Library system in Stak Scheme

@raviqqe

February 18, 2024

Contents

Stak Scheme

- A bytecode compiler and virtual machine (VM) for Scheme
- The compiler is written in Scheme.
- The VM is written in Rust.
- It aims to support R7RS-small.

Library system in R7RS

Defining a library

- Libraries export symbols.
- Libraries import symbols from other libraries.
- Libraries are "called" but only once.

```
(define-library (foo)
  (export foo)

  (import (scheme base))

  (begin
    (define (foo x)
      (write-u8 x))))
```

Library system in R7RS

Importing a library at a top level

```scheme
(import (foo))

(foo 65)
```

Where to put libraries?

- Where to put libraries?
    - Inlining library clauses (e.g. Gauche)
    - Libraries as files (e.g. Chibi Scheme)
- Stak Scheme took the inlining solution.

```scheme
(define-library (foo)
  (export foo)

  (import (scheme base))

  (begin
    (define (foo x)
      (write-u8 x))))

(import (foo))

(foo 65)
```

Implementation in a compiler

Pipelines

1. Read source.
2. Expand libraries. <- **new!**
    - Read all `(define-library ...)` clauses.
    - Expand all `(import ...)` clauses.
3. Expand macros.
4. Compile expressions.
5. Encode objects.
6. Write bytecodes.

Library expansion

- Environments of libraries are separated by symbol prefixes.
  - e.g. `foo` -> `$42$foo` where `42` is the ID of a library
- Importing symbols from a library converts all symbols' prefixes.
- Top-level symbols do not have any prefix.

Future work

- Library system
  - `(rename ...)`
  - `(prefix ...)`
- `eval` procedure

Summary

- Building a library system is fun!