```ruby
require 'irb/ruby-lex'
require 'stringio'
#
class MimickIRB < RubyLex
  attr_accessor :started

  class Continue < StandardError; end
  class Empty < StandardError; end

  def initialize
    super
    set_input(StringIO.new)
  end

  def run(str)
    obj = nil
    @io << str
    @io.rewind
    if l = lex
      case l.strip
      when 'reset'
        @line = ''
      when 'time'
        @line = "puts %{You started \#{IRBalike.started.since} ago.}"
      else
        @line << l << "\n"
        raise Continue if @ltype || @continue || @indent > 0
      end
    else
      raise Empty if @line == ''
    end
    obj = eval @line, TOPLEVEL_BINDING, '(irb)', @line_no unless @line.empty?
    @line_no += @line.scan(/\n/).length
    @line = ''
    @exp_line_no = @line_no

    @indent = 0
    @indent_stack = []

    $stdout.rewind
    output = $stdout.read
    $stdout.truncate(0)
    $stdout.rewind
    [output, obj]
```

```
  rescue Object => e
    case e when Empty, Continue
           else @line = ''
    end
    raise e
  ensure
    set_input(StringIO.new)
  end
end

CURSOR = '>>'.freeze
IRBalike = MimickIRB.new
$stdout = StringIO.new

require 'green_shoes'
Shoes.app do
  @str = [CURSOR + ' ']
  @cmd = ''
  stack width: 1.0, height: 1.0 do
    background '#555'
    stack width: 1.0, height: 50 do
      para 'Interactive Ruby ready.', fill: white, stroke: red
    end
    @scroll =
      stack width: 1.0, height: -50, scroll: true do
        background '#555'
        @console = para @str, font: 'Monospace 14px', stroke: '#0f0'
        @console.cursor = -1
      end
  end
  keypress do |k|
    case k
    when "\n"
      begin
        out, obj = IRBalike.run(@cmd + ';')
        @str += ["#{@cmd}\n",
                 span("#{out}=> #{obj.inspect}\n", stroke: '#fda'),
                 "#{CURSOR} "]
        @cmd = ''
      rescue MimickIRB::Empty
      rescue MimickIRB::Continue
        @str += ["#{@cmd}\n.. "]
        @cmd = ''
      rescue Object => e
        @str += ["#{@cmd}\n", span("#{e.class}: #{e.message}\n", stroke: '#fcf'),
```

```
                  "#{CURSOR} "]
        @cmd = ''
      end
    when String
      @cmd += k
    when :backspace
      @cmd.slice!(-1)
    when :tab
      @cmd += '  '
    when :alt_q
      quit
    when :alt_c
      self.clipboard = @cmd
    when :alt_v
      @cmd += clipboard
    end
    @console.replace(*(@str + [@cmd]))
    @scroll.scroll_top = @scroll.scroll_max
  end
end
```