Index of the slides:

Monads

Fix a monoid M.
Notation: elements of M will be called a,b,c,...
Our archetypal monad is based on the functor (×M):Set->Set.
A monad over a functor T (T=(×M) for us) is triple (T,η,μ)
where η and μ are natural transformations:

```
     η          μ                                                  .
  X -----> TX <----- TTX      x |---> x,1
                                      x,ab <---| x,a,b
```

η and μ have to obey certain equations:

```
  (ηT;μ)=(Tη;μ)=id and
  (μT;μ)=(Tμ;μ).
```

If we downcase those equations using x=>x,a as the archetype
they become exactly ∀a.a1=1a=a and ∀a,b,c.a(bc)=(ab)c.

```
    TX --ηT--> TTX --μ--> TX       x,a |---> x,a,1 |----> x,a1
    TX --Tη--> TTX --μ--> TX       x,a |---> x,1,a |----> x,1a
    TX ---------id------> TX       x,a |----------------> x,a

  TTTX --μT--> TTX --μ--> TX   x,a,b,c |---> x,a,bc |---> x,a(bc)
  TTTX --Tμ--> TTX --μ--> TX   x,a,b,c |---> x,ab,c |---> x,(ab)c
```

Here are square conditions for the two natural transformations.
For any objects X and Y and any morphism x|->y,

```
       ηX
   X -----> TX      x |---> x,1
   |        |       -        -
   | f      | Tf    |        |
   v        v       v        v
   Y -----> TY      y |---> y,1
       ηY

       μX
  TX <---- TTX     x,ab <---| x,a,b
   |        |       -           -
   | Tf     | TTf   |           |
   v        v       v           v
  TY <---- TTY     y,ab <---| y,a,b
       μY
```

Where our notation fails

Our notation based on (×M) is not totally faithful -
sometimes it suggests that two morphisms are the same
when they may be different...
The square below does not need to commute.

```
         TTTX                            x,a,b,cd
       /      ^                        \          ^
  TμX /        \ μTTX                 /            \
     v          \                    v              /
  TTX             TTTTX    x,ab,cd                    x,a,b,c,d
     ^          /                    ^              \
  μTX \        / TTμX                 \            /
       \      v                        /          v
         TTTX                            x,ab,c,d
```

But all canonical morphisms from T^nX to TX are equal -

A presentation of monads due to Manes

```
A monad can also be given by an object function S
and operations

     f                 f#
 (X ---> S(Y)) |-> (S(X) ----> S(Y)     (x|->y,a) |-> (x,b|->y,ab)

          ηX
 X |--> (X ----> S(X))                  x -.> (x|->x,1)

Subject to:

   (ηX)^# = 1_S(X)
   ηX;f^# = f
 f^#;g^# = (f;g^#)^#

Let's expand these equations:

   ηX                x|->x,1
 ------             ----------
 (ηX)^# = 1_S(X)    x,a|->x,1a = x,a|->x,a

      f                      x|->y,a
     ---                     ----------
 ηX   f^#            x|->x,1  x,b|->y,ab
 --------             -------------------
  ηX;f^#  = f           x|->y,a1        = x|->y,a
                   g                                       y|->z,a
                  ---                                     ----------
  f     g     f   g^#     x|->y,c     y|->z,a   x|->y,c  y,b|->z,ab
 ---   ---   -------    ----------   ----------  -------------------
 f^#   g^#    f;g^#     x,d|->y,cd   y,b|->z,ab       x|->z,ac
 ---------   ---------  -----------------------     -----------
  f^#;g^#  = (f;g^#)^#      x,d|->z,acd         =    x,d|->z,acd
```

The Kleisli category of a monad

```
Notation:
            x:xT    x     xT     xTT
          -  -        -
          |  | f       \ f
          |  v          v
  f;TG;mu | y:yT    y     yT     yTT
          |  -        -      -
          |  | g     g \      \ Tg
          v  v          v      v
            z:zT    z     zT <-| zTT
                             mu
The adjunction:

      x':x'T <===== x'
        -           -
  f;eta |    <--|   |
        v           v
       x:xT <====== x'
        -           -
      g |    <--|   | g
      g |    |-->   | g
        v           v
       y:yT ======> yT
        -           -
      h |    |-->   | Th;mu
        v           v
      y':y'T ====> y'T
```

The Eilenberg-Moore category of a monad

The adjunction:

```
          (x'T <--| x'TT) <=== x'
             -        -        -
      f;eta |        |   <-|   | f
             v        v        v
          (xT <---| xTT) <==== x
        --|     ---|           -
Tg;\aa /       /      <--|     | g
    h /       /      |-->      | eta;h
     v \aa   v                 v
  (y <---| yT) <============== y
    -       -                  -
 k |       |        |-->       | k
   v       v                   v
  (y' <--| y'T) <============= y'
```

The category of resolutions for a monad

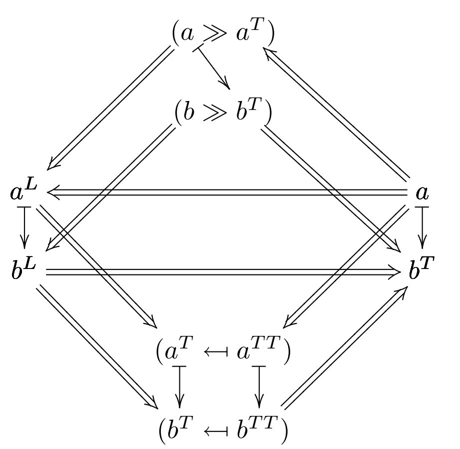
```
The Kleisli category is the initial resolution,
the Eilenbeg-Moore category is the final one.

Notation: O[yL] is not necessarily an image by L -
in some arrows it just just an arbitrary object of
the "other category" in an adjunction diagram.
Also: O[yT] = O[yLR].

            /=== x:xT <==\
           //     -      \\
          //      |       \\
         //       v        \\
        //   /== y:yT ===\  \\
       //   //           \\  \\
      //   //             \\  \\
     \/   //               \\  \\
   xL <========================== x
   -  \\//                   \\ //-
   |  /\\                     \// |
   v \/ \\                    //v v
   yL ===\\==================//=> yT
      \\  \\                //  /\
       \\  \\              //  //
        \\  \/            \/  //
         \\   (xT <-| xTT)   //
          \\   -       -    //
           \\  |       |   //
            \/ v       v  //
              (yT <-| yTT)

The functor x:xT => xL acts on objects as L
and on morphisms as the transposition.

The functor yL => (yT <-| yTT) acts on objects as R
(plus the counit: O[yL] |-> (yL <-| yLRL) |-> (yLR <-| yLRLR))
and on morphisms as R.
```



E-mail to Seely (1)

First, consider a function "f:A->B":

```
  f: A --> B
     a |-> f(a)
```

Let's write it like this (we're putting our physicist's hats on):

```
        b:=f(a)
     a |-------> b
```

We're seeing it as relating the value of two variables - the output, b, is a function of the input; the rule "b:=f(a)" tells us how to calculate the output from the input.

Now let's think that A is a one-dimensional vector space, whose points are the possible values of a, and B is another one-dimensional vector space, whose points are the possible values for b. If our f is a linear function A->B (I will not write the arrow as a lollipop), then we can write b:=f(a) as

```
        b:=ka
     a |-----> b
```

where k is a constant.

Now let's make things worse. A becomes a two-dimensional vector space, X×Y, and B becomes a three-dimensional vector space, Z×W×T, where X is the space of the possible values for a variable x, and the same for the other letters: Y, Z, W, T. Now the value of A is a point in a two-dimensional space; we can think of it as being the pair of (numeric) values (x,y) - and the same for b: b=(z,w,t).

A linear transformation f:A->B sets the value of b from the value of a; or, in freshman's terms, it sets the value of (z,w,t) from the value of (x,y). It can be, for example, that f is like this:

```
         z:=1x+2y
         w:=3x+4y
         t:=5x+6y
  (x,y) |--------> (z,w,t)
      a |--------> b
```

Now let me pick the case where A and B are both one-dimensional again, but I will look at a polynomial function, b := 3a^2 + 4a, and I will represent it as a linear function from !A to B.

!A is an infinite-dimensional space; a point in !A is a certain quantity of a^0, plus a certain quantity of a^1, plus a certain quantity of a^2, etc - note that I'm sort of confusing variables and the base elements of vector spaces, that is not some random macarronic dialect of mathematics, it's the trick that makes everything work... anyway, ahead; writing the tensor product as "@", the basis of A is {a}, and the basis of !A is {a^0, a, a@a, a@a@a, a@a@a@a, ...}, where a^0 is the 0-th @-power of a; in another notation, this basis is {1, a, a^2, a^3, a^4, ...}.

E-mail to Seely (2)

```
The polynomial b := 3a^2 + 4a now becomes

                 b:=3a^2+4a
  (1,a,a^2,...) |----------> b
             !A -----------> B,

a linear function.

This is just the beginning, of course - but I've worked out all
diagrams until D.2 in page 8, and some diagrams and equations ahead of
that, and everything became very nice in this notation - only the
dotted lines seem to need some tricks that I don't have yet... Just
for curiosity, here are two diagrams (with no explanations here, but I
can provide them later):

      \           /            \           /
    A  \         / !A       da \         / !a'
        \       /                \       /
    =================        =================
    |       |       |        |       |       |
    |    !A |       |        |    !a | a^3 := 3a'^2da
    |    ___|___    |        |    ___|___    |
    |   |       |   |        |   |       |   |
    |   |   f   |   |        |   |   f   |   |
    |   |_______|   |        |   |_______|   |
    |       |       |        |       |       |
    |     B |       |        |     b := a^3  |
    |_______|_______|        |_______|_______|
            |                        |
          B |                b_{a}da := 3a'^2da
            |                        |

The differential combinator:

      \           /            \           /
    A  \         / !A       da \         / !a'
        \       /                \       /
     ===============          ===============
            |                        |
         !A |                    d!a | a^3 := 3a'^2da
     _______|_______          _______|_______
    |       |       |        |       |       |
    |    !A |       |        |    !a |       |
    |    ___|___    |        |    ___|___    |
    |   |       |   |        |   |       |   |
    |   |   f   |   |        |   |   f   |   |
    |   |_______|   |        |   |_______|   |
    |       |       |        |       |       |
    |     B |       |        |     b := a^3  |
    |_______|_______|        |_______|_______|
            |                        |
          B |                     db := 3a'^2da
            |                        |
```

E-mail to Seely (3)

```
In these examples I'm always pretending that A and B are
one-dimensional - everything works when they are more-dimensional, but
then I use some dictionary tricks to "read" the downcased diagrams as
something some general; formally, the downcased names for the wires
become just abstract names for things that are more complex - but
there is some precise relation between these abstract names (they're
not totally abstract or arbitrary) and the "real things" behind
them...
```

Diagrams (1)

[D.1] Constant maps:
(here a^0 := 0 da ⊗ a^0)

```
   \             /
  a \           / !a
     \         /
====================
|         |        |
|   (·0)  | a^0    |
|      ___|___     |
|    |         |   |
|    |    e    |   |
|    |_________|   |
|         |        |
|         | 1      |
|_________|________|
          |
          | 1
          |
```

```
       \                 /                  |                 |                          |           |
      a \               / a^7               |                 | a^7                      |           | a^7
          \            /                    |               __|__                     a  |         __|__
===================================      a  |              /      \                      |       /      \
|                |                |         |             |   Δ    |                      \     |   Δ     |
|          (·8)  | a^8            |         |              \______/                        \___  \______/
|              __|__              |         |              /      \                             \/      \
|            /       \            |           \          / a^2     \                      ______/\        \ a^4
|           |    Δ    |           |            \        /            |                   /        \        |
|            \_______/            |       ==================         | a^5         a^3 /      ==================
|            /       \            |       |        |       |         |                /       |        |       |
|      a^3  /         \ a^5       |       | (·3)   | a^3   |         |               /        | (·5)   | a^5   |
|    _____/_         _\_____      |       |     ___|___    |      ___|___         ___|___     |     ___|___    |
|   |         |     |         |   |       |  |         |   |    |         |     |         |   |   |         |  |
|   |    f    |     |    g    |   |  =    |  |    f    |   |    |    g    |  +  |    f    |   |   |    g    |  |
|   |_________|     |_________|   |       |  |_________|   |    |_________|     |_________|   |   |_________|  |
|           \         /           |       |       |        |       /                 |        |       /        |
|          b \       / c          |       |      b \       |      /                   \       |      / c       |
|             _\___/_             |       |_________\______|     / c                 b \      |___/____________|
|           |         |           |                   \         /                        \        /
|           |    @    |           |                    \       /                          \      /
|           |_________|           |                    _\_____/_                          _\_____/_
|                |                |                   |         |                        |         |
|                | b@c            |                   |    @    |                        |    @    |
|________________|________________|                   |_________|                        |_________|
                 |                                         |                                  |
                 | b@c                                     | b@c                              | b@c
                 |                                         |                                  |
```

Diagrams (2)

```
[D.3] Linear maps:

  \           /
 a \         / a^0           |       |
    \       /                |       | a^0
=================            |    ___|___
|       |       |            |   |       |
|  (·1) | a^1   |          a |   |  \ee  |
|    ___|___    |            |   |_______|
|   |       |   |            |       :
|   |  \ee  |   |            |       : 1
|   |_______|   |            |       :
|       |       |            | .....'
|       | a     |   =        0'
|    ___|___    |            |
|   |       |   |            | a
|   |   f   |   |         ___|___
|   |_______|   |        |       |
|       |       |        |   f   |
|       | b     |        |_______|
|_______|_______|            |
        |                    | b
        | b                  |
        |
```

Diagrams (3)

[D.4] Chain rule:

```
                                    |               |
                                    |               | a^11
                                    |             __|__
                                  a |            /     \
      \           /                 |           |   Δ   |
     a \         / a^11             |            \_____/
        \       /                   |            /     \
 =======================            |       a^3 /       \ a^8
|           |           |            \         /         \
|     (·12) | a^12      |        =================     ___\___________
|    _______|_______    |        |       |       |    |    \          |
|   |       |       |   |        |  (·4) | a^4   |    |     \ a^4     |
|   |       | a^4   |   |        |    ___|___    |    |    __\____    |
|   |    ___|___    |   |        |   |       |   |    |   |       |   |
|   |   |       |   |   |        |   |   f   |   |    |   |   f   |   |
|   |   |   f   |   |   |        |   |_______|   |    |   |_______|   |
|   |   |_______|   |   |        |        \      |    |       |       |
|   |       |       |   |        |       b \     |    |      _|_ b    |
|   |      _|_ b    |   |   =    |__________\____|    |_____/___\_____|
|   |_____/___\_____|   |                    \               /
|           |           |                   b \             / b^2
|           | b^3       |                      \           /
|        ___|___        |                    =================
|       |       |       |                    |       |       |
|       |   g   |       |                    |  (·3) | b^3   |
|       |_______|       |                    |    ___|___    |
|           |           |                    |   |       |   |
|           | c         |                    |   |   g   |   |
|___________|___________|                    |   |_______|   |
            |                                |       |       |
            |                                |       | c     |
                                             |_______|_______|
                                                     |
                                                     | c
                                                     |
```

Vector spaces

```
Fix a base field for our vector spaces: R, C, Q,
an arbitrary field K - it doesn't matter.
Let's denote by FinVec the category of finite-dimensional
vector spaces over that base field; morphisms are linear
transformations.
Each morphism in FinVec has a "dual", or "transpose",
going in the opposite direction.
A good notation (?) for morphisms:

       /1 2\
       |3 4|
       \5 6/
  R^2 -------> R^3            c := 1a + 2b
               /c\            d := 3a + 4b
  /a\          | |            e := 5a + 6b
  | | |------> |d|      a,b |--------------> c,d,e
  \b/          | |
               \e/

We should be able to do a lot using morphisms
that are not as general as these - matrices that
are all 0s, with just one entry being a 1.
An example:

        d:=a
  a,b |------> c,d,e

Then our previous morphism is:

  1(c:=a) + 2(c:=b) +
  3(d:=a) + 4(d:=b) +
  5(e:=a) + 6(e:=b)
```

The differential box

```
      \           /              \           /
    A  \         / !A            da \         / !a'
        \       /                    \       /
    =================            =================
    |       |       |            |       |       |
    |    !A |       |            |    !a | a^3 := 3a'^2da
    |    ___|___    |            |    ___|___    |
    |   |       |   |            |   |       |   |
    |   |   f   |   |            |   |   f   |   |
    |   |_______|   |            |   |_______|   |
    |       |       |            |       |       |
    |     B |       |            |     b := a^3  |
    |_______|_______|            |_______|_______|
            |                            |
          B |                    b_{a}da := 3a'^2da
            |                            |
```

The differential combinator

```
  \           /            \           /
A  \         / !A        da \         / !a'
    \       /                \       /
 ===============          ===============
        |                        |
     !A |                    d!a | a^3 := 3a'^2da
 _______|_______          _______|_______
|       |       |        |       |       |
|    !A |       |        |    !a |       |
|    ___|___    |        |    ___|___    |
|   |       |   |        |   |       |   |
|   |   f   |   |        |   |   f   |   |
|   |_______|   |        |   |_______|   |
|       |       |        |       |       |
|     B |       |        |     b := a^3  |
|_______|_______|        |_______|_______|
        |                        |
      B |                     db := 3a'^2da
        |                        |
```

Comonad structure

Two natural transformations, ε and δ,

```
      ε        δ
 X <----- !X -----> !!X
```

Obeying the duals of what would be this,
if I downcased my favourite monad:

```
 (ηT;μ)=(Tη;μ)=id
 (μT;μ)=(Tμ;μ)

   TX --ηT--> TTX --μ--> TX       x,a |---> x,a,1 |----> x,a1
   TX --Tη--> TTX --μ--> TX       x,a |---> x,1,a |----> x,1a
   TX ---------id------> TX       x,a |----------------> x,a

 TTTX --μT--> TTX --μ--> TX   x,a,b,c |---> x,a,bc |---> x,a(bc)
 TTTX --Tμ--> TTX --μ--> TX   x,a,b,c |---> x,ab,c |---> x,(ab)c
```

The duals are:

```
 (δ;ε!)=(δ;!ε)=id
 (δ;δ!)=(δ;!δ)

   !X <--ε!-- !!X <--δ-- !X
   !X <--!ε-- !!X <--δ-- !X
   !X <---------id------ !X

 !!!X <--δ!-- !!X <--δ-- !X
 !!!X <--!δ-- !!X <--δ-- !X
```

Coalgebra modalities

The comonad (!,δ,ε) is a _coalgebra modality_
if we are given, for every object X, maps:

```
       e
 ⊤ <----- !X -----> !X⊗!X
```

Obeying:

```
       -- !X --            !X --------> !X⊗!X
   id /    |   \ id          |             |
     /   Δ |    \          Δ |             | Δ⊗1
    v      v     v           v    1⊗Δ      v
 !X <---- !!X ---> !X      !X⊗!X -----> !X⊗!X⊗!X
      1⊗e     e⊗1
```

And this ("δ is a morphism of comonoids"):

```
          δ                         δ
     !X -----> !!X           !X ----------> !!X
        \       |              |              |
      e \       | e          Δ |              |
          \     v              v    δ⊗δ       v
           ---> ⊤            !X⊗!X ------> !!X⊗!!X
```

The term calculus

```
The new rule:

        a;b|-c  a|-b  a|-db
(Dt)    -------------------
            a|-c_{b}db

or, in long form:

        a:A,b:B|-c(a,b):C   a:A|-b(a):B   a:A|-db(a):B
        ----------------------------------------------
              a:A|-c_b(a,b(a))db(a):C

The equations ([S07], slide 20): the usual ones (?), plus these:

(Dt.1)   (c+c')_{b}db=c_{b}db+c'_{b}db
         0_{b}db=0

(Dt.2)   c_{b}(db+db')=c_{b}db+c_{b}db'

(Dt.3a)  b_{b}db=db

(Dt.3b)  c_{(b,b')}(db,0)=c_{b}db
         c_{(b,b')}(0,db')=c_{b'}db'

(Dt.4)   (c,c')_{b}db=(c_{b}db,c'_{b}db)

(Dt.5)   e_{b}db=e_{c}c_{b}db

(Dt.6)   (c_{b}db)_{db}db'=c_{b}db'
```

The term calculus: long version

```
Longer version, showing all the hypotheses:

          a,b|-c   a,b|-c'   a|-b   a|-db
(Dt.1)    --------------------------------
          a|-(c+c')_{b}db=c_{b}db+c'_{b}db

          [a,b|-0]   a|-b   a|-db
          -----------------------
              a|-0_{b}db=0

          a,b|-c   a|-b   a|-db   a|-db'
(Dt.2)    ---------------------------------
          a|-c_{b}(db+db')=c_{b}db+c_{b}db'

          [a,b|-b]   a|-b   a|-db
(Dt.3a)   -----------------------
             a|-b_{b}db=db

          a,b,b'|-c   a|-b   a|-b'   a|-db
(Dt.3b)   --------------------------------
             a|-c_{(b,b')}(db,0)=c_{b}db

          a,b,b'|-c   a|-b   a|-b'   a|-db
          --------------------------------
           a|-c_{(b,b')}(0,db')=c_{b'}db'

          a,b|-c   a,b|-c'   a|-b   a|-db
(Dt.4)    ----------------------------------
          a|-(c,c')_{b}db=(c_{b}db,c'_{b}db)

          a,b,c|-e   a,b|-c   a|-b   a|-db
(Dt.5)    ----------------------------------
             a|-e_{b}db=e_{c}c_{b}db

          a,b|-c   a|-b   a|-db   a|-db'
(Dt.6)    ------------------------------
           a|-(c_{b}db)_{db}db'=c_{b}db'
```

The D combinator

```
In the categorical setting the D works like this:

       dx,x       x
         -        -
         |D[f]    |f
         v        v
        dy        y
    (=y_{x}dx)

In the term calculus the derivative works on one of the variables, not
on all at once, so we need a trick to convert between the two:

          a
          -
          |<<⟦a|-db⟧,0>,<⟦a|-b⟧,1>>
          v
     (db,0),(b,a)            b,a
          -                   -
          |D[⟦a,b|-c⟧]        |⟦a,b|-c⟧
          v                   v
     c_{(b,a)}(db,0)          c
     =c_{b}db
     =dc

So:

    ⟦a|-c_{b}db⟧ := <<⟦a|-db⟧,0>,<⟦a|-b⟧,1>> D [⟦a,b|-c⟧]

[Note: [S07] uses "b" before "a"...]

In the other direction,

    D[⟦x|-y⟧] = dx,x|-y_{x}dx

    (dx,x),x'|-y    dx,x|-x    dx,x|-dx
    -----------------------------------
           dx,x|-y_{x}dx



         a,b|-c  a|-b  a|-db
(Dt)     -------------------
             a|-c_{b}db

And the other direction?
```

Universal and couniversal arrows

```
Units are universal:
                             /          a  \
                             |          -  |
  /           a \            |          |  |
  |           - |            |          |  |
  |           | | |-> (b -.> | a^L      |  |)
  |           v |            |  -   ♮   |  |
  \ a^L ==> a^LR/            |  |  |->  |  |
                             |  v       v  |
                             \  b ===> b^R /

Counits are couniversal:
                                / a^L <=== a  \
                                |  -       -  |
  /b^RL <== b^R \               |  |  <-|  |  |
  |  -          |               |  |   ♮   v  |
  |  |          | |-> (a^op -.> |  |      b^R |)
  |  v          |               |  |          |
  \  b          /               |  |          |
                                |  v          |
                                \  b          /

When a|->a^LR is a unit then the NT "b -.> diagram" is a
natural iso (i.e., for each B the arrow ♮ is a bijection
between hom-sets; this is the definition of "a|->a^LR is
universal").

When b^RL|->b is a counit the NT "a^op -.> diagram" is a
natural iso.
```