H2get — a simple H2 client

Frederik Deweerdt

Overview

- Written in C
- Small
- Extensible
- Integrates with mruby

Overview

- One H2 object for the connection
- send_* functions to send Frames
- send_header to send a header frame
- read() to receive a frame

Overview

- src/h2get.c
  - entry point
  - connection establishment
  - sending frames
- src/h2get_read.c
  - Receiving frames
- src/h2get_mruby.c
  - calls functions in h2get.c or h2get_read.c
- src/hpack.c
  - hpack implementation

ruby

```
h2g = H2.new
h2g.connect("www.fastly.com")
h2g.send_prefix()
# Ack settings
while true do
  f = h2g.read(-1)
  puts f.to_s
  if f.type == "SETTINGS" and (f.flags & 1 == 1) then
    next
  elsif f.type == "SETTINGS" then
    h2g.send_settings_ack()
    break
  end
end
```

ruby

```
prio_low = H2Priority.new(0, 0, 16)
prio_high = H2Priority.new(0, 0, 32)
req = {
    ":method" => "GET",
    ":authority" => "www.fastly.com",
    ":scheme" => "https",
}
req1 = req.merge(":path" => "/?1")
req2 = req.merge(":path" => "/?2")
h2g.send_headers(req1, 15, PRIORITY | END_STREAM, prio_low)
h2g.send_continuation({}, 15, END_HEADERS)
h2g.send_headers(req2, 17, END_STREAM | END_HEADERS, prio_high)
```

ruby

```
while open_streams
    f = h2g.read(-1)
    puts "type:#{f.type}, stream_id:#{f.stream_id}, len:#{f.len},
flags:#{f.flags}"
    if f.type == "GOAWAY" then
        puts f.to_s
    elsif f.type == "PING" then
        f.ack()
    elsif f.type == "DATA" and f.len > 0 then
        h2g.send_window_update(0, f.len)
        h2g.send_window_update(f.stream_id, f.len)
    elsif f.type == "HEADERS" then
        puts f.to_s
    end

    if f.type == "DATA" or f.type == "HEADERS" then
        if f.is_end_stream
            # stop tracking stream
        end
    end
end
```