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

@adamgordonbell

Earthly.dev
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.

Denys Shabalin @den_sh · Sep 11, 2017
I'd love to come! Let's discuss the details over the email.
Learning FP

@adamgordonbell
Debashish & DDD

@adamgordonbell

Earthly.dev
trait AccountService[Account, Amount, Balance] {
    type AccountOperation[A] = Kleisli[Valid, AccountRepository, A]
}

Account, Amount Balance are type parameters!
def debit(no: String, amount: Amount): AccountOperation[Account]
def credit(no: String, amount: Amount): AccountOperation[Account]
def transfer(from: String, to: String, amount: Amount): AccountOperation[((Account, Account)] = for 
  a <- debit(from, amount)
  b <- credit(to, amount)
} yield ((a, b))
Someone is wrong on the internet

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

fun countUniqueThingCategories(xs: Iterable<Thing>) =

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.
Constraints Are Freedoms

def foo(a: Int): Int
def foo[A](a: A): A
Back At SEDaily

Type-Driven Development with Idris

@adamgordonbell
A Podcast is born

Learn from software development experts. Listen here!
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.
Dependent Haskell

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

```haskell
{-# 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

```python
16   -- A regular expression for selecting the directories "dir" basen
17   -- and extension "ext" from a filepath
18
19   path = [re]/*(?P<dir>[^/]+)*/(?P<base>[^\./]+)(?P<ext>\..*)?[/]
```
Regex for Unix Paths

```
21 -- match the regular expression against the string
22 -- returning a dictionary of the matched substrings
23 filename = "dth/regexp/Example.hs"
24 dict = fromJust (match path filename)
```
Regex for Unix Paths

```
27
28  x = getField @"base" dict
29  y = getField @"dir" dict
30  z = getField @"ext" dict
```
Regex for Unix Paths

27
28 x = getField @"base" dict -- : String
29 y = getField @"dir" dict -- : [String]
30 z = getField @"ext" dict -- : Maybe String
What happens if you run this?

```python
28  x = getField "base" dict
29  y = getField "dir" dict
30  z = getField "ext" dict
31
32  w = getField "f" dict
```
What happens if you run this?

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

@adamgordonbell

Earthly.dev
Selling FP
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.”
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.
Thank you

@adamgordonbell
Open Source

@adamgordonbell

Earthly.dev
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

@adamgordonbell

Earthly.dev