A Domain Specific Language for

Name Modifiers

Favonia

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
  using (a)
  renaming (a to b)
  hiding (b)
```

Modifiers are effectful

```
open M
using (a)
renaming (a to b)
```

Modifiers are effectful

Explicit Sequencing

```
module List =
struct
   include 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 = ...
let M.a.x = ...
let M.a.y = ...
```

Groups of Names Sharing a Prefix

Implicit Namespaces

Prevent Typos

Detect Empty Namespaces

Make Unions Explicit

A Small Kernel

empty-check
sequencing

scoping
union

renaming
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