What is Penrose?

Designing Declarative Language Tutorials: a Guided and Individualized Approach
Anael Kuperwaj Cohen, Wode Ni, Joshua Sunshine

Defining Visual Mathematical Narratives Declaratively
Max Krieger, Wode Ni, Joshua Sunshine
What is Penrose?
Math notations are great!
But that’s not the whole story!
But that’s not the whole story picture!
Let’s make a diagram
Manipulation of low-level attributes

Meaning of the diagram is **lost**

*Let’s make a diagram (painfully)*
Introducing Penrose

**Domain**
Defines all possible notation in a domain to be visualized

- Set theory $\cap$
- Real Analysis $\mathbb{R}$
- Ray Tracing
- Linear Algebra $\nabla$
- Neural Networks

**Substance**
Declare objects and relationships with high-level notation

```
Set A, B
A \cap B
```

**Style**
Map the objects and relationships to a concrete visual *representation*

```
Set X { shape = Circle{} } 
X \cap Y {
    ensure X.shape contains Y.shape
}
```
Let’s make a diagram again
Buy 1 get \( n \) free
Diagramming with style(s)

```
1 Set A, B, C, D, E, F, G
2
3 IsSubset(B, A)
4 IsSubset(C, A)
5 IsSubset(D, B)
6 IsSubset(E, B)
7 IsSubset(F, C)
8 IsSubset(G, C)
9
10 NotIntersecting(E, D)
11 NotIntersecting(F, G)
12 NotIntersecting(B, C)
13
14 AutoLabel All
15```

So many languages!

Domain

Defines all possible notations to be visualized

Set $A, B$

$A \cap B$

Substance

Declare objects and relationships with high-level notation

Set $X \{ \text{shape} = \text{Circle} \}$

$X \cap Y \{ \\
\text{ensure} \ X.\text{shape} \text{ contains } Y.\text{shape} \\
\}$

Style

Map the objects and relationships to a concrete visual representation

Substance (end-users)

Domain/Style (experts)
**Domain**
Defines all possible notation in a domain to be visualized

Set theory ∩ Real Analysis

Ray Tracing ⊂ Linear Algebra ⊃

Neural Networks ⊂

---

**Substance**
Declare objects and relationships with high-level notation

Set $A, B$

$A \cap B$

**Style**
Map the objects and relationships to a concrete visual *representation*

```plaintext
Set $X \{ \text{shape} = \text{Circle}\} \}$

$X \cap Y \{$

  ensure $X$.shape contains $Y$.shape

$
```
**Domain**

Defines all possible notation in a domain to be visualized

- **set theory**
  \[ A \cap B \]

- **type** Set
- **predicate** IsSubset : Set s1 * Set s2
- **notation** "A ⊆ B" ~ "IsSubset(A, B)"

---

**Substance**

Declare objects and relationships with high-level notation

- \[ \text{Set } A, B \]
- \[ A \cap B \]

**Style**

Map the objects and relationships to a concrete visual *representation*

- \[ \text{Set } X \{ \text{shape} = \text{Circle}() \} \]
- \[ X \cap Y \{ \]
  
  - **ensure** X.shape contains Y.shape

---

**What’s in the Domain?**
Designing Declarative Language Tutorials: a Guided and Individualized Approach

Anael Kuperwajs Cohen, Wode Ni, Joshua Sunshine
So...what now?
Too many DSLs!

Set $A, B, C$

IsSubset($A, B$)

IsSubset($B, C$)

VectorSpace $U$

Vector $u_1 \in U$

Vector $u_2 \in U$

Scalar $c := \det(u_1, u_2)$

Real $a, b$

$A := [a, b] \subseteq R$

Real $l \in A$

$f : A \rightarrow R$

Continuous($f$)

Real $f_l$

$f_l := f(l)$

Real $d, e, i, j$

$I := (d, e) \subseteq R$

$J := [i, j] \subseteq R$

Real $u_1, u_2$

$u_1 \in \mathbb{R}^2$

$u_2 \in \mathbb{R}^2$

Scalar $c := \det(u_1, u_2)$

Too many DSLs!
Higher-order Functions

Scala allows the definition of higher-order functions. These are functions that take other functions as parameters, or whose result is a function. Here is a function `apply` which takes another function `f` and a value `v` and applies function `f` to `v`:

```scala
1. def apply(f: Int => String, v: Int) = f(v)
```

Note: methods are automatically coerced to functions if the context requires this.

Here is another example:

```scala
1. class Decorator(left: String, right: String) {
  2.   def layout(A: x: A) = left + x.toString() + right
  3. }
  4. 
  5. object FunTest extends App {
  6.   def apply(f: Int => String, v: Int) = f(v)
  7.   val decorort = new Decorator("", ")
  8.   println(apply(decoror.layout, 7))
  9. }
```

Building your first TypeScript file

In your editor, type the following JavaScript code in `greeter.ts`:

```javascript
function greeter(person) {
  return "Hello, " + person;
}
```

```javascript
let user = "Jane User";
```

```javascript
document.body.textContent = greeter(user);
```

Compiling your code

We used a `.ts` extension, but this code is just JavaScript. You could have copy/pasted this straight out of an existing JavaScript app.

At the command line, run the TypeScript compiler:

```sh
tsc greeter.ts
```

The result will be a file `greeter.js` which contains the same JavaScript that you fed in. We're up and running using TypeScript in our JavaScript app!

Now we can start taking advantage of some of the new tools TypeScript offers. Add a `: string` type annotation to the `person` function argument as shown here:

```javascript
function greeter(person: string) {
  return "Hello, " + person;
}
```

```javascript
let user = "Jane User";
```
Language tutorials are hard to follow!
GENTLE PUSH

Press and hold (Shift) while walking to gently push these civilians out of your way without making them drop what they are holding.

REACH THE MARKER BY USING THE GENTLE PUSH.

Wheatley: Okay, what you're doing there is jumping. You just... you just jumped. But nevermind. Say 'Apple'. 'Aaaapple.'

Video game tutorials?
Objective

Show two linearly dependent vectors

1. Autolabel All
2. VectorSpace U
3. In(A, U)
4. In(B, U)
5. Vector A
6. Vector B
7. Vector C
8. LinearlyDependent(A, B)
9. Scalar s
10. LinearlyDependent(B, C)

Add linear dependency, type "LinearlyDependent(A, B)"

A guided tour of Penrose
Recognition: Recognizing the associations between program outputs and source programs
Recall: Recalling language constructs of the DSL
Future Work
Future Work
Questions?
Defining Visual Mathematical Narratives Declaratively

Max Krieger, Wode Ni, Joshua Sunshine
Let’s prove

$$a^2 + b^2 = c^2$$

(Pythagorean Theorem)
A textual proof?
The proof is as follows:

1. Let ACB be a right-angled triangle with right angle CAB.
2. On each of the sides BC, AB, and CA, squares are drawn, CBDE, BAGF, and ACIH, in that order. The construction of squares requires the immediately preceding theorems in Euclid, and depends upon the parallel postulate.[9]
3. From A, draw a line parallel to BD and CE. It will perpendicularly intersect BC and DE at K and L, respectively.
4. Join CF and AD, to form the triangles BCF and BDA.
5. Angles CAB and BAG are both right angles; therefore C, A, and G are collinear. Similarly for B, A, and H.
6. Angles CBD and FBA are both right angles; therefore angle ABD equals angle FBC, since both are the sum of a right angle and angle ABC.
7. Since AB is equal to FB and BD is equal to BC, triangle ABD must be congruent to triangle FBC.
8. Since A-K-L is a straight line, parallel to BD, then rectangle BDLK has twice the area of triangle ABD because they share the base BD and have the same altitude BK, i.e., a line normal to their common base, connecting the parallel lines BD and AL. (lemma 2)
9. Since C is collinear with A and G, square BAGF must be twice in area to triangle FBC.
10. Therefore, rectangle BDLK must have the same area as square BAGF = AB².
11. Similarly, it can be shown that rectangle CKLE must have the same area as square ACIH = AC².
12. Adding these two results, AB² + AC² = BD x BK + KL x KC
13. Since BD = KL, BD x BK + KL x KC = BD(BK + KC) = BD x BC
14. Therefore, AB² + AC² = BC², since CBDE is a square.
A visual proof?

BUT...

WHERE DO I EVEN START?
A visual proof?

And that’s not the whole story or picture!
“QED, I GUESS”
~ Mr. Pythagoras

A visual narrative

Decomposability
Local Structure
Global Structure
Visual Narrative
Noun. A story told through pictures.
aka Comics, Manga, Cave Paintings...
Visual narratives are hard to make.
We’ve talked about creating diagrams declaratively with *Penrose*

Could we also make diagrammatic *narratives?*

*We haven’t implemented any of this yet. *This is just inquiry!*
What does the language look like?

Set \( A, B \)
\( A \cap B \)
**Conjunction**

Consider the set $A$ and the set $B$.

**Multiples**

Set of $B$ and $B$ is a subset of $C$.

**Predicates**

**Foci**

**Emanata**

**Nesting**
Conjunction

Multiples

Predicates

Foci

Emanata

Nesting
Consider the sets \([A] \) and \([B]\)
Fig 1.25, The Knot Book

NestPanels \( (p_1, [(\text{thm1}, t_1), (\text{thm2}, t_2), (\text{lemma1}, t_3)]) \)

Proof (Fig 1.26)

WUT? II Planar isotopy HUH? III HOW? I

Nesting
These lines are both radius of the same circle and so equal.
These lines are both radius of the other circle and so equal too.
So we have a △ triangle with all lines equal to one another. Boom.

SuppletePanel ({ IsSubset(A,B) }, [A,B], "[A] is a subset of [B]"

Eli Parra's Elements

Circle C,D
{ Focus(C) Panel ([C,D]) }
What emerges?

Parallel Counterfactuals

Future work: sampled layout, interactivity

Gutter as implication

What if?
Thank you!

Acknowledgements

Co-Authors Nimo Ni and Joshua Sunshine

References via Katherine Ye, Neil Cohn, Eli Parra

QUESTIONS?