?

Emil

- Never worked at Google
- Work at Wave Mobile Money (prev. Shopify, DigitalOcean, etc.)
- Friends with Xooglers

Murali

- NetOps, SRE at Google (11 years)
- At Snowflake since February
- ??? This talk started as therapy for my culture shock. ???

Ways forward - Onboarding

- For things that everyone needs... do them ahead of start date, or at least make them automated/self-service.
- Have a well lit path for the tasks you expect everyone to do
 - have a working dev environment on every cleanly imaged computer
- People shouldn't need to tweak or configure **anything**, save enrolling a 2FA device.

Ways forward - CI/CD

- Fail fast, fail close to user, fail loudly, fail clearly
 - don't eat exceptions and reraise
 - make it obvious when something is a permission issue
 - don't make people request permissions one by one, if everyone on a team has the same privileges
- Transparency
 - make it trivial to diff your permissions with a team-mate
 - make it obvious how to expand your permissions (file a ticket here, raise a self service request here, ...)
 - security by obscurity is not what you want here

Ways forward - CI/CD

- CI/CD - just do it
 - It seems like a lot of effort when you're 1-5 engineers
 - But doing it from the start is way cheaper than trying to retrofit it in later on.
 - And doing it early means you'll sand off the rough edges early.
- Have one way of ~~deploying things to production~~ doing any given thing
 - Building software
 - Running tests
 - Deploying to production
 - Getting logs
 - ...
- One AND ONLY ONE way

Ways forward - Ownership

- Delegation, delegation, delegation
- Tools should support hierarchical namespaces, with delegation at arbitrary points.
 - Otherwise you're doomed to comms bottlenecks when trying to (re)organise work.
- Code reflects the organization and the organization reflects JIRA

Ways Forward

- Our companies build products that have a UX we care about (i.e. they're socio-technical systems)
- We often ignore the UX employees experience
- SREs have an outsized impact on that UX
 - Automating development tooling (this could be DevInfra, productivity, whatever, but you end up needing pipelines to implement CI/CD, build, test etc)
 - IAM tooling is usually owned or co-owned by SREs
 - You can delete the boxes JIRA runs on

Thank You

@msuriar

@EmilStolarsky