The `include` syntax in Stak Scheme

@raviqqe

August 17, 2025

Contents

- Stak Scheme
- Progress
    - The `include` syntax
    - Other R7RS compatibility improvements
- 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 the R7RS-small standard.
- Forked from Ribbit Scheme

References

- GitHub
- Website

Progress

- (Almost) full R7RS-small compatibility
  - The `include` syntax
- The website improvements

The (almost) full R7RS-small compatibility

- Stak Scheme is (almost) fully compatible with R7RS-small.
    - Other than the `include` and `include-ci` syntaxes...
    - Stak Scheme does not implement the full numeric tower either.

The `include` syntax

- The `include` syntax embeds S-expressions read from a specified path into source code at the location.
  - The `include-ci` syntax is its case-insensitive variant.
- Its argument must be **a string literal**.
- The `include` syntax is in the `(scheme base)` library.

```scheme
(include "./foo.scm") ; -> (write-string "Hello, world!")
```

The current implementation in Stak Scheme

- The `include` syntax is the top-level "meta" syntax.
    - Similar to `define-library` and `import` library syntaxes.
- It is not part of the `(scheme base)` library.
- The syntax is processed by the compiler completely statically.
    - You cannot even call it with any arguments of the other macros whose denotations are string literals.

Other compatibility improvements

- `define-syntax` syntax in `define-syntax` syntax
- The `write`, `write-simple`, and `write-shared` procedures
- Partial implementation of SRFI 1

Other progress

- Rewriting Aruba in Go.
  - https://github.com/raviqqe/aruba-go
- Monza editor, the `textarea` element with syntax highlight
- Using a proper terminal library in REPL on the website

Future work

- Unicode support in the `(scheme char)` library
- `close` primitive optimization
- Closure space safety
- Stack trace

Summary

- Building Scheme is fun! 🥳