

flatMappy bird

Martin Carolan

<https://github.com/mcarolan/flatmappy-bird>

@mcarolan88

mail@mcarolan.net

goals:

FRP – A Short Primer by Gergely Patai at Helsinki FRP Meetup

April 1, 2015

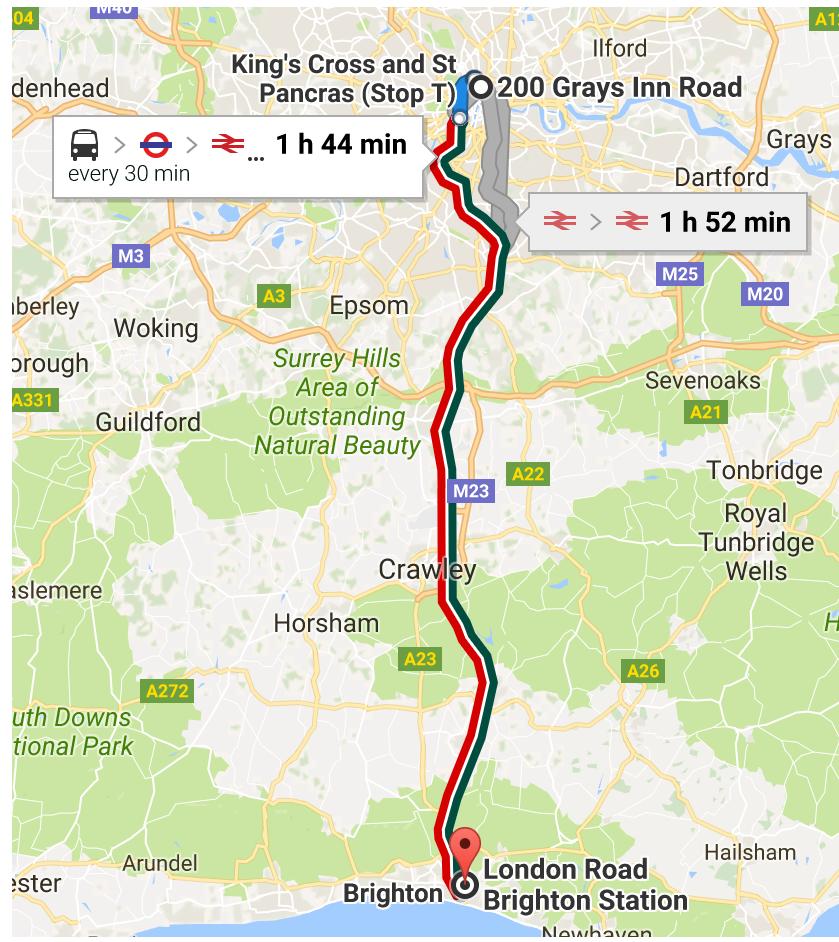
build a game out of referentially transparent functions

Treat the whole lifetime of a time-varying quantity as a value

https://www.youtube.com/watch?v=_BICQhz6tkM

Behaviors -> player falls, pipe moves. Function of time =>
???





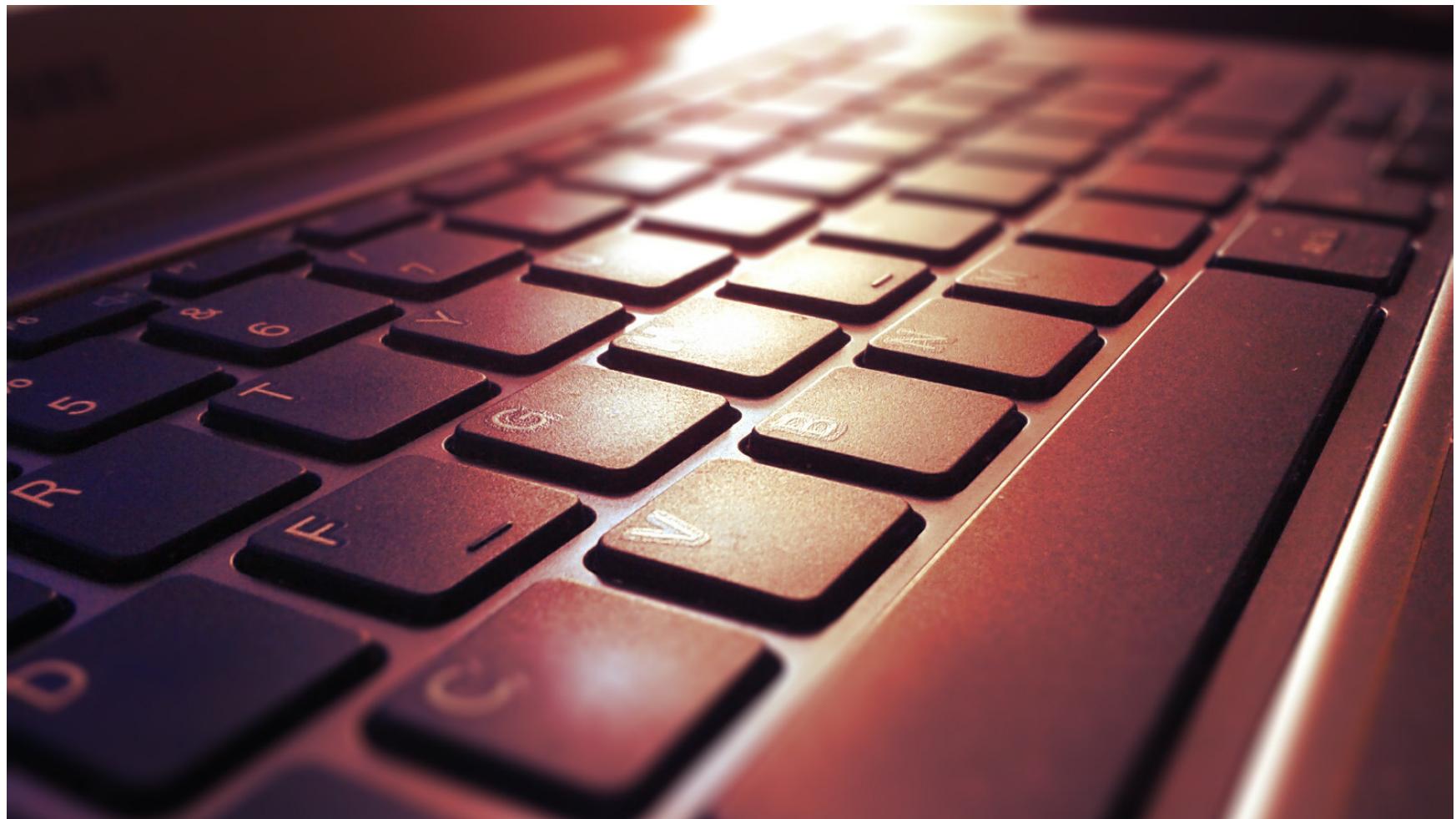
















Hands-on Scala.js

www.lihaoyi.com/hands-on-scala-js/ haoyi scala workbe →

Dashboard [Jenki... Dashboard [Jenki... Details Puppet [Jenkins] TemplateSyntaxE... New GIG Team St... CDM CDM Servers

Hands-on Scala.js

Intro to Scala.js

About Javascript

About Scala.js

Hands On

Getting Started

Making a Canvas App

Interactive Web Pages

The Command Line

Cross Publishing Libraries

Integrating Client-Server

In Depth

Advanced Techniques

Hands-on Scala.js

Writing client-side web applications in Scala

`var x = 0.0`

`type Graph = (String, Double => Double)`

`val graphs = Seq[Graph](`

`("red", sin),`

`("green", x => abs(x % 4 - 2) - 1),`

`("blue", x => sin(x/12) * sin(x))`

`).zipWithIndex`

`dom.setInterval(() => {`

`x = (x + 1) % w; if (x == 0) clear()`

`for (((color, f), i) <- graphs) {`

`val offset = h / 3 * (i + 0.5)`

`val y = f(x / w * 75) * h / 30`

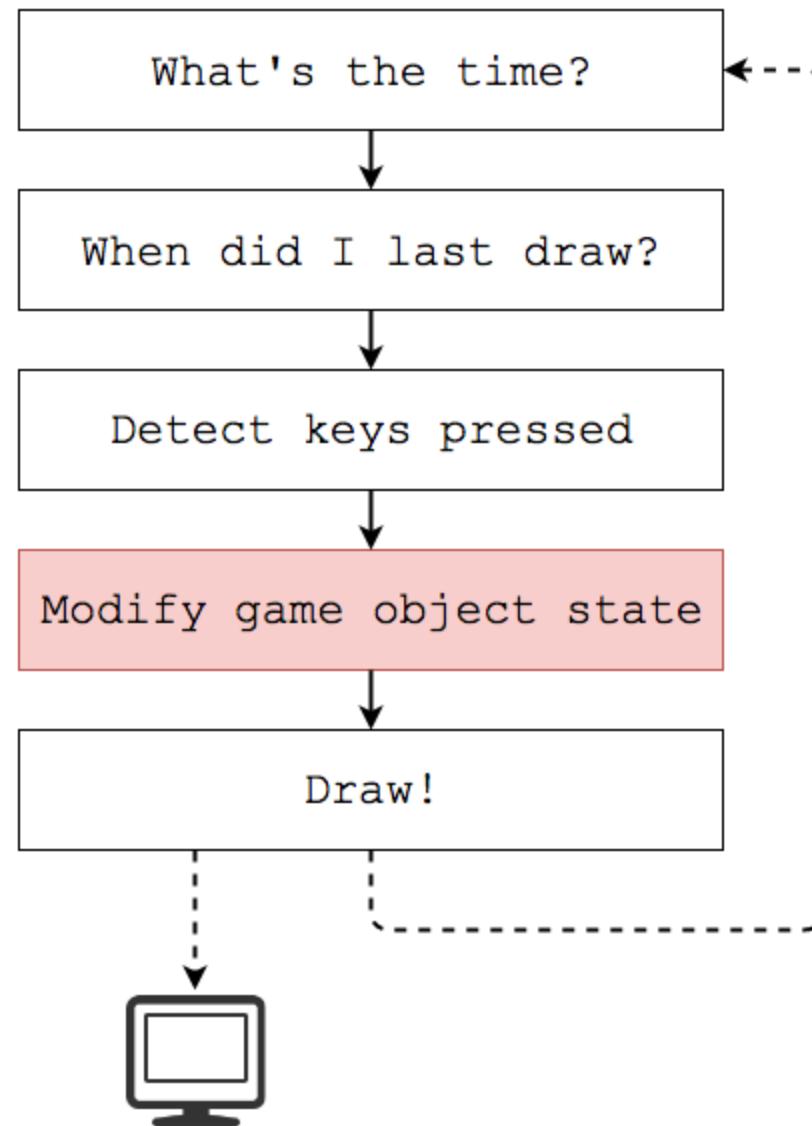
`brush.fillStyle = color`

`brush.fillRect(x, y + offset, 3, 3)`

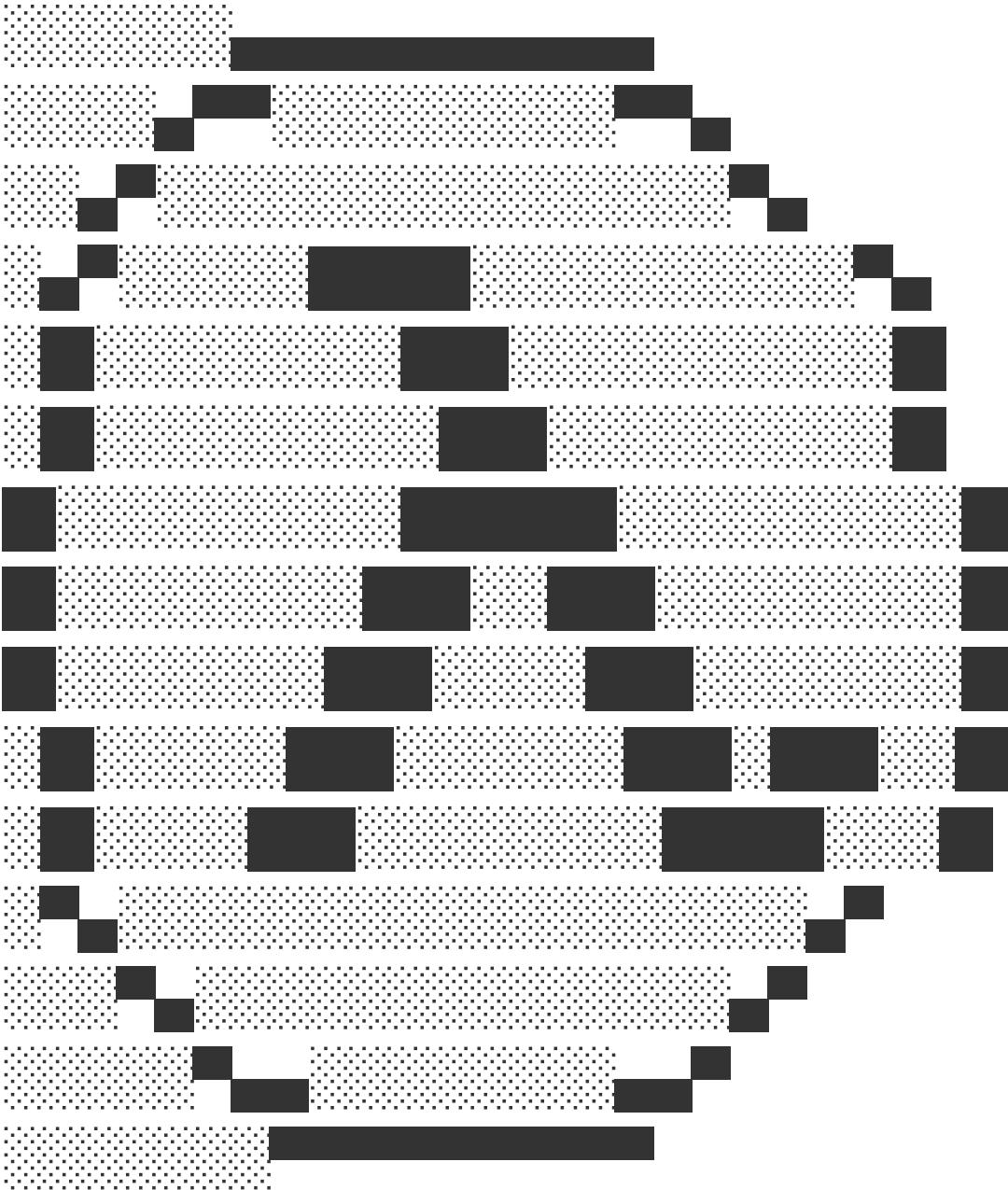
`}`

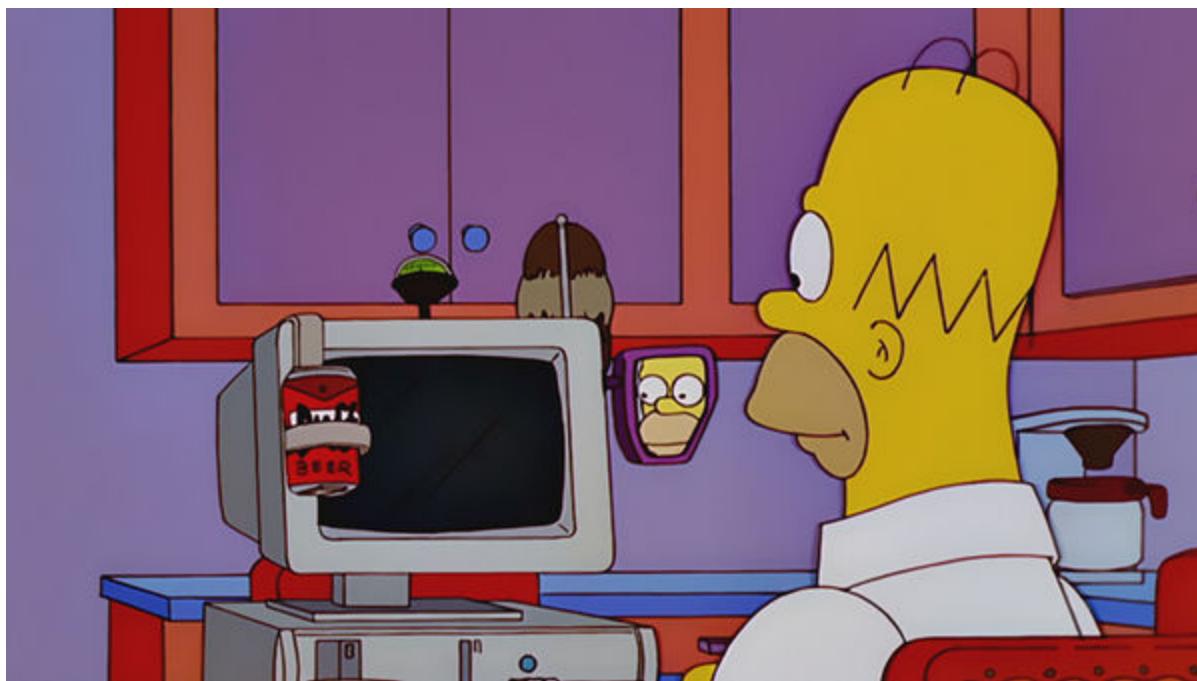


<http://www.lihaoyi.com/hands-on-scala-js/>









This repository Search Pull requests Issues Gist

Watch 44 Star 249 Fork 7

Code Issues 1 Pull requests 1 Projects 0 Wiki Pulse Graphs

Branch: master codeblog / 2012 / 2012-01-17-declarative-game-logic-afrp.md Find file Copy path

blastrock Fixed typos ca581f2 on 1 Oct 2013

2 contributors

429 lines (301 sloc) 18.2 KB

Raw Blame History

Purely Functional, Declarative Game Logic Using Reactive Programming

In the [previous article](#) I introduced the `Coroutine` data type. In this second part I will show how coroutines can be used to implement a fixed time-step reactive programming library and use that library for modeling a simple game. The code examples will require a [basic proficiency](#) in reading Haskell code.

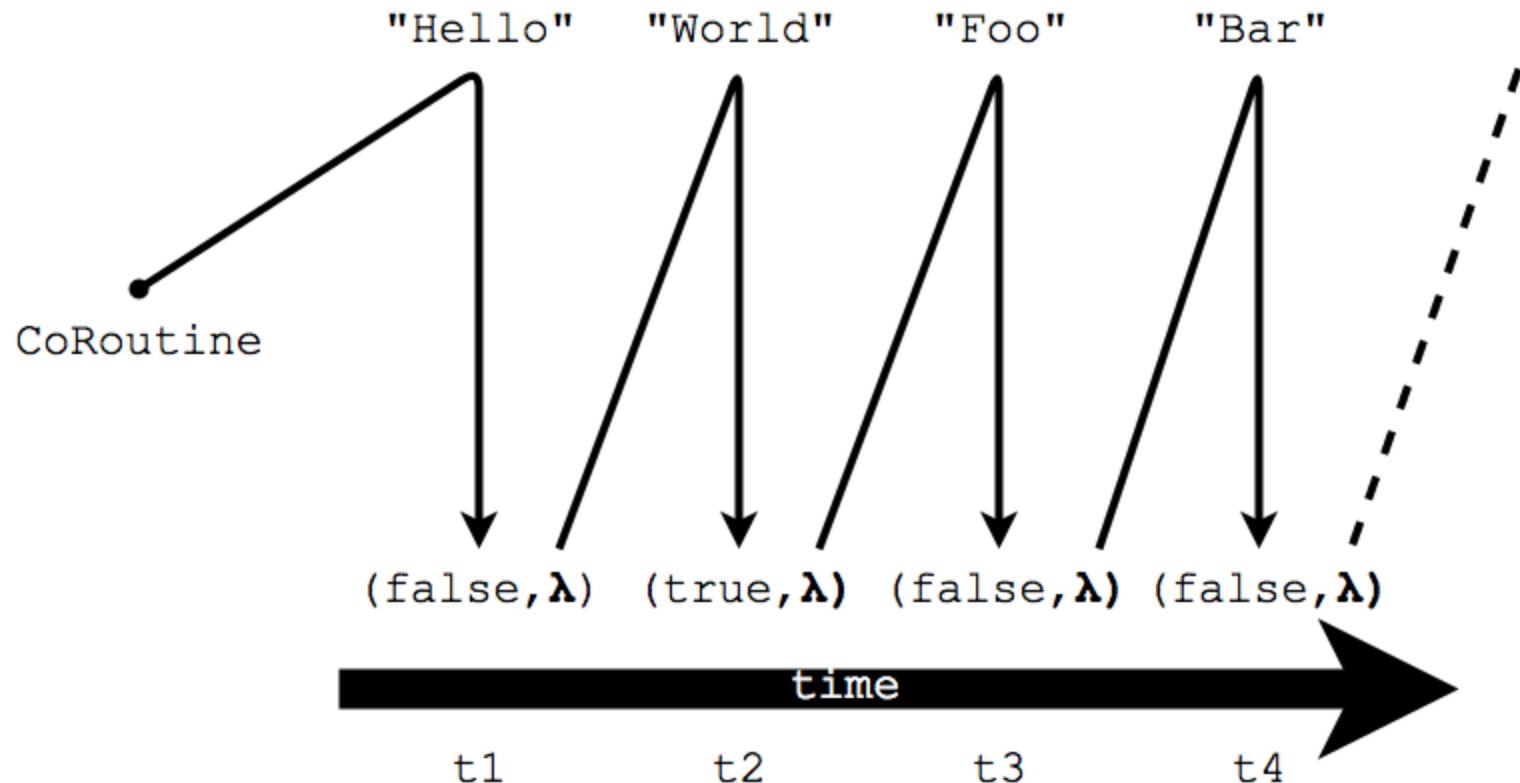
<https://github.com/leonidas/codeblog/blob/master/2012/2012-01-08-streams-coroutines.md>

<https://github.com/leonidas/codeblog/blob/master/2012/2012-01-17-declarative-game-logic-afrp.md>

CoRoutine

Problem: identify when the provided string is "World" AND the previously provided string is "Hello"

Problem: identify when the provided string is "World" AND the previously provided string is "Hello"



```
trait CoRoutine[A, B] {  
    def apply(input: A): (B, CoRoutine[A, B])  
}
```

```
trait CoRoutine[A, B] {
    def apply(input: A): (B, CoRoutine[A, B])
}

object CoRoutine {
    def lift[A, B](f: A => (B, CoRoutine[A, B])): CoRoutine[A, B]
        new CoRoutine[A, B] {
            override def apply(input: A): (B, CoRoutine[A, B]) =
                f(input)
        }
}
```

7 minutes

How does this relate to games programming?

Redraw gameloop as game function

1. Intro
2. Fundamentals of a CoRoutine
3. How does can this be applied to a game?
4. Case study: flatMappy bird player
5. When would you use this?
6. Why would you use this?
7. Questions

<https://www.flickr.com/photos/jonathankosread/6184019357/in/photolist-aqsHvp-2g17mh-dR3Y9x-gponG-kKZytZ-pNhwM5-77ihw8-oTwyQb-dteNS-bkSYHH-4u3pAF-a6Yj8h-6pypTD-kZSzJ3-fcryvD-ajC5LU-9De9QT-7zizEa-eCgF8-kj3Ru-eV1rEG-aEuC9-6dVK9j-6pypBR-jm7aq-5g1duM-xoEbF2-o5epCf-6pCXbS-6h6Zu8-6pyiwZ-6pCvB7-6pCy21-4cog75-7AcTfS-e6XAqK-4uo2MP-6pCs1m-6ASZJU-e1YtBB-6huVMT-d2iPZW-8jEqvi-977fud-6A7aJZ-6vmzV-gjmyZ-69QmoH-8cNM7c-pTTjA>

<http://www.deviantart.com/art/Gaming-Icons-179546229>

<https://www.flickr.com/photos/marianopaulin02/16895155137/in/photolist-rJY61e-dN1SXG-49sN6T-dZa635-tfS3xJ-tfTcFJ-xNNHo6-bvVDie-wR9Mt9-5Z5q7d-o4EsFw-7q9qBF-6dYoeL-4qyEp4-cRGCeL-72tLMF-bFxRCV-5tWxwf-5tWxV3-7x7o1q-d8atD7-5tSaJe-4nWKA7-ese74Y-xAMYw-4myiwF-oqisgw-cRGChh-9BVHmk-7dcFo-8jqLdc-6sHJFL-6yq1bj-g9FyfG-DWtde-7bNZYS-cRGCD1-662Gp4-cRJxT1-7edja8->

