Functional Programming is 1/0

And other stories from podcasting

@adamgordonbell

Earthly.dev

About Me: Adam Gordon Bell

- Developer Advocate
- Canadian
- Software Engineering Podcaster

@adamgordonbell

Earthly.dev

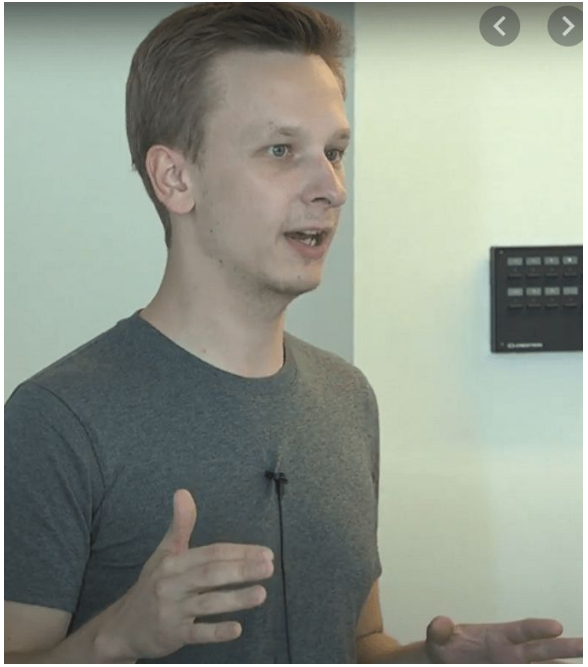
What is going to be covered

- Stories from my podcast
  - Things I learned
  - Things that blew my mind
  - Mistakes are mine, insights are theirs

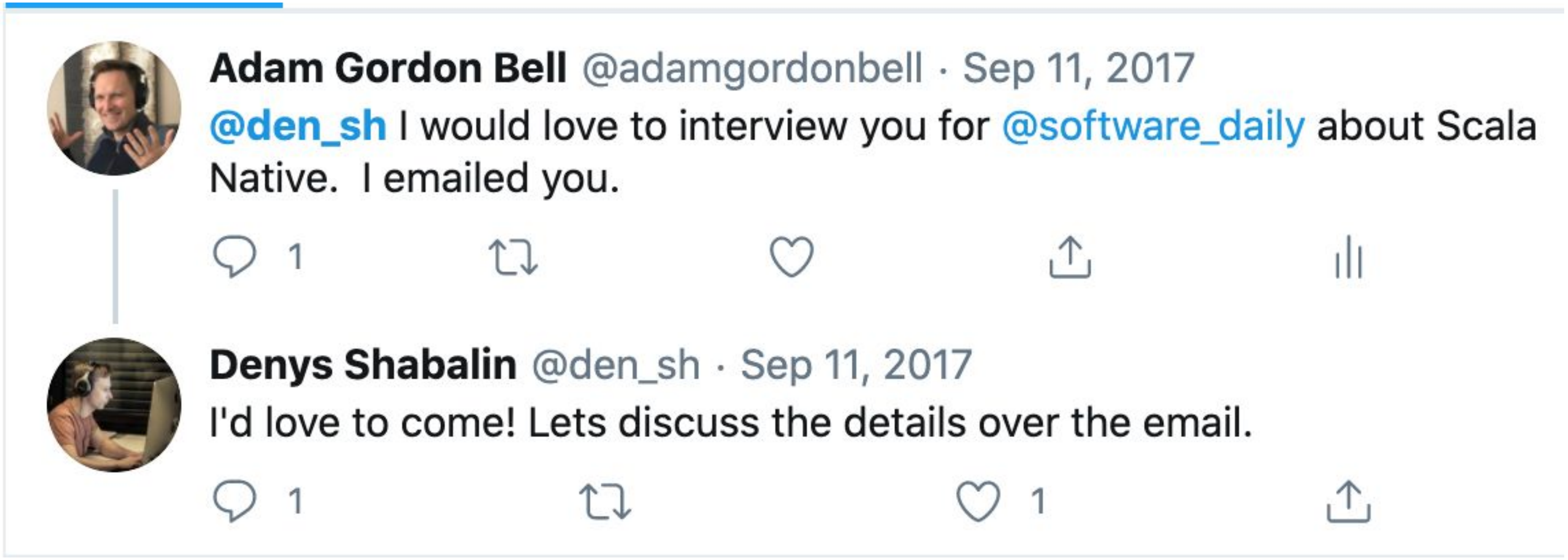
How it started: Jeff & Denys

SOFTWARE ENGINEERING DAILY

The World Through the Lens of Software



How it started: Jeff & Denys



Adam Gordon Bell @adamgordonbell · Sep 11, 2017

@den_sh I would love to interview you for @software_daily about Scala Native. I emailed you.

1

Denys Shabalin @den_sh · Sep 11, 2017

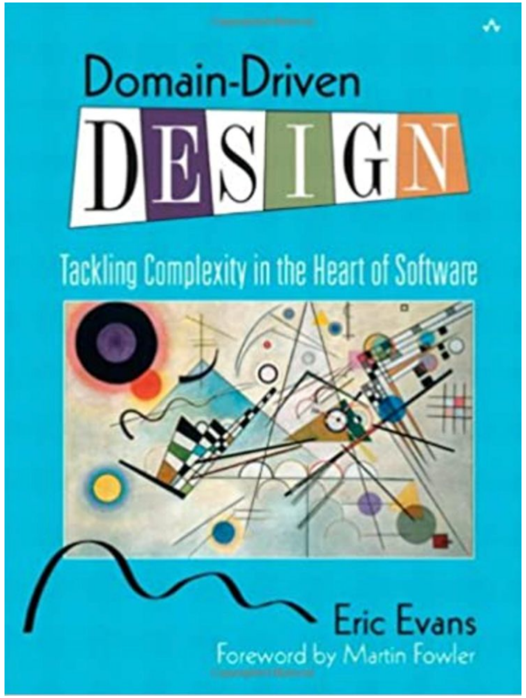
I'd love to come! Lets discuss the details over the email.

1 1

@adamgordonbell

Earthly.dev

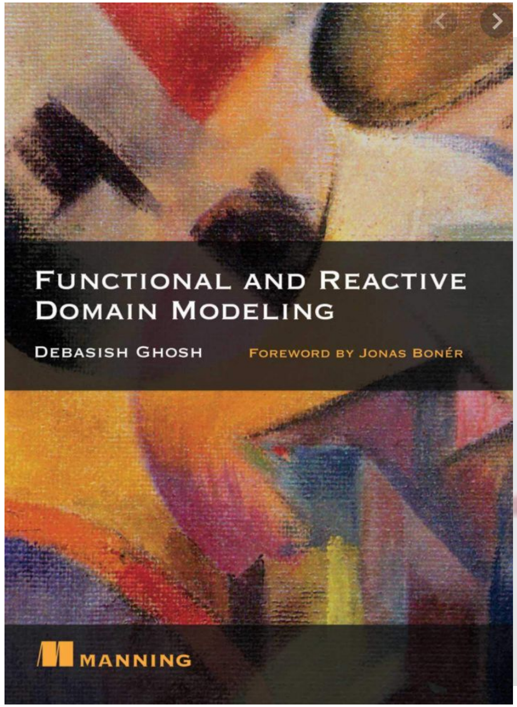
Learning FP


Domain-Driven
DESIGN
Tackling Complexity in the Heart of Software
Eric Evans
Foreword by Martin Fowler

@adamgordonbell

Earthly.dev

Debashish & DDD


FUNCTIONAL AND REACTIVE DOMAIN MODELING

DEBASISH GHOSH

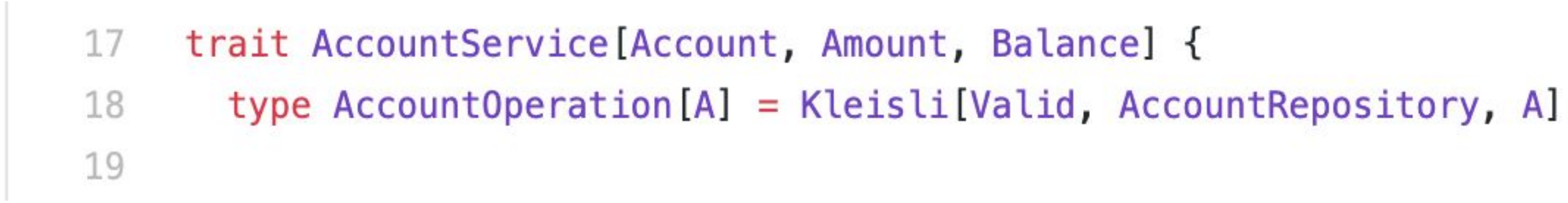
FOREWORD BY JONAS BONÉR

MANNING

@adamgordonbell

Earthly.dev

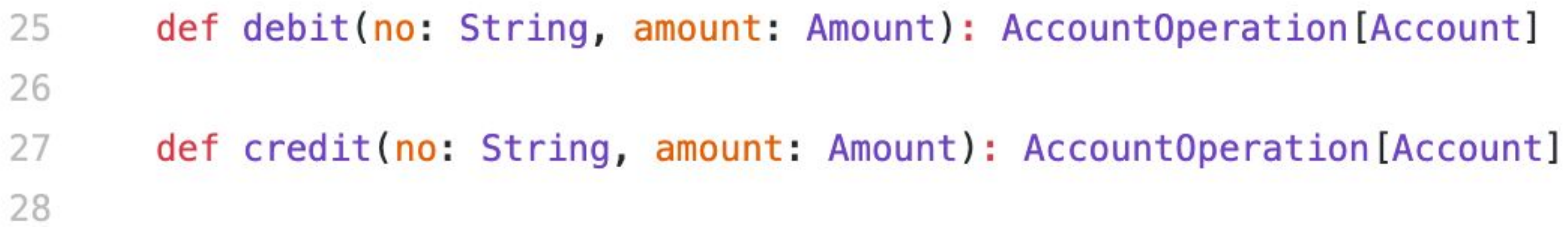
Debashish & DDD



```
trait AccountService[Account, Amount, Balance] {
  type AccountOperation[A] = Kleisli[Valid, AccountRepository, A]

```

Account, Amount Balance are type parameters!

Debashish & DDD



```
    def debit(no: String, amount: Amount): AccountOperation[Account]

    def credit(no: String, amount: Amount): AccountOperation[Account]

```

@adamgordonbell

Earthly.dev

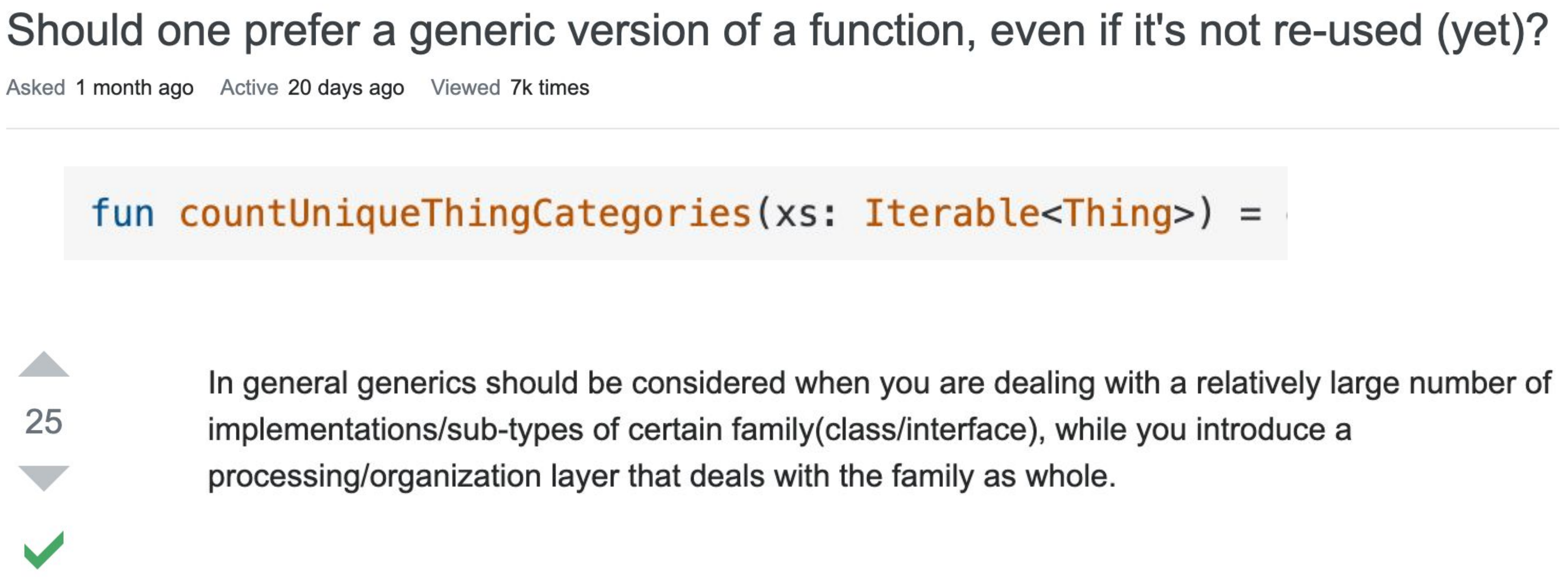
Debashish & DDD

```
def transfer(from: String, to: String, amount: Amount): AccountOperation[(Account, Account)] = for
  a <- debit(from, amount)
  b <- credit(to, amount)
} yield ((a, b))
```

@adamgordonbell

Earthly.dev

Someone is wrong on the internet



Should one prefer a generic version of a function, even if it's not re-used (yet)?

Asked 1 month ago Active 20 days ago Viewed 7k times

```
fun countUniqueThingCategories(xs: Iterable<Thing>) =
```

25

✓

In general generics should be considered when you are dealing with a relatively large number of implementations/sub-types of certain family(class/interface), while you introduce a processing/organization layer that deals with the family as whole.

@adamgordonbell

Earthly.dev

Runar

Functional Programming
in
Scala
Paul Chiusano
Rúnar Bjarnason
Foreword by Martin Odersky
MANNING
A Distributed
Pi-Calculus

Runar

Constraints Liberate, Liberties Constrain

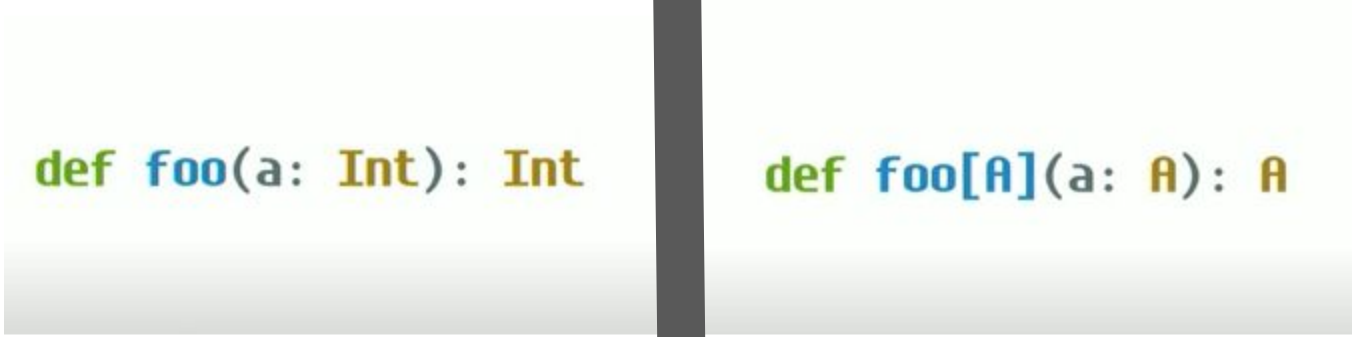
Rúnar Bjarnason @runarorama

Scala World 2015

@adamgordonbell

Earthly.dev

Constraints Are Freedoms



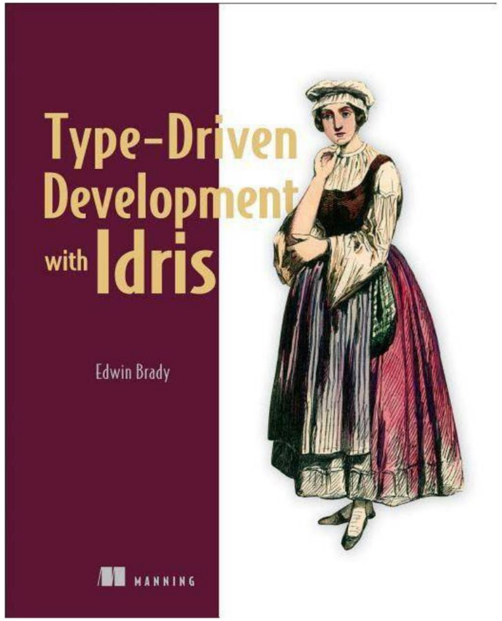
```
def foo(a: Int): Int
```

```
def foo[A](a: A): A
```

@adamgordonbell

Earthly.dev

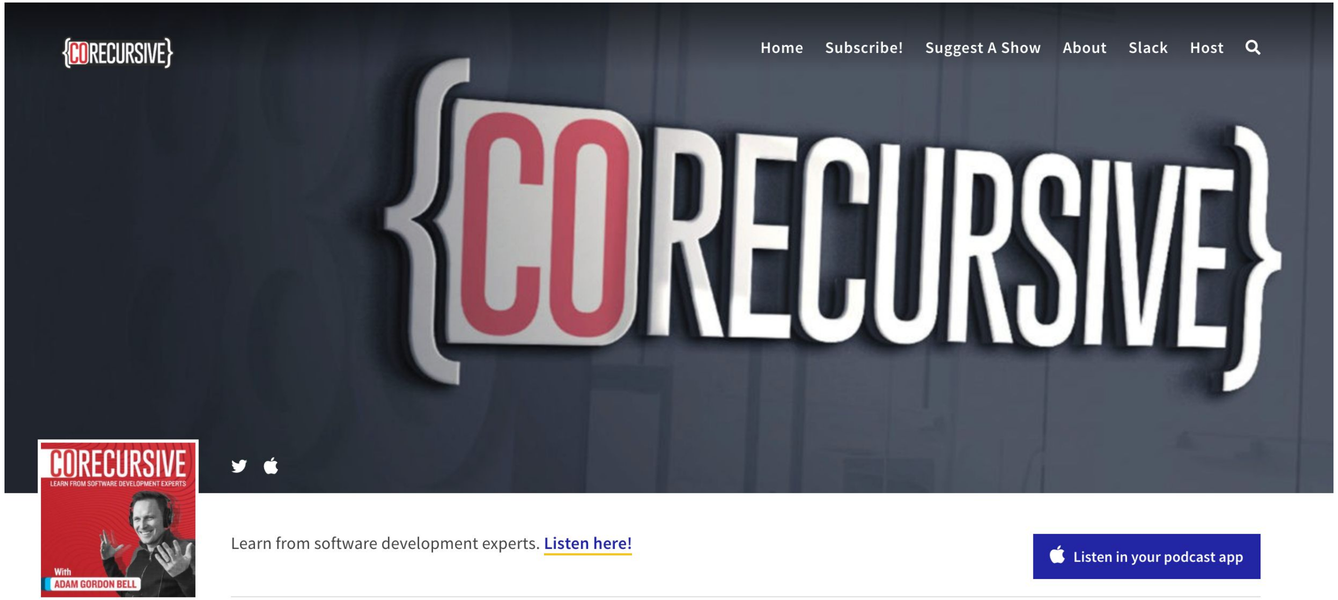
Back At SEDaily



@adamgordonbell

Earthly.dev

A Podcast is born

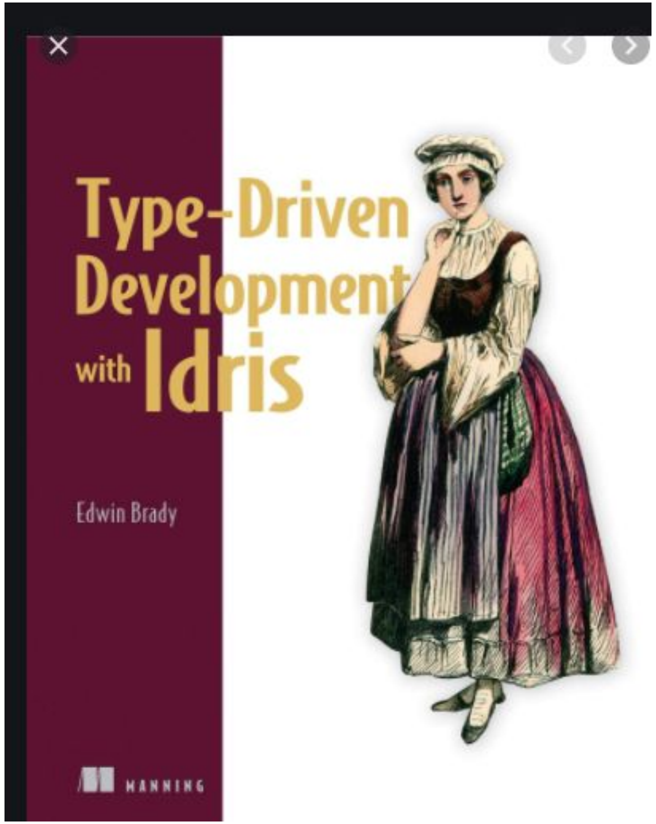


Learn from software development experts. Listen here!

@adamgordonbell

Earthly.dev

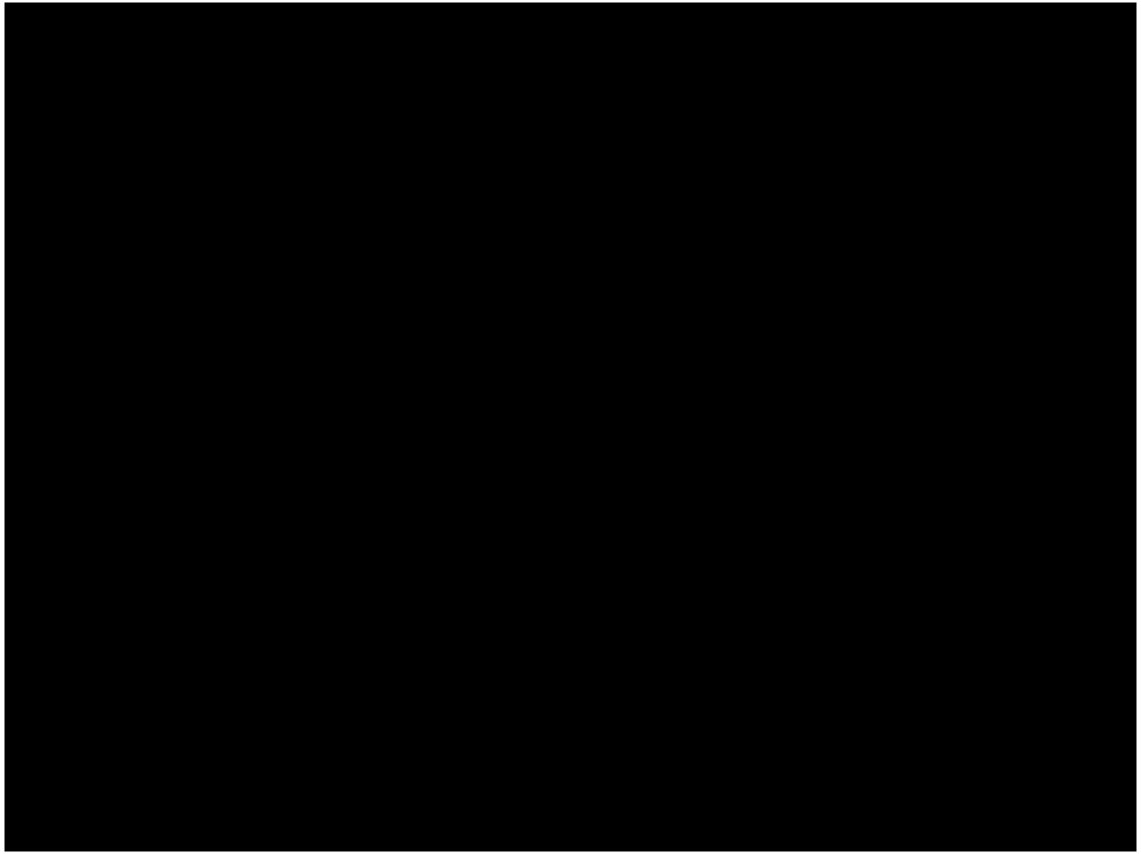
Idris



@adamgordonbell

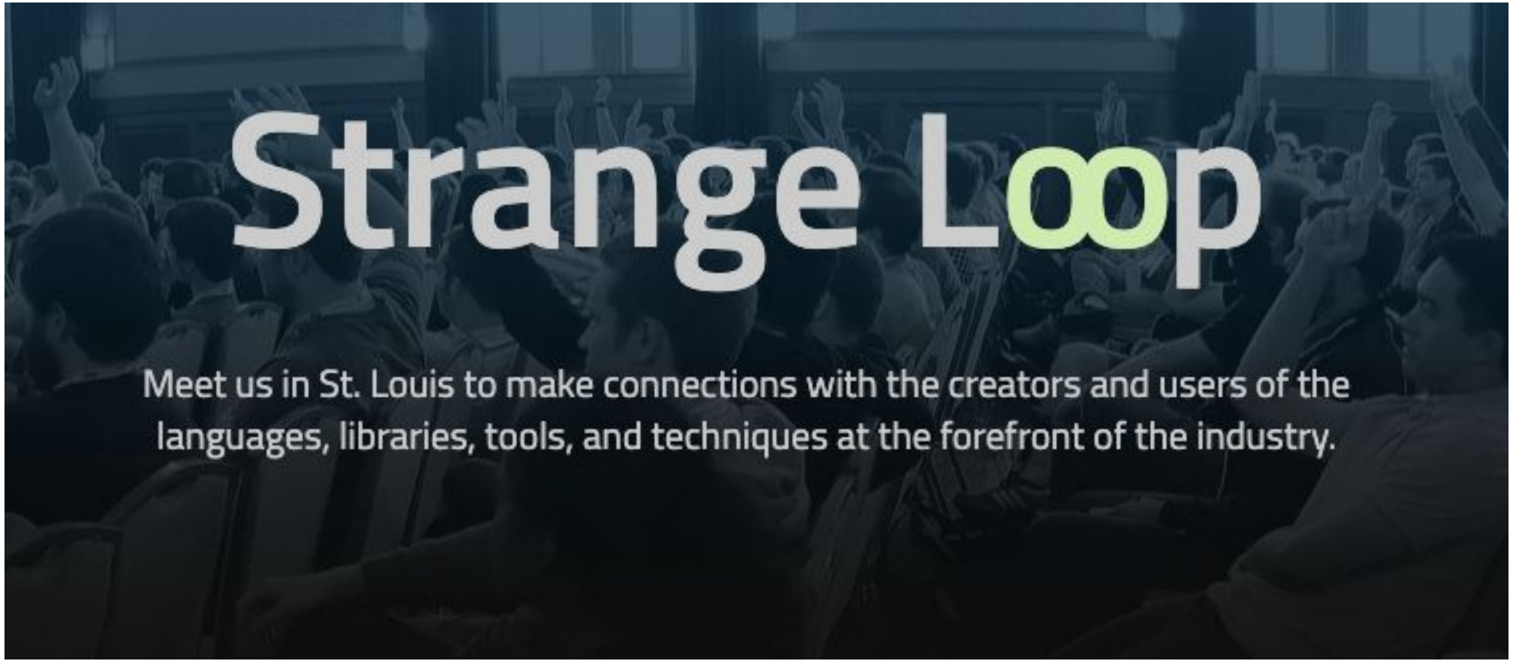
Earthly.dev

Idris



@adamgordonbell

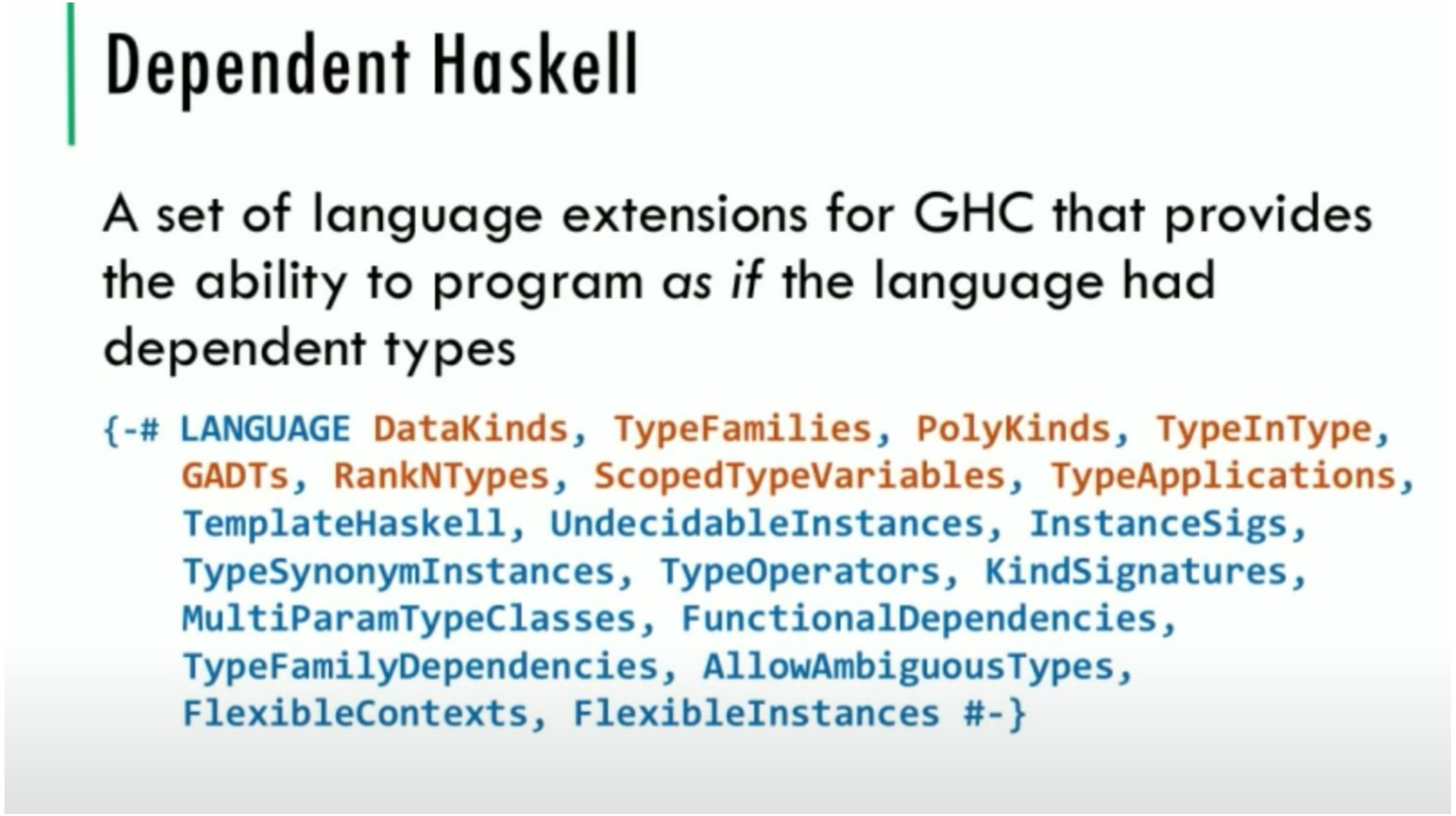
Earthly.dev



Meet us in St. Louis to make connections with the creators and users of the languages, libraries, tools, and techniques at the forefront of the industry.

@adamgordonbell

Earthly.dev

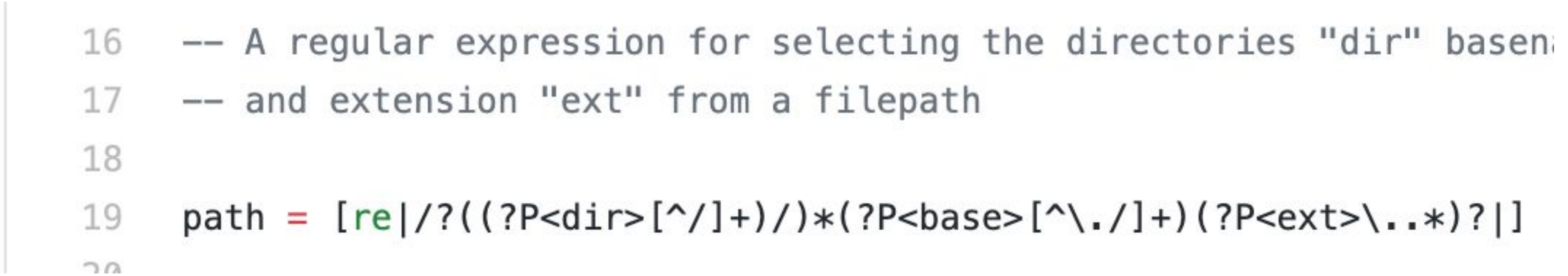


Dependent Haskell

A set of language extensions for GHC that provides the ability to program *as if* the language had dependent types

```
{-# LANGUAGE DataKinds, TypeFamilies, PolyKinds, TypeInType,
    GADTs, RankNTypes, ScopedTypeVariables, TypeApplications,
    TemplateHaskell, UndecidableInstances, InstanceSigs,
    TypeSynonymInstances, TypeOperators, KindSignatures,
    MultiParamTypeClasses, FunctionalDependencies,
    TypeFamilyDependencies, AllowAmbiguousTypes,
    FlexibleContexts, FlexibleInstances #-}
```

Regex for Unix Paths

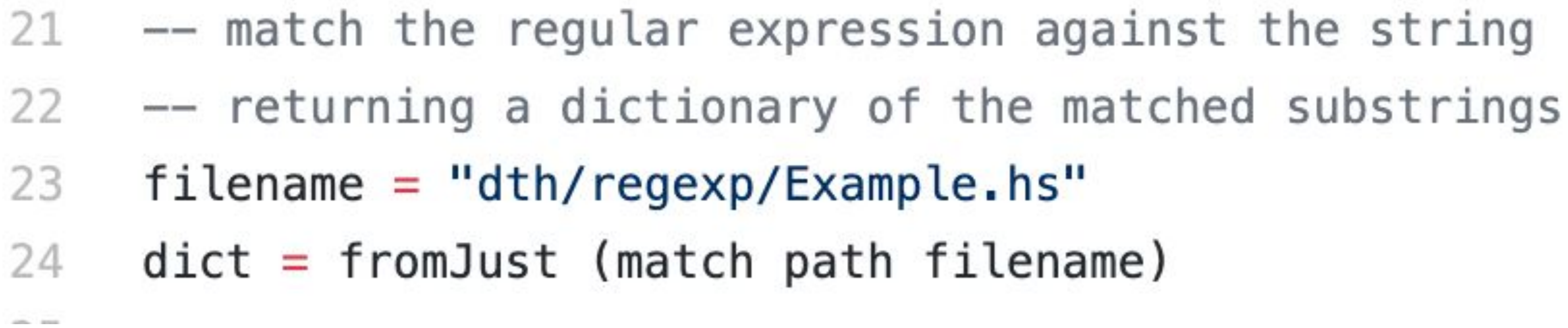


```
-- A regular expression for selecting the directories "dir" basen
-- and extension "ext" from a filepath

path = [re|/?((?P<dir>[^/]+)/)*(?P<base>[^\./]+)(?P<ext>\..*)?|]
```

@adamgordonbell

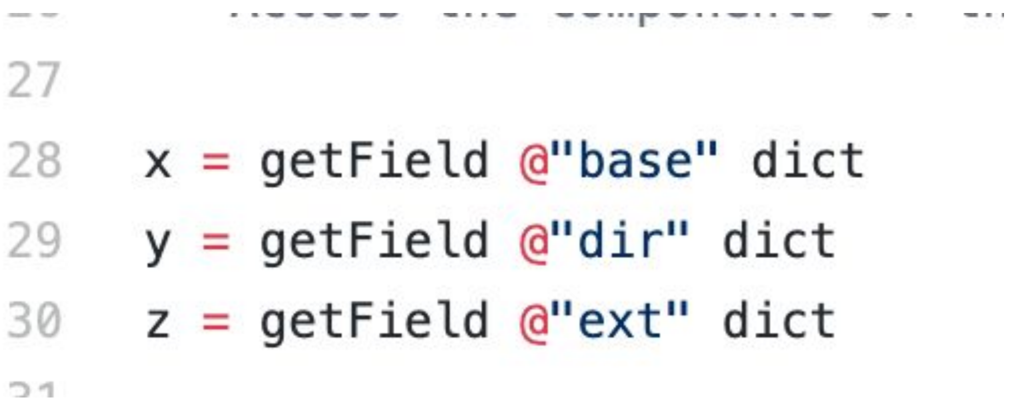
Earthly.dev

Regex for Unix Paths



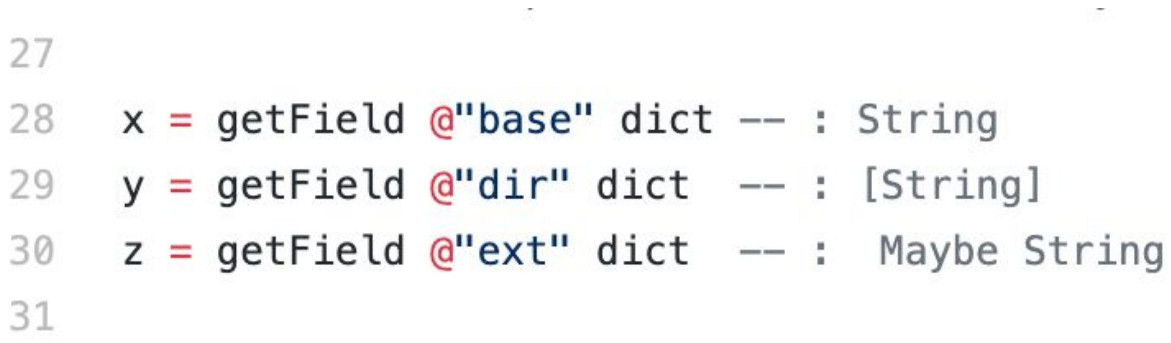
```
-- match the regular expression against the string
-- returning a dictionary of the matched substrings
filename = "dth/regexp/Example.hs"
dict = fromJust (match path filename)
```

Regex for Unix Paths



```
x = getField @"base" dict
y = getField @"dir" dict
z = getField @"ext" dict
```

Regex for Unix Paths

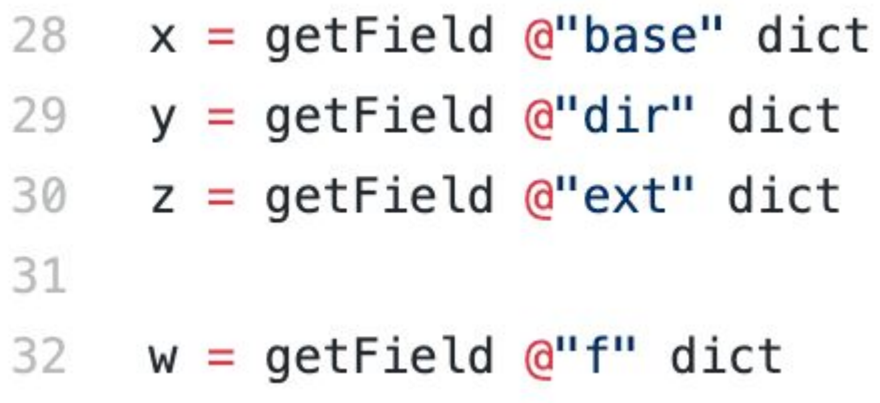


```haskell

x = getField @"base" dict -- : String
y = getField @"dir" dict  -- : [String]
z = getField @"ext" dict  -- :  Maybe String

```

@adamgordonbell

Earthly.dev

What happens if you run this?

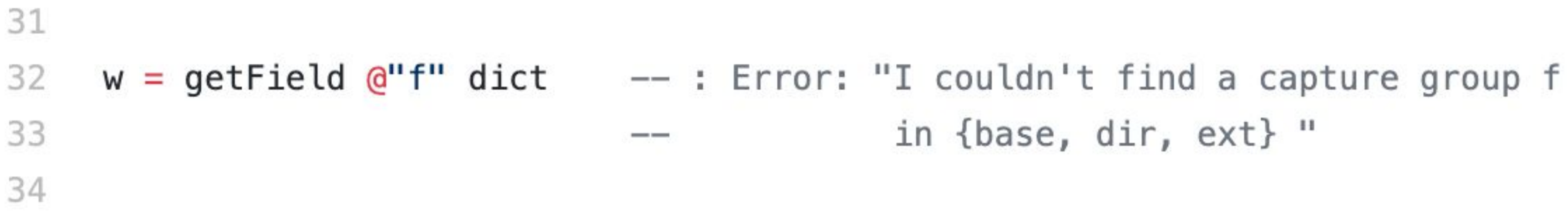


```
x = getField @"base" dict
y = getField @"dir" dict
z = getField @"ext" dict

w = getField @"f" dict
```

@adamgordonbell

Earthly.dev

What happens if you run this?



```haskell

w = getField @"f" dict    -- : Error: "I couldn't find a capture group f
                          --           in {base, dir, ext} "

```

@adamgordonbell

Earthly.dev

Mind blown

@adamgordonbell

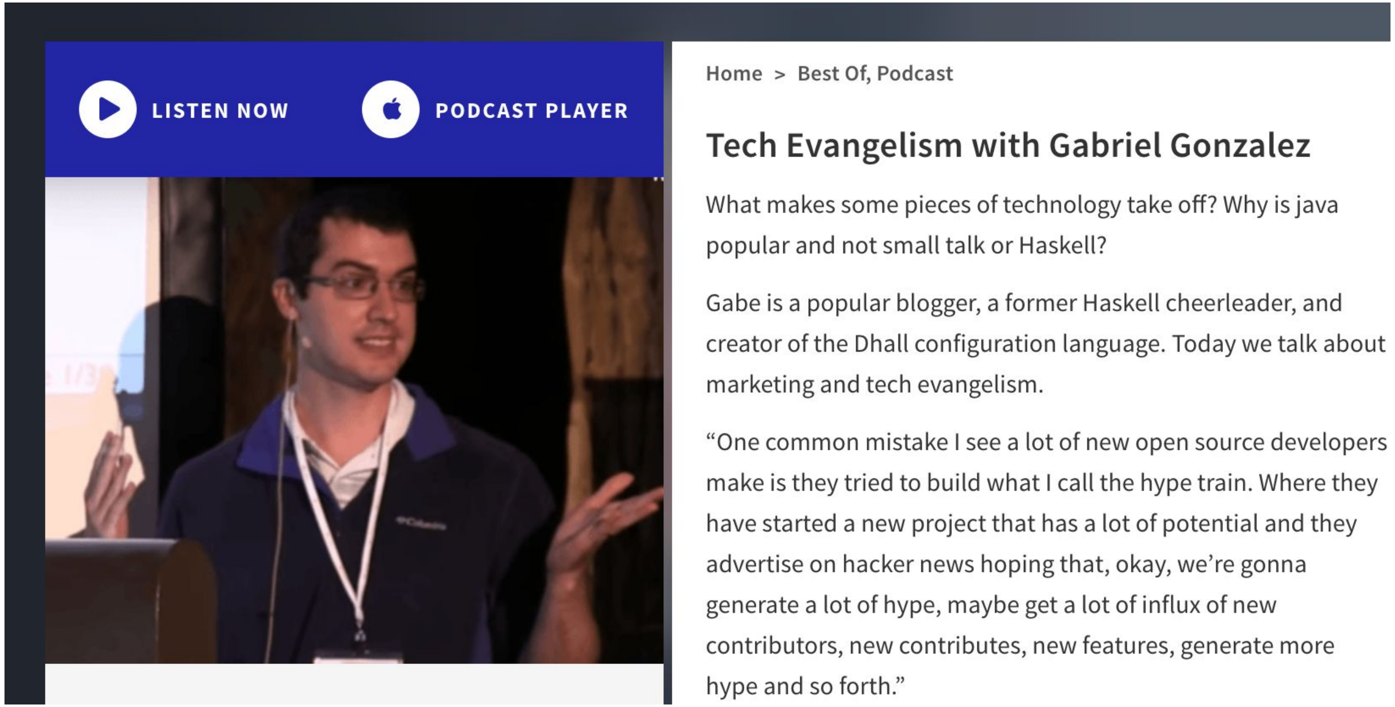
Earthly.dev

Selling FP

@adamgordonbell

Earthly.dev

Tech Evangelism



Home > Best Of, Podcast

Tech Evangelism with Gabriel Gonzalez

What makes some pieces of technology take off? Why is java popular and not small talk or Haskell?

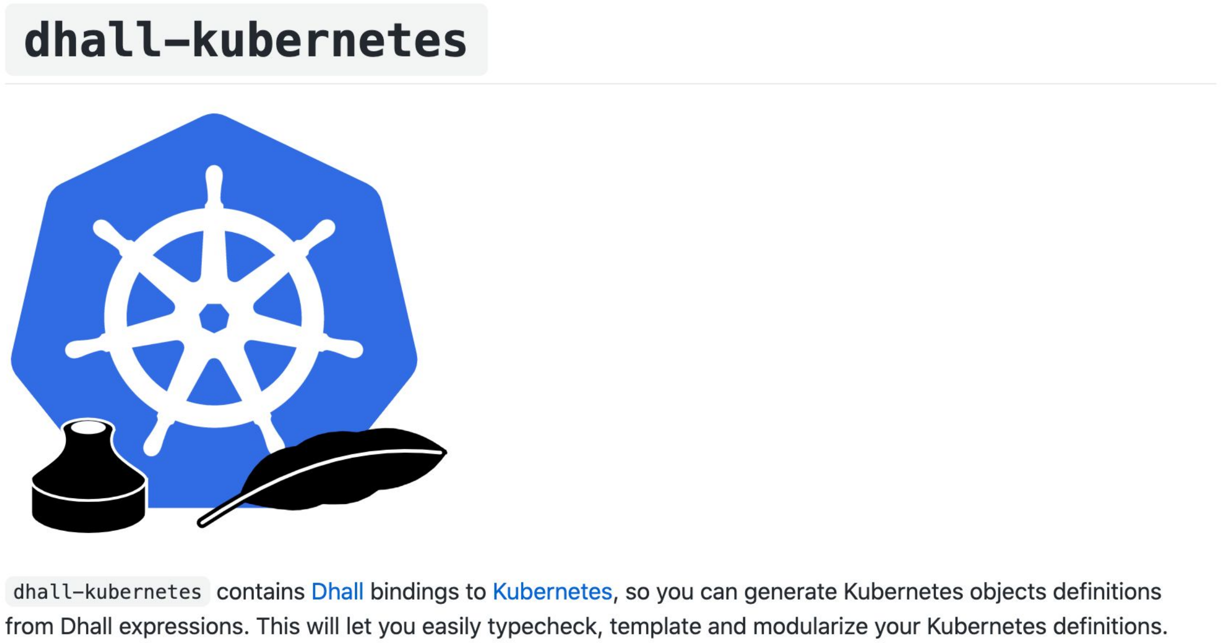
Gabe is a popular blogger, a former Haskell cheerleader, and creator of the Dhall configuration language. Today we talk about marketing and tech evangelism.

“One common mistake I see a lot of new open source developers make is they tried to build what I call the hype train. Where they have started a new project that has a lot of potential and they advertise on hacker news hoping that, okay, we’re gonna generate a lot of hype, maybe get a lot of influx of new contributors, new contributes, new features, generate more hype and so forth.”

@adamgordonbell

Earthly.dev

Tech Evangelism



`dhall-kubernetes`

`dhall-kubernetes` contains Dhall bindings to Kubernetes, so you can generate Kubernetes objects definitions from Dhall expressions. This will let you easily typecheck, template and modularize your Kubernetes definitions.

@adamgordonbell

Earthly.dev

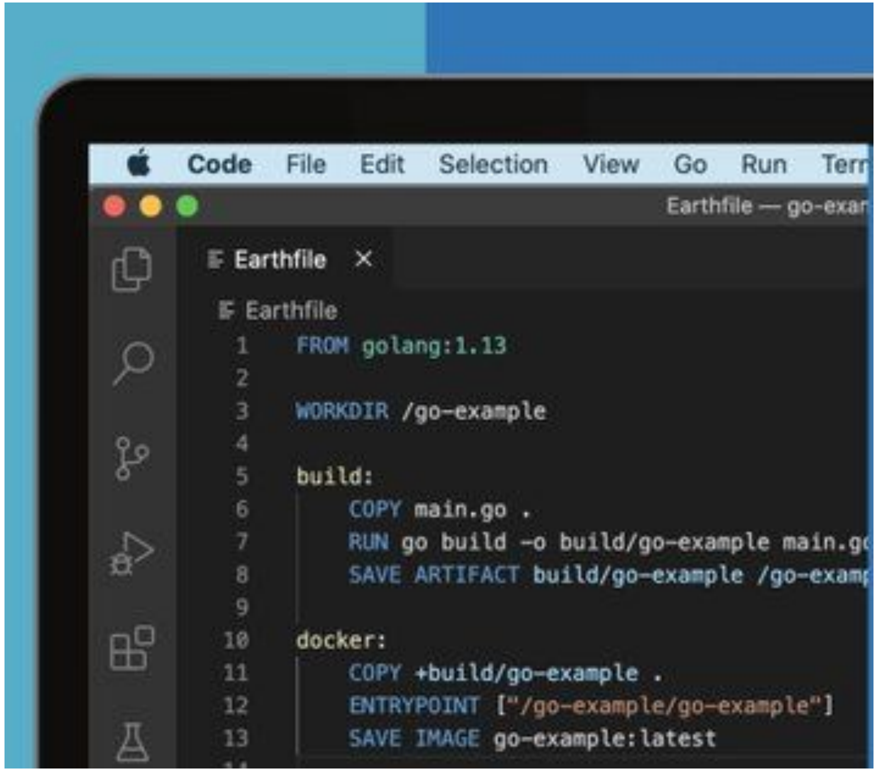
Thank you

EARTHLY

@adamgordonbell

Earthly.dev

Open Source

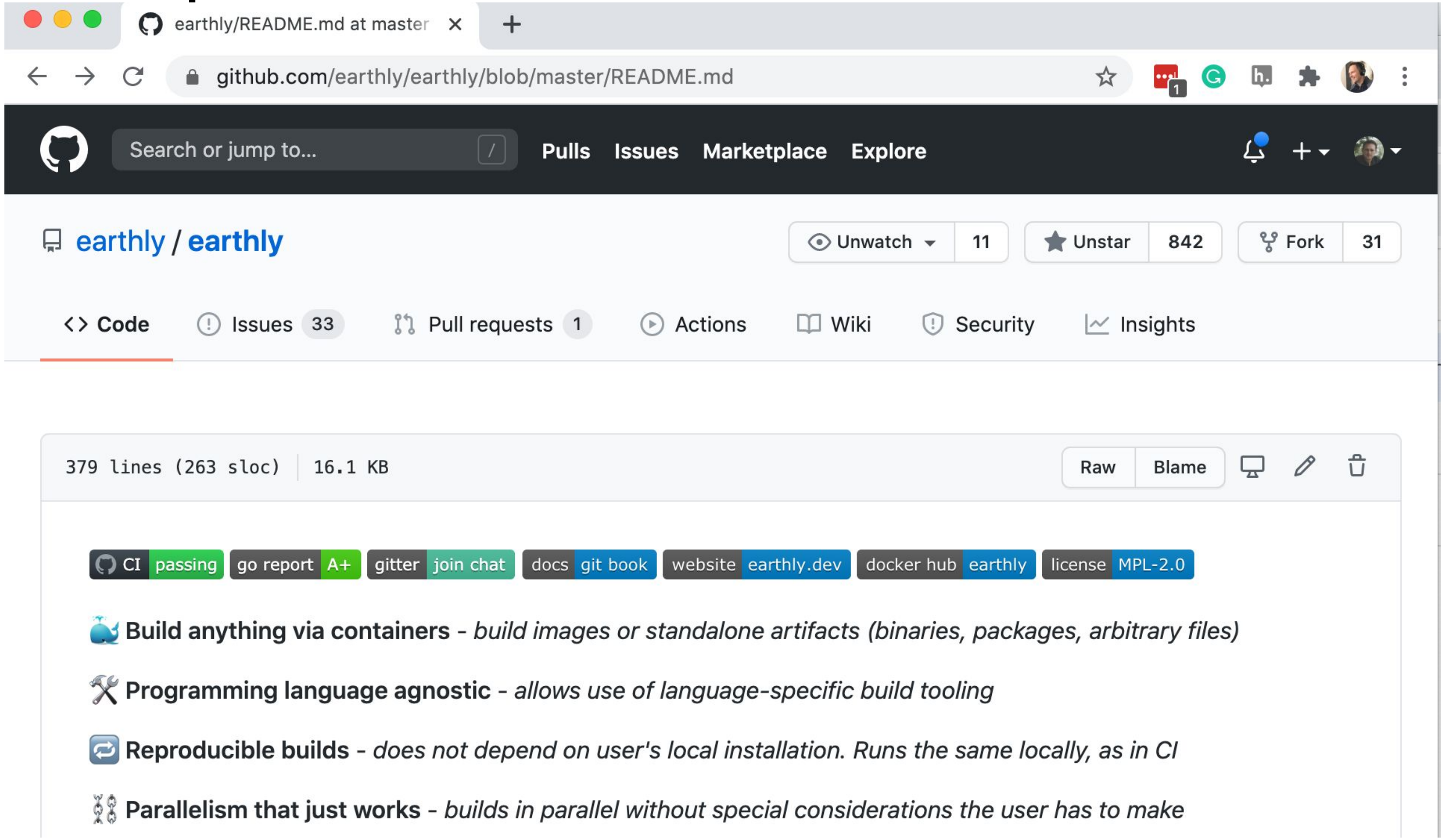


```
FROM golang:1.13

WORKDIR /go-example

build:
    COPY main.go .
    RUN go build -o build/go-example main.go
    SAVE ARTIFACT build/go-example /go-examp

docker:
    COPY +build/go-example .
    ENTRYPOINT ["/go-example/go-example"]
    SAVE IMAGE go-example:latest
```

@adamgordonbell

Earthly.dev

Example 2



earthly/README.md at master

github.com/earthly/earthly/blob/master/README.md

Search or jump to... Pulls Issues Marketplace Explore

earthly / earthly

Unwatch 11 | Unstar 842 | Fork 31

Code | Issues 33 | Pull requests 1 | Actions | Wiki | Security | Insights

379 lines (263 sloc) | 16.1 KB

Raw | Blame

CI passing | go report A+ | gitter join chat | docs git book | website earthly.dev | docker hub earthly | license MPL-2.0

🐳 **Build anything via containers** - *build images or standalone artifacts (binaries, packages, arbitrary files)*

🛠 **Programming language agnostic** - *allows use of language-specific build tooling*

🔁 **Reproducible builds** - *does not depend on user's local installation. Runs the same locally, as in CI*

⛓ **Parallelism that just works** - *builds in parallel without special considerations the user has to make*