

**A Domain Specific Language for**

# **Name Modifiers**

**Favonia**

**Build Another**

# **Proof Assistant**

Build Yet<sup>2</sup> Another

# Proof Assistant

RedPRL redtt

Build Yet<sup>3</sup> Another

# Proof Assistant

**RedPRL** **redtt** **cooltt**

Build Yet<sup>4</sup> Another

# Proof Assistant

**RedPRL** **redtt** **cooltt** **algaett**

Build Yet<sup>4</sup> Another

# Proof Assistant

RedPRL redtt cooltt algaett

Goal: Build Yet<sup>100</sup> Another

# Core / NbE

Syntax Highlighting

Linters/Formatters

Structured Editing

Metaprogramming

Error Messages

Records=Modules

Library Management

Separate/Incremental Compilation

LSP

Unfolding Control

Unicode

Universe Levels

Inductive Types

Parsing

Typing Information

Namespaces

Pretty Printing

**Core / NbE**

Syntax Highlighting

Linters/Formatters

Structured Editing

Metaprogramming

Error Messages

Records=Modules

Library Management

Separate/Incremental Compilation

LSP

Unfolding Control

Unicode

Universe Levels

Inductive Types

Parsing

Typing Information

**Namespaces**

Pretty Printing



# SIX PRINCIPLES

**Expressiveness**

**Explicit Sequencing**

**Implicit Namespaces**

**Typo Prevention**

**Small Kernel**

**Algebraic Effects**

# **Expressiveness**

**Anything You Can Do, I Can Do Better**

open M

using (a)

renaming (a to b)

hiding (b)

open M

**Modifiers are effectful**

hiding (b)

open M

**Modifiers are effectful**  
**Explicit Sequencing**

```
module List =  
  struct  
    open List  
    let new_fun = ...  
  end
```

```
let List.new_fun = ...
```

```
sig
  type t
  val x : t
end
```

**Abstraction (+)**  
**Inflexible (-)**  
**Often Impossible (-)**

An expressive language of signatures [Ramsey et al]



**Modules are Typed**  
**Namespaces are NOT**

let **M.a** = ...      **Groups of Names**  
let **M.a.x** = ...  
let **M.a.y** = ...      **Sharing a Prefix**

# Implicit Namespaces

# Prevent Typos

**Detect Empty Namespaces**

**Make Unions Explicit**

# A Small Kernel

**empty-check**   **scoping**   **renaming**  
**sequencing**   **union**   **hook (extension)**

# Algebraic

# Effects

# SIX PRINCIPLES

**Expressiveness**

**Explicit Sequencing**

**Implicit Namespaces**

**Typo Prevention**

**Small Kernel**

**Algebraic Effects**

# **Racket's DSL**

## **Lexical Scoping**

### **Unused Imports/Defs**

# **Write Yourself A Proof Assistant**

**<https://github.com/RedPRL/yuujinchou>**