Progress in Stak Scheme

October 13, 2024

Contents

- Startup time optimization
- Tree shaking
- `(scheme time)` library

Startup time optimization

- Now, the Scheme interpreter written by Stak Scheme itself starts in ~200 ms on macOS!
- It's basically REPL without the print part.
  - It doesn't support `define-library`.
- Libraries and macros bundled in the interpreter was huge.
  - They are automatically compiled into the resulting bytecodes.
- Optimizing their data structures helped a lot.

Tree shaking

- `(import (shake (scheme base)))` imports only symbols used in the codes below.
- Not in R7RS
- It's technically the same as `(import (only (scheme base) ...))` enumerating all used symbols.
- Definitions are not removed like JavaScript/TypeScript's tree shaking.

```
(import (shake (scheme base)))

(write-string "Hello, world!")
```

`(scheme time)` library

- Time procedures
  - `current-jiffy`
  - `current-second`
  - `jiffies-per-second`
- Implemented by Rust's `std` or `libc`

Triplet rib

- Previously, Stak Scheme had a quartet data structure of `(type car cdr tag)`.
  - This is internally two 64-bit words.
  - `type` is a tag in a `car` pointer.
  - `tag` is a tag in a `cdr` pointer.
- But it is `(car cdr tag)`.
- `type` and `tag` are merged into one.
- This should make the new bytecode encoding easier...

Others

- Adopting `core::error`

Future work

- Developing the new bytecode encoding takes time.
- Hopefully, it's done on the next meetup.
- We'll see... 😃