`eval` in Stak Scheme

@raviqqe

May 12, 2024

Contents

- Stak Scheme
- `eval` in R7RS
- Implementation in Stak Scheme
- Future work

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.

Progress

- **The `eval` procedure**
  - Only procedures available
  - No macro support yet
- The `stak-profile` command
  - Traces and profiles Stak Scheme codes.

`eval` in R7RS

- The `eval` procedure evaluates an S-expression.
- Only global bindings in a given environment are accessible.

```
(eval <expr-or-def> <environment>)
```

Example

```
(import (scheme base) (scheme eval))

(eval
  '(display "Hello, world!")
  (environment '(scheme write)))
```

Environments in R7RS

- `(environment <specifier> ...)`
  - Imports immutable environments of specifiers.
  - Normal libraries (e.g. `(scheme base)` and `(scheme write)`) can be used for the specifiers.
- `(interactive-environment)`
  - A mutable environment for REPL

Implementation in Stak Scheme

- The compiler injects library and macro information built in a compiler into target codes.
  - `($$libraries)` and `($$macros)` primitives
- Keeps portability of the compiler.
  - The other Scheme implementation can be used to run the compiler.
- Duplicates codes related to compilation and macro expansion in a `(scheme eval)` library.

Demo

Future work

- Macros in `eval`
- Deduplication of codes between a compiler and the `(scheme eval)` library

Summary

- Building `eval` is fun!