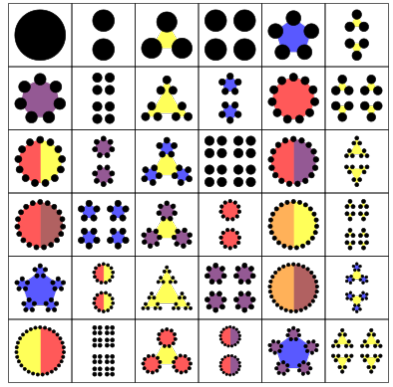
Diagrams: Declarative Vector Graphics in Haskell



Brent Yorgey

NY Haskell Users' Group
November 25, 2013

Part I: Demo!

Part II: Lessons for EDSL design

Take home

Domain analysis is hard!

Take home

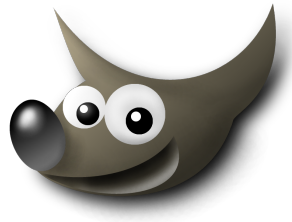
Domain analysis is hard!

Be in it for the long haul.

History

April 2008.

Wanted: declarative, programmatic drawing.



METAPOST

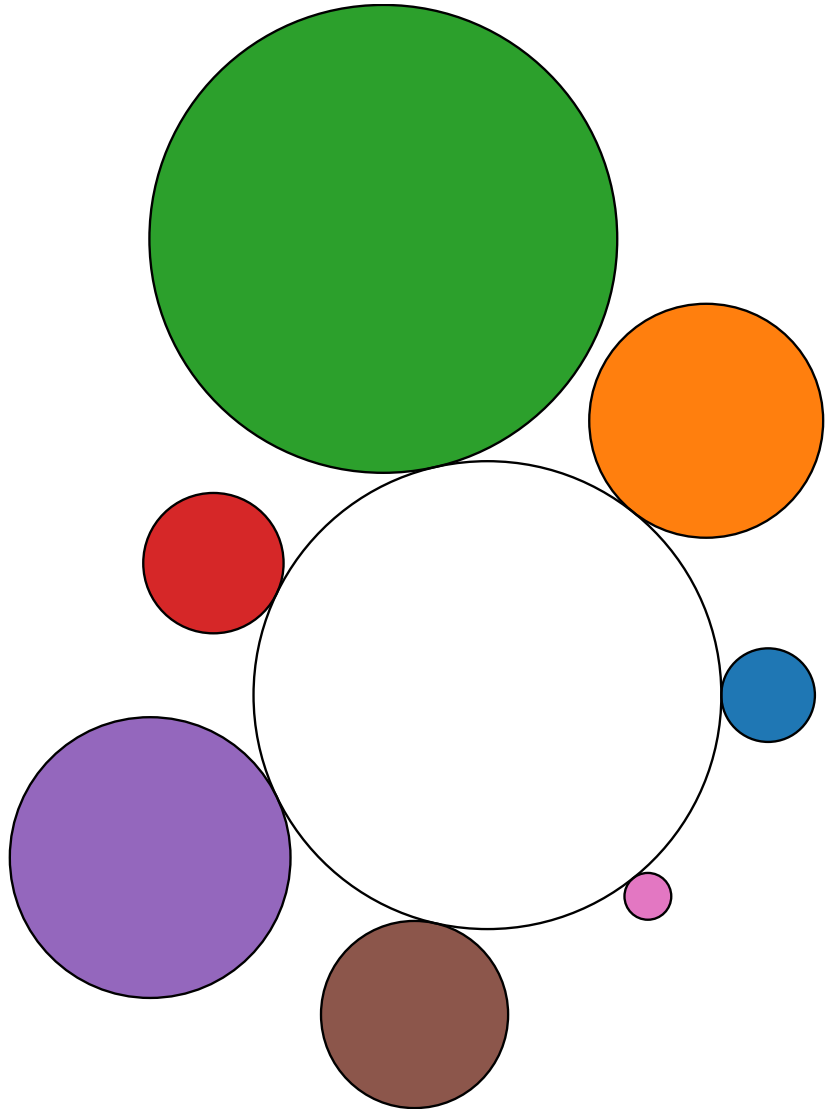
PGF/TikZ

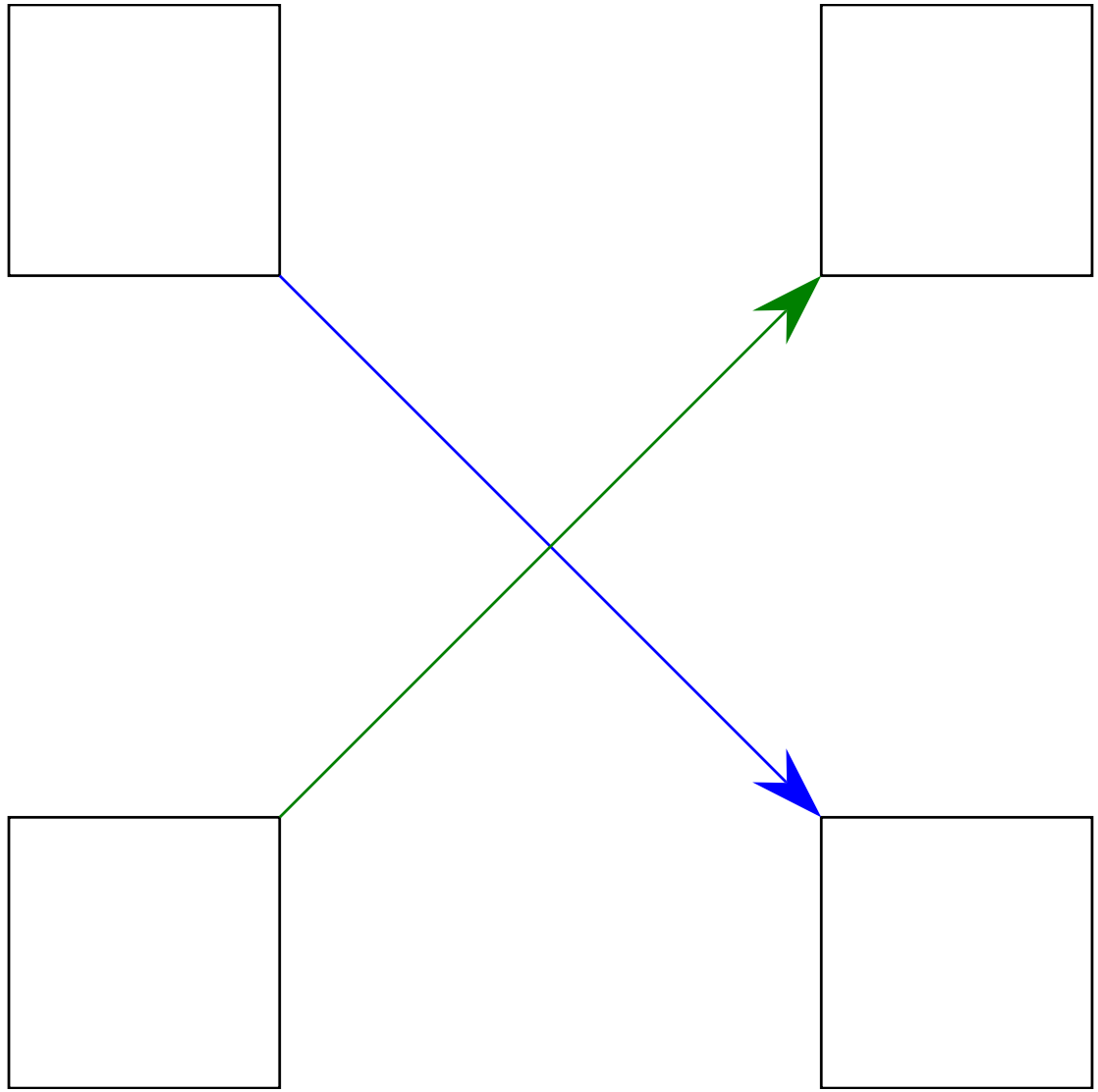
“How hard could it be?”

After two weeks of feverish hacking, diagrams was born!

After two weeks of feverish hacking, diagrams was born!

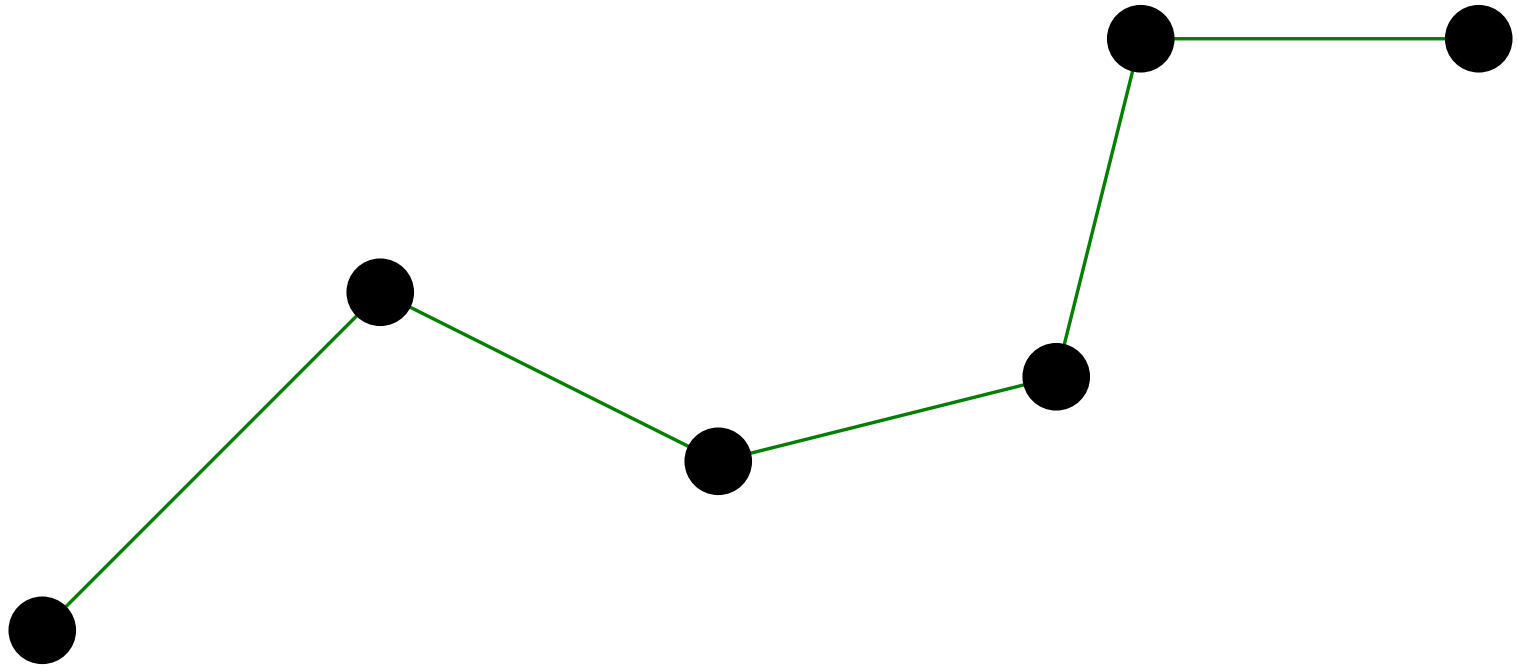
It sucked.





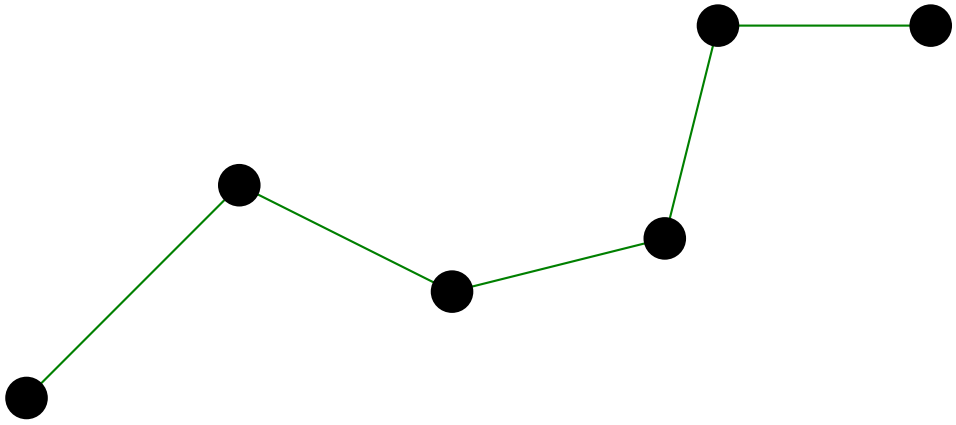
Paths

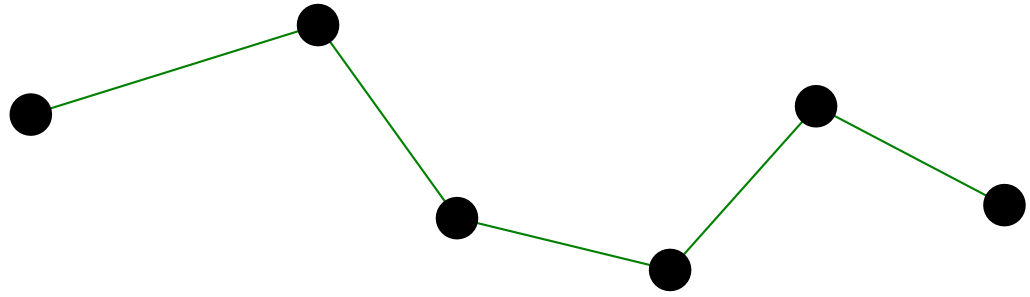
What is a **path**?



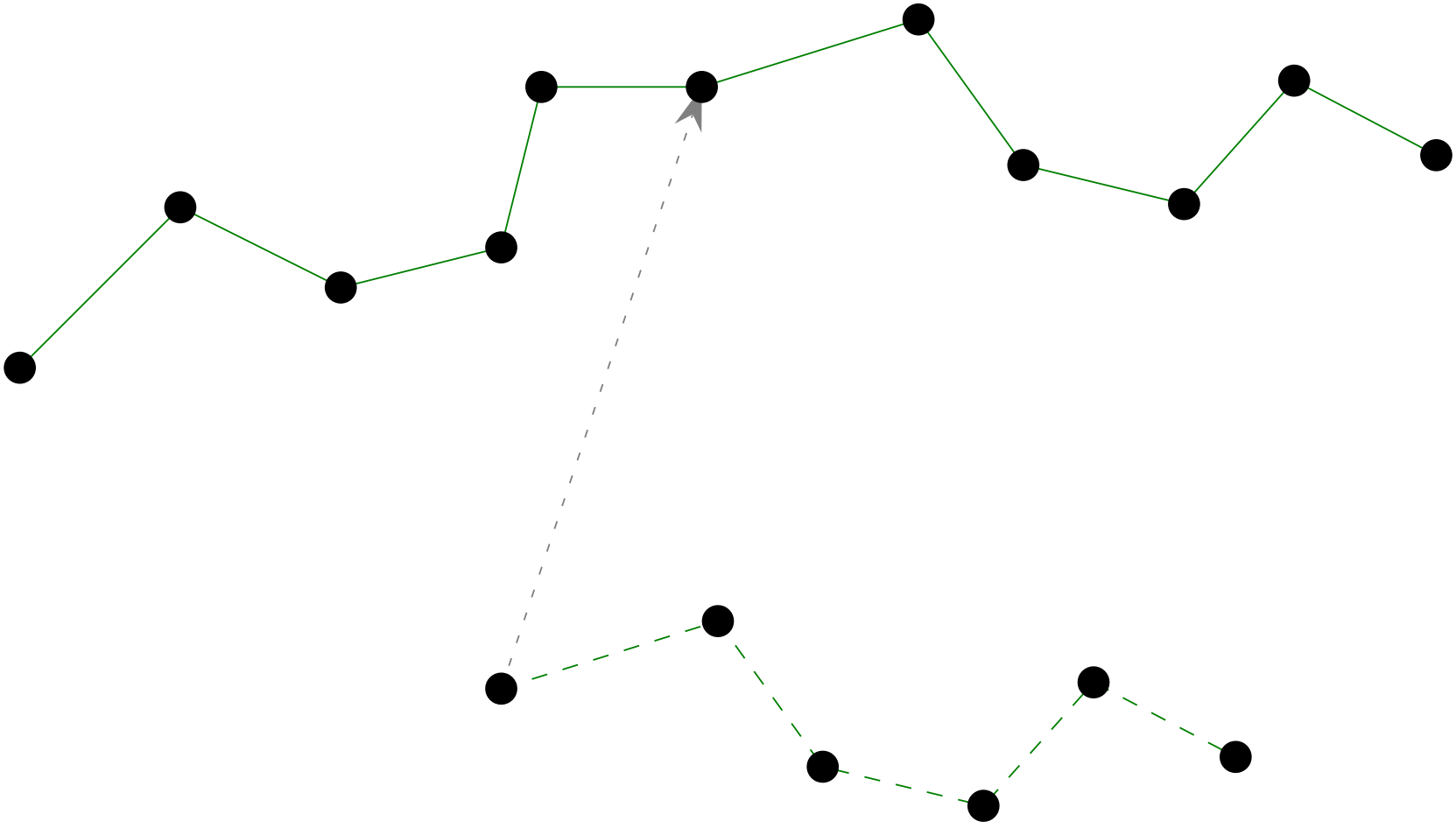
$$\textbf{type}\ \mathit{Path} = [\mathit{Point}]$$

Problem 1

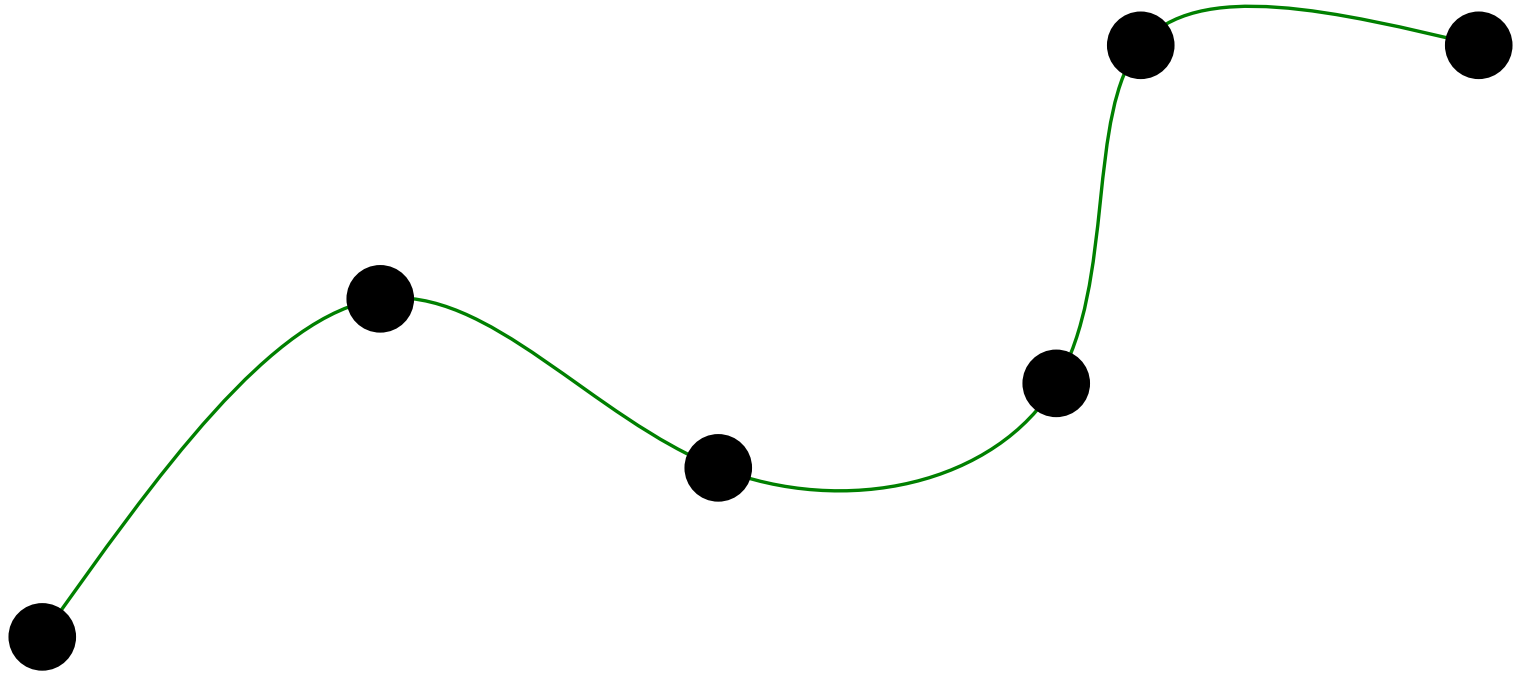




?



Problem 2



$$\textbf{type}\ \mathit{Path} = [(\mathit{P2}, \mathit{CurveSpec})]\ ?$$

Affine spaces

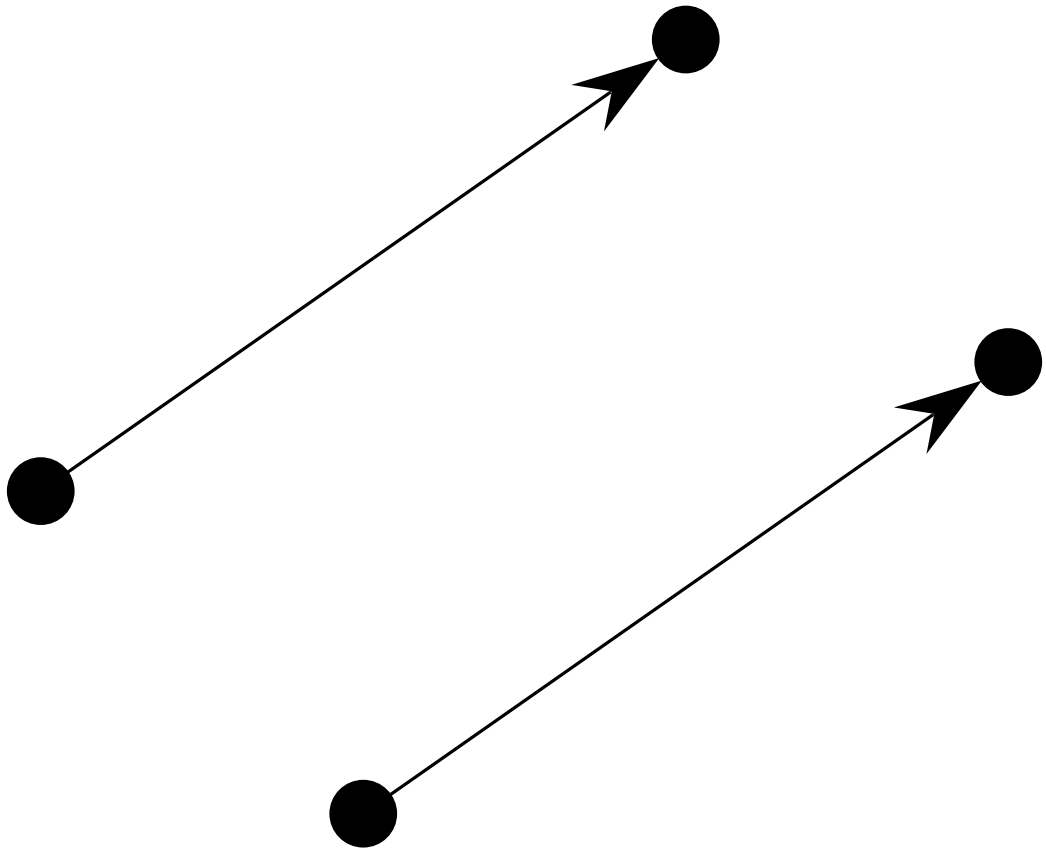
Find the bug

$$\begin{aligned}
&\textbf{type } \mathit{Point} \;\; = (\mathit{Double}, \mathit{Double}) \\
&\textbf{type } \mathit{Vector} = (\mathit{Double}, \mathit{Double}) \\
&\textbf{instance } (\mathrm{Num}\ a, \mathrm{Num}\ b) \Rightarrow \mathrm{Num}\ (a, b) \textbf{ where} \\
&\quad \ldots \\
&\mathit{parallelogram} :: \mathit{Point} \to \mathit{Point} \to \mathit{Point} \to \mathit{Point} \\
&\mathit{parallelogram}\ p_1\ p_2\ p_3 = p_1 - p_3 - p_2
\end{aligned}$$

Affine spaces for programmers

Confusing points and vectors is a type error!

Affine spaces



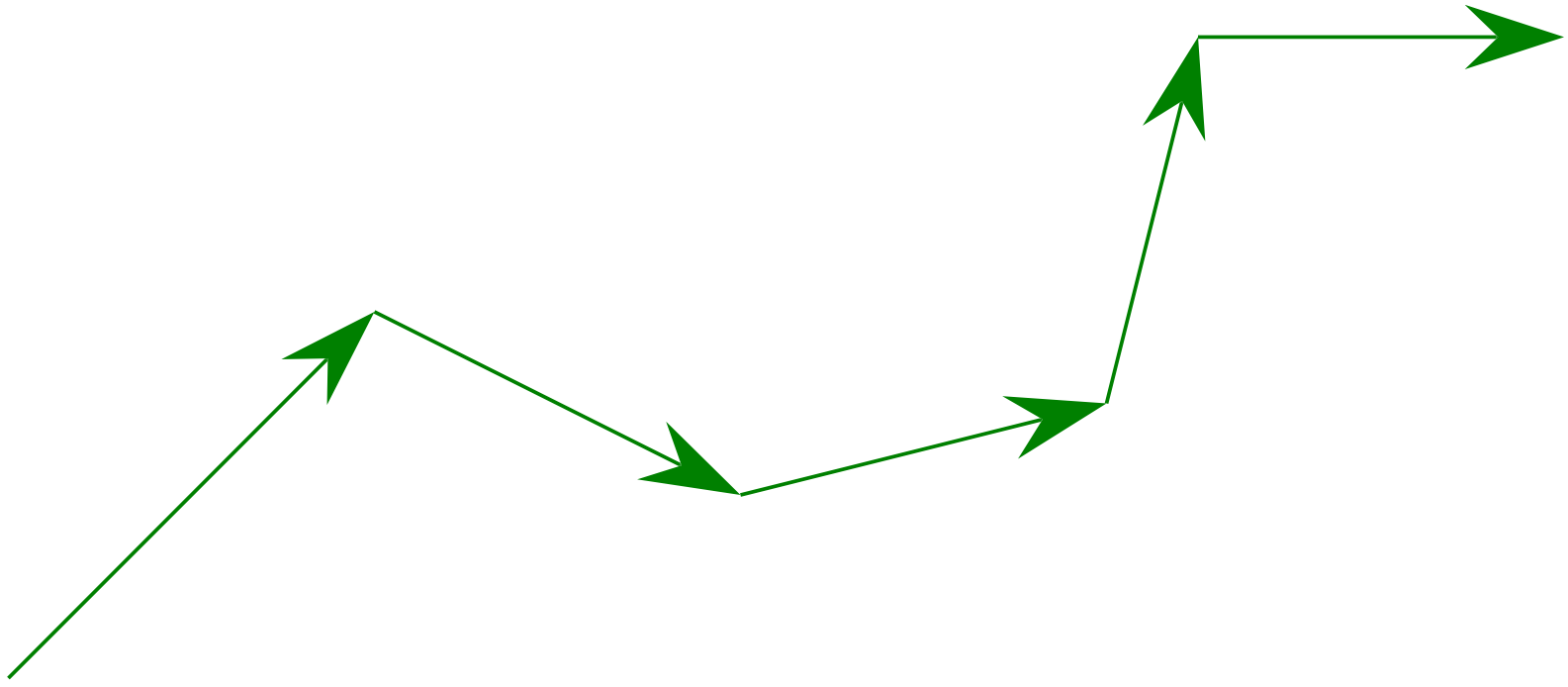
$$translate\ (p_1 - p_2) \equiv translate\ p_1 - translate\ p_2$$

$$translate\ (p_1 - p_2) \equiv translate\ p_1 - translate\ p_2$$

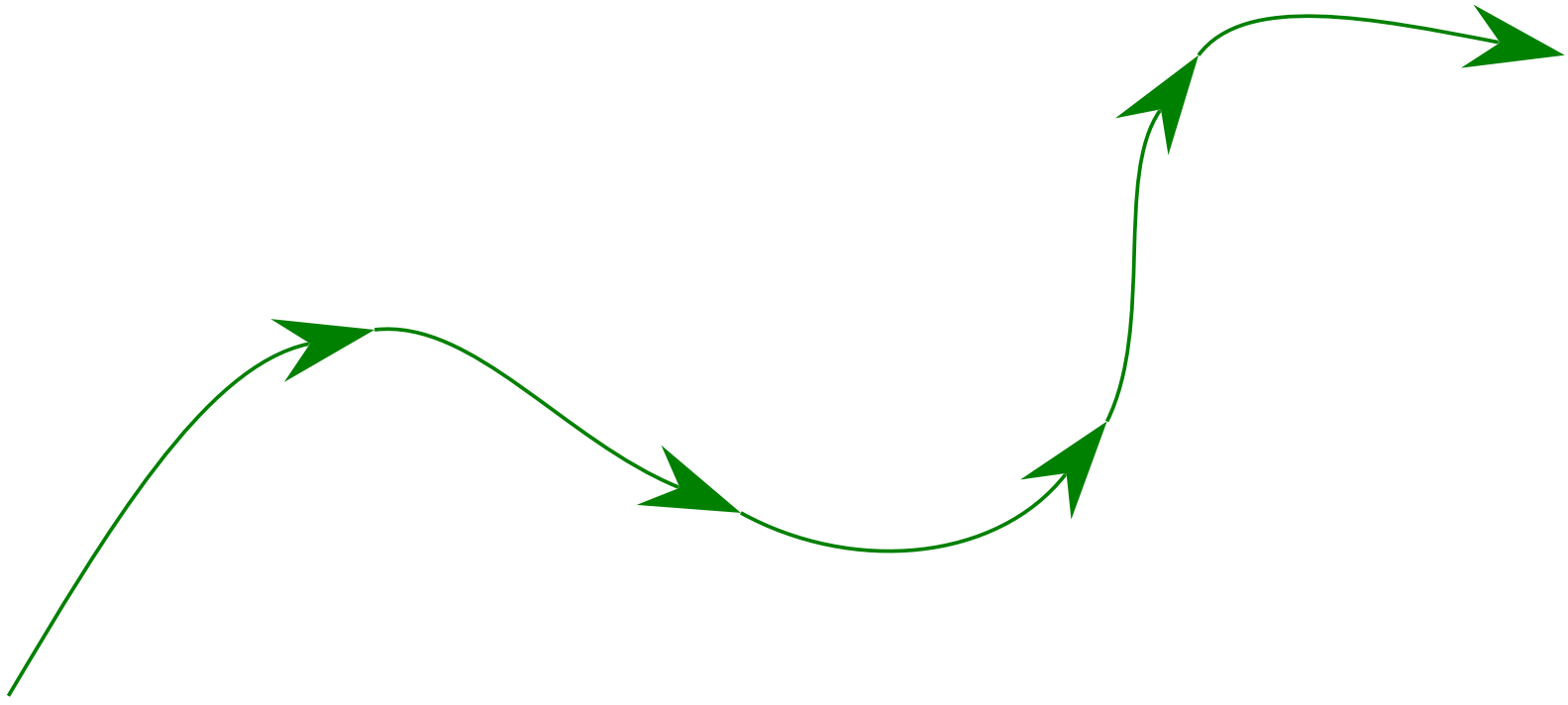
Translations apply to points but not to vectors!

$$
\begin{aligned}
(\hat{} + \hat{}) &:: \mathit{Vector} \to \mathit{Vector} \to \mathit{Vector} \\
(. + \hat{}) &:: \mathit{Point} \to \mathit{Vector} \to \mathit{Point} \\
(. - .) &:: \mathit{Point} \to \mathit{Point} \to \mathit{Vector}
\end{aligned}
$$

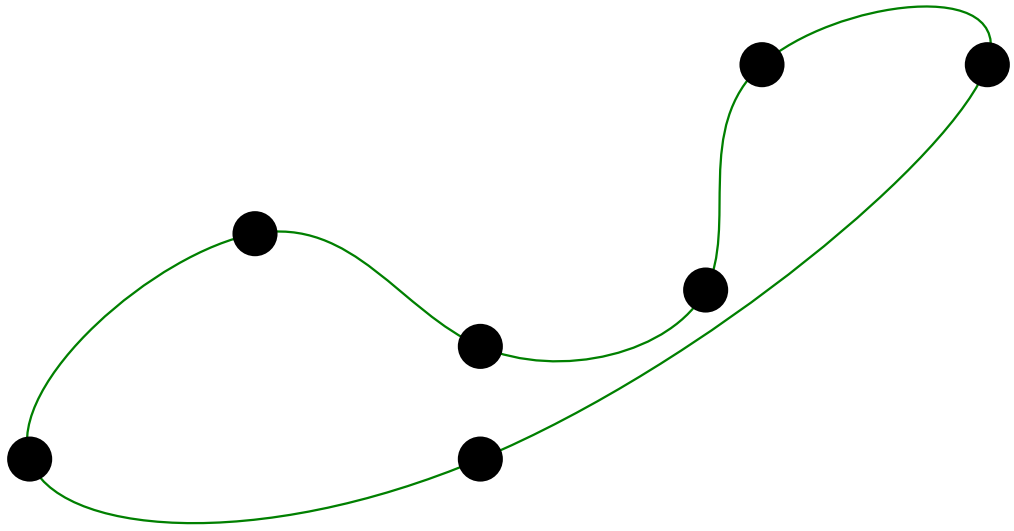
. . . Paths Again

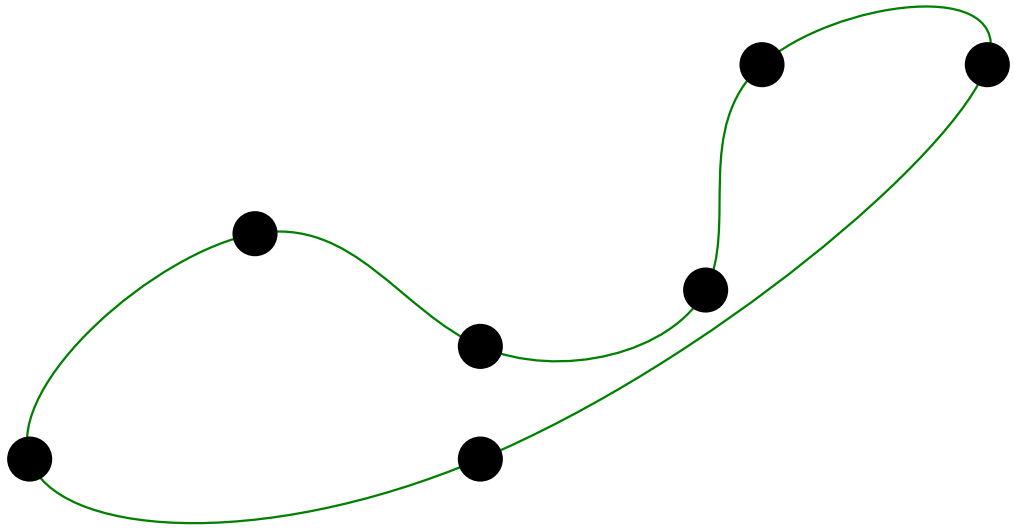


type $Path = [Vector]$

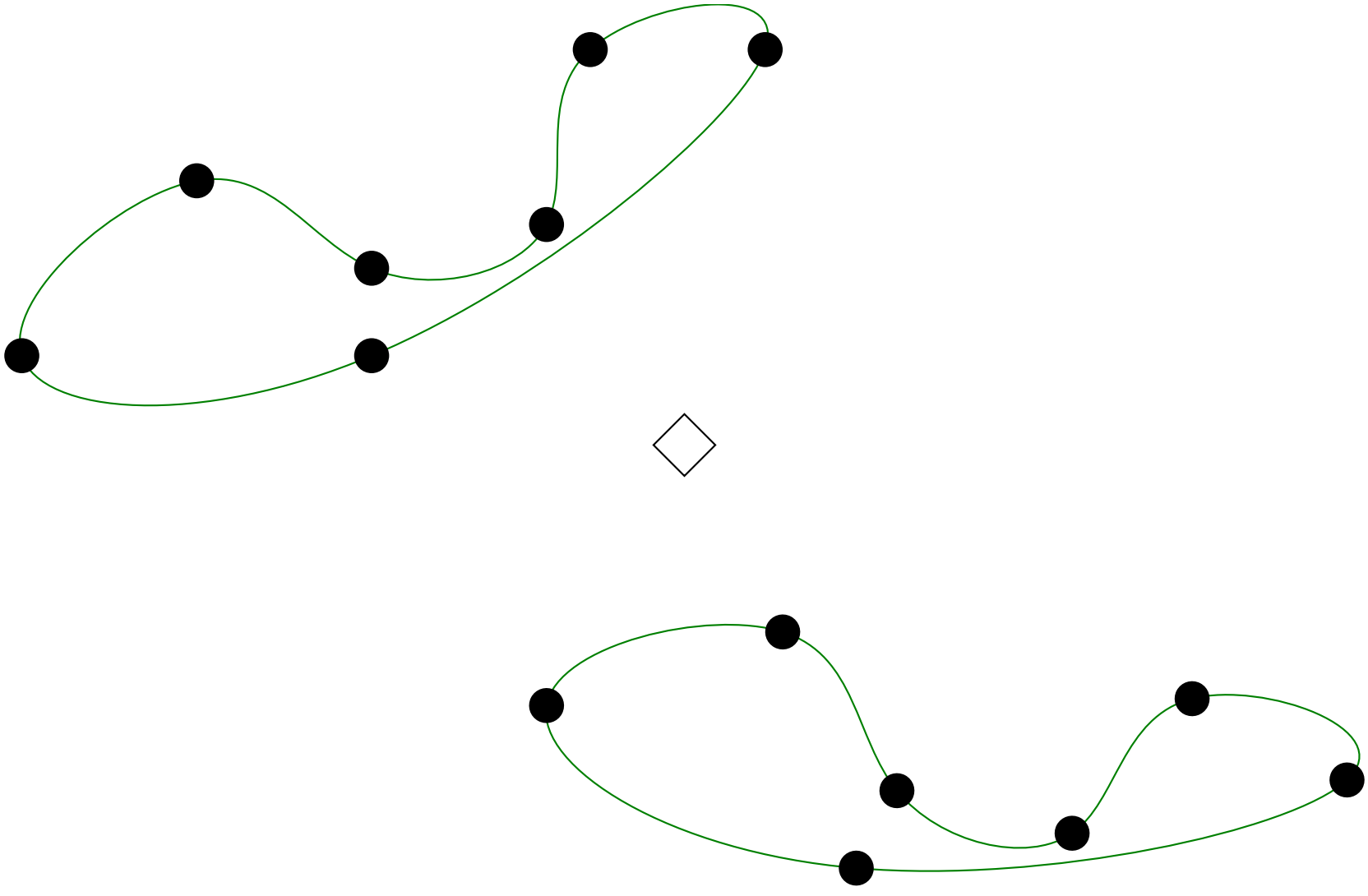


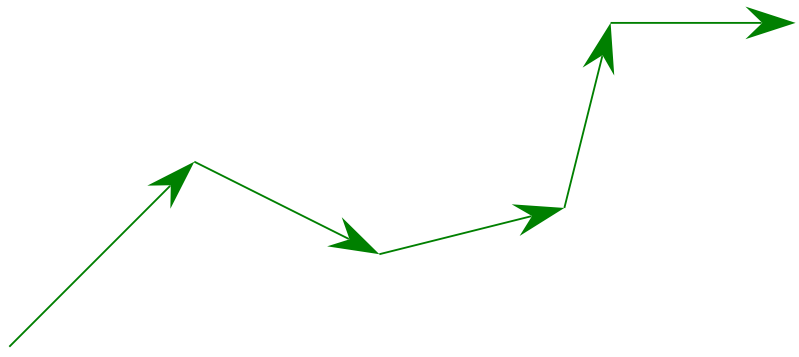
type $Path = [Segment]$

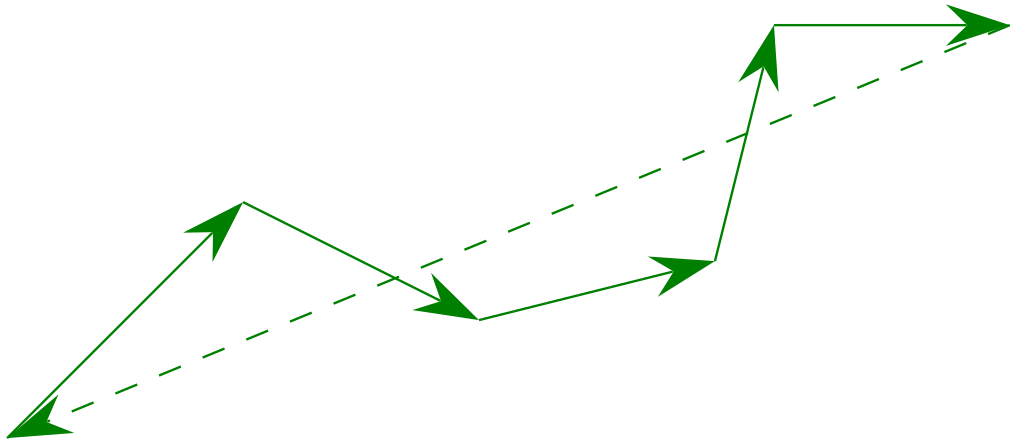


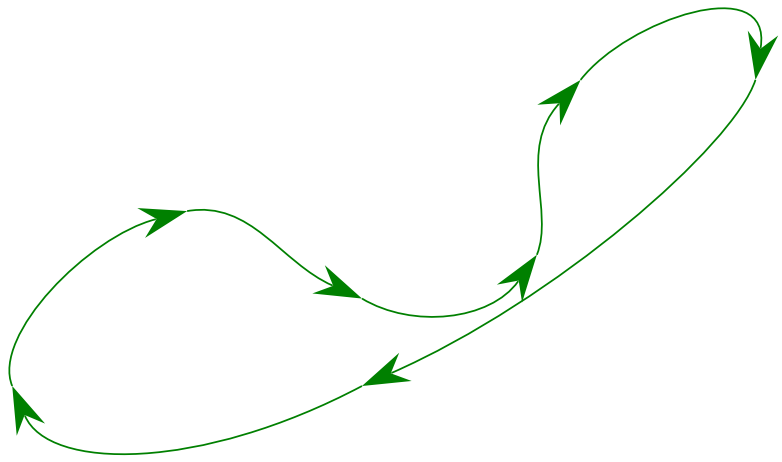


$$\textbf{type}\ \mathit{Path} = ([\mathit{Segment}], \mathsf{Bool})\ ?$$



( , *True*)?



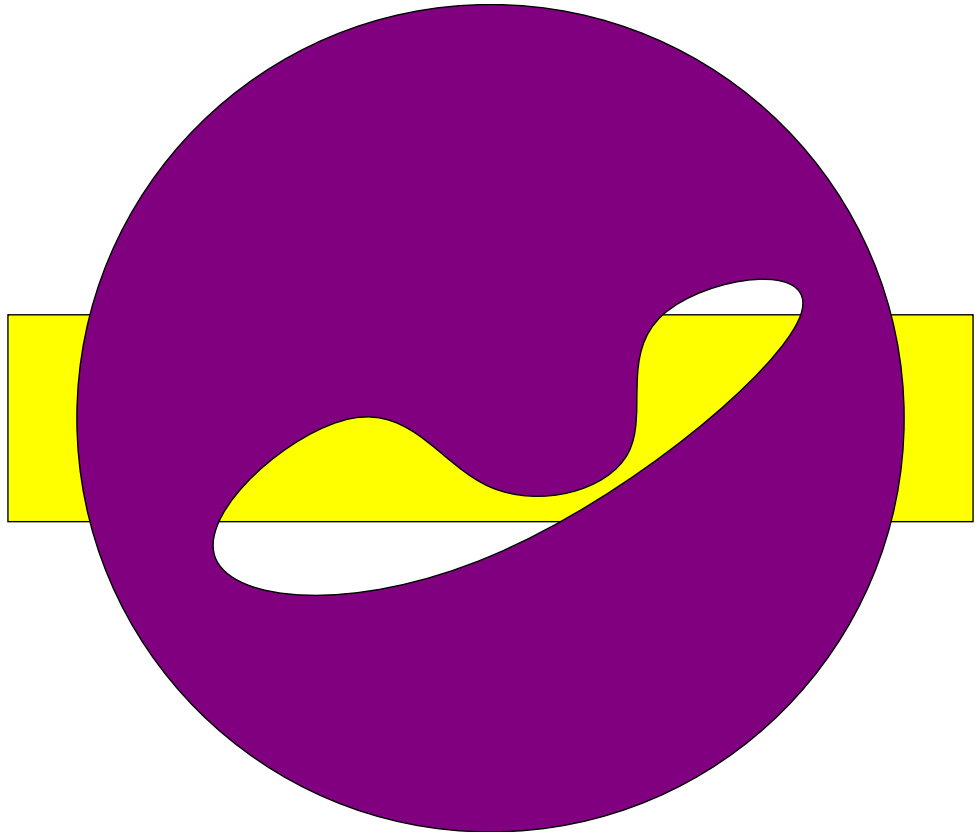
( , *True*)?

Our solution

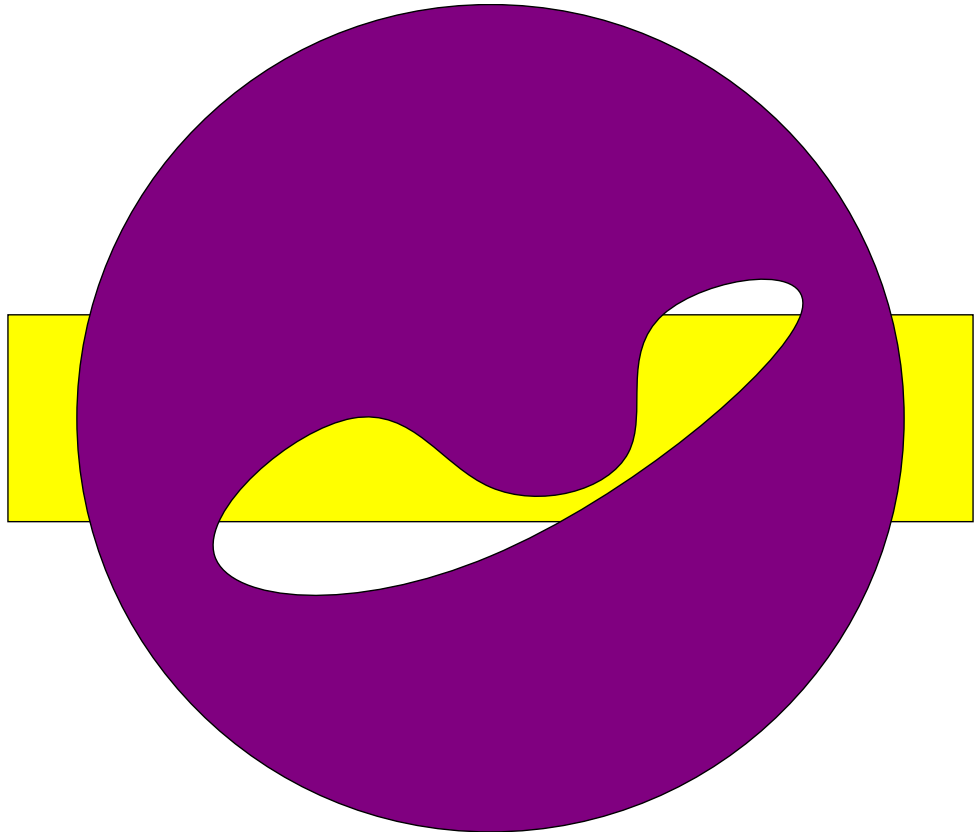
```
data Offset c v where
  OffsetOpen   :: Offset Open v
  OffsetClosed :: v → Offset Closed v

data Segment c v = Linear (Offset c v)
                 | Cubic v v (Offset c v)
```

```
data Trail' l v where
  Line  :: [Segment Closed v] → Trail' Line v
  Loop :: [Segment Closed v] → Segment Open v
                             → Trail' Loop v

glueLine  :: Trail' Line v → Trail' Loop v
closeLine :: Trail' Line v → Trail' Loop v

cutLine   :: Trail' Loop v → Trail' Line v
```

Problem 3



Problem 3



$$\textbf{type } \mathit{Trail} = [\mathit{Segment}] \ldots$$
$$\textbf{type } \mathit{Path} = [(\mathit{Point}, \mathit{Trail})]$$

Our solution

$$\textbf{data}\ \mathit{Located}\ a = \mathit{Loc}\ \{\mathit{loc} :: \mathit{Point}\ (V\ a), \mathit{unLoc} :: a\}$$

$$\textbf{newtype}\ \mathit{Path}\ v = \mathit{Path}\ [\mathit{Located}\ (\mathit{Trail}\ v)]$$