U0GPGFQQY : Just as a brain bending exercise. `a` is an `id`, and `fn` is a constructor of the record, I want to have tuples with `(id, record)` to create a Dict out of them
U17MSA88M : <@U0GPGFQQY> does elm have something like tuple sections?
U17MSA88M : if there is a tuple constructor function you definitely can
U0GPGFQQY : `(,)` is a tuple constructor
U17MSA88M : `(,) :: a -> b -> (a, b)`
U17MSA88M : ?
U0GPGFQQY : Yes, this would construct a tuple
U17MSA88M : <http://pointfree.io/>
U17MSA88M : returns `f = ap ((.) . (.) . (.) . (,))`
U17MSA88M : `ap` is defined as `ap f g = \x -> f x (g x)`
U0GPGFQQY : oh my god
```
f : (a -&gt; b -&gt; c -&gt; d -&gt; x) -&gt; a -&gt; b -&gt; c -&gt; d -&gt; ( a, x )
f fn a =
    ((&lt;&lt;) &lt;&lt; (&lt;&lt;) &lt;&lt; (&lt;&lt;) &lt;&lt; (,)) a (fn a)
```

U17MSA88M : do you not have `ap` in Elm? ^^ It's the applicative instance for Function or S combinator
U0GPGFQQY : It's not included in the Higher-Order Helpers section here
<http://package.elm-lang.org/packages/elm-lang/core/5.1.1/Basics>
U0GPGFQQY : But I guess it's fine cause it is easy to define myself
U17MSA88M : I think we're obscure enough at this point
U17MSA88M : marvelous :slightly_smiling_face:
U0GPGFQQY : My original code is much better and more understandable
U0F01KLV6 : Wow... That is sick
U17MSA88M : > My original code is much better and more understandable
Oh no doubt :smile:

U17MSA88M : This talk describes the trade-offs in point free and how it's actually a spectrum really well
<https://www.youtube.com/watch?v=seVSlKazsNk>
U0GPGFQQY : Oh, cool, thanks for reminding that I should watch this video. Amar is my colleague at SoundCloud
U17MSA88M : Oh wow, you're lucky! Extend my thanks to him for the talk if you happen to talk to him
:slightly_smiling_face:
U3LUC6SNS : To the extent that this has value (other than fun), it is more esthetic than scientific. He he he!
:slightly_smiling_face:
It is important to have alpha < 1.0 so that colors mix.

U17MSA88M : Pretty :star:
U3LUC6SNS : Thx!
U0GPGFQQY : kritzcreek:
this may look a little nicer

```
((&lt;&lt;) &lt;&lt; (&lt;&lt;) &lt;&lt; (&lt;&lt;)) ((,) a) (fn a)
```

U3LUC6SNS : That is beginning to look like Lisp
U2LAL86AY : <@U3LUC6SNS> super ! Can this be used for creating particle effects? Is it performant ?
U3LUC6SNS : I don't know how performant it is -- I should try with many particles. For best results, I think the `Graph` library that I wrote for this is an interface to SVG. I don't thinks it is very efficient; it should be redone in WebGL. <@U2FP79HN3> posted something very interesting in this regard -- please see his Ellie: <https://ellie-app.com/3vwGN4c4fTga1/0> . I plan to rewrite `Graph` in the next few days in WebGL following <@U2FP79HN3> and will post the code for you.
U3LUC6SNS : <@U2LAL86AY> Here is the reference for WebGL:
<http://package.elm-lang.org/packages/elm-community/webgl/latest>
U2FP79HN3 : Heehee
U2FP79HN3 : Triangles and squares were fun
U2FP79HN3 : this was more fun <http://cloud.jorisooms.be/1s3J3R1p0U13> <@U3LUC6SNS>
U3LUC6SNS : <@U2FP79HN3>, that is beautiful!

U55CZT6T1 : What are the common approaches to profiling Elm apps?