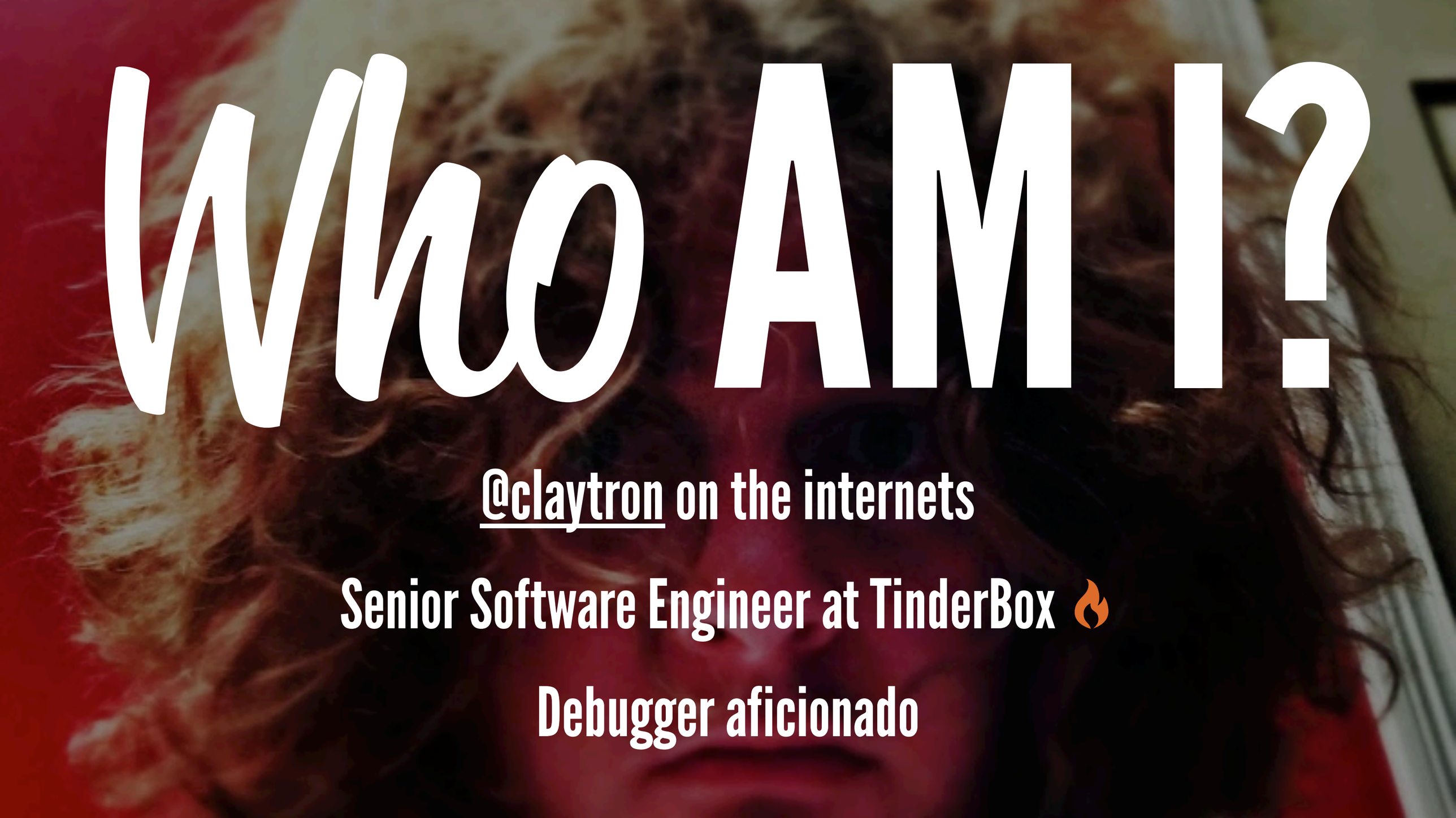
EXPLORING CODE WITH *Pry*!

@indy.rb 02.2016


Who AM I?
@claytron on the internets
Senior Software Engineer at TinderBox 🔥
Debugger aficionado

Pry is a *powerful* alternative to the standard IRB shell for Ruby.
It features syntax highlighting, a flexible *plugin* architecture,
runtime invocation and *source* and documentation *browsing*

– pryrepl.org

GETTING *Started*

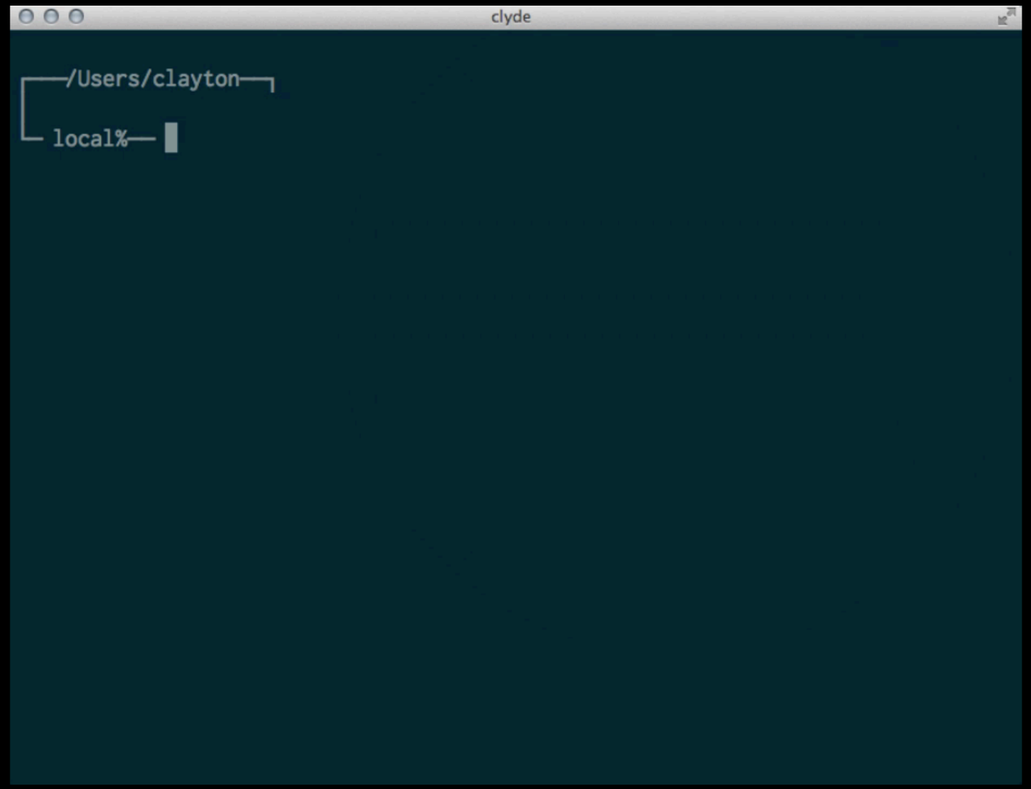
```
$ gem install pry pry-doc
$ pry
[1] pry(main)> puts 'hello'
hello
=> nil
[2] pry(main)>
```

ANATOMY OF THE *Prompt*

```
[1] pry(main)>
 |    |    |
 |    | Current Context
 |    |
 |   Prompt Name
 |
Command history
```

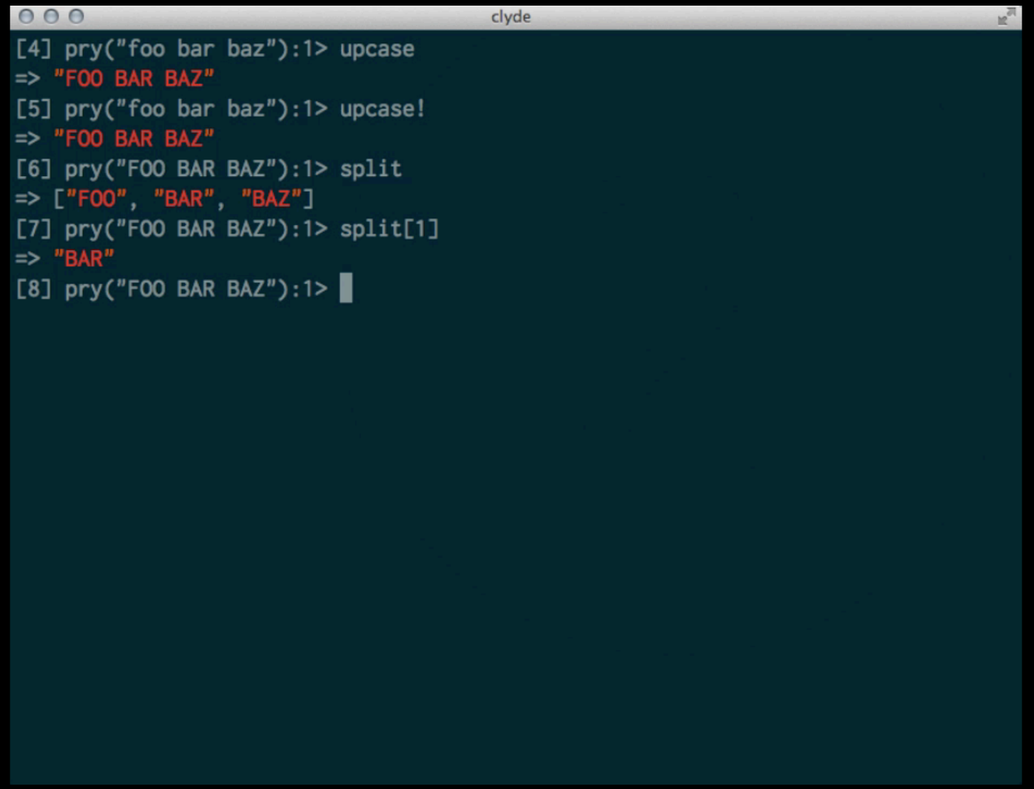

AND NOW FOR MY FIRST TRICK...

Context Switching



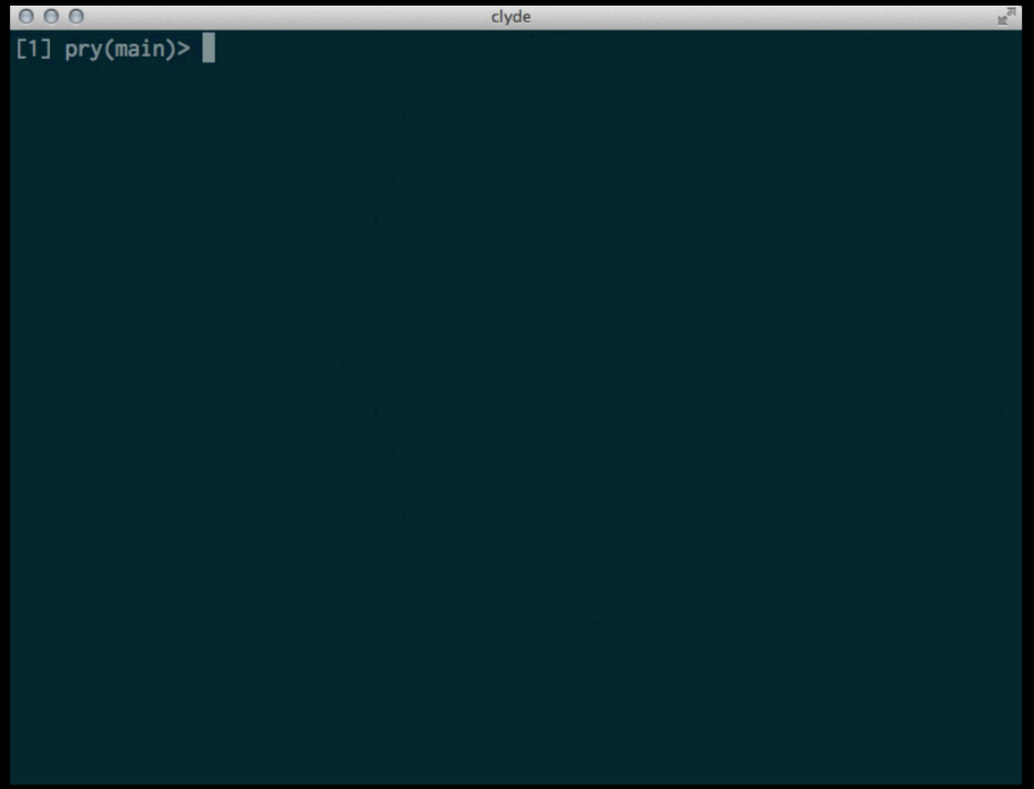
⭐ Watch the Video ⭐

In Memory Editing



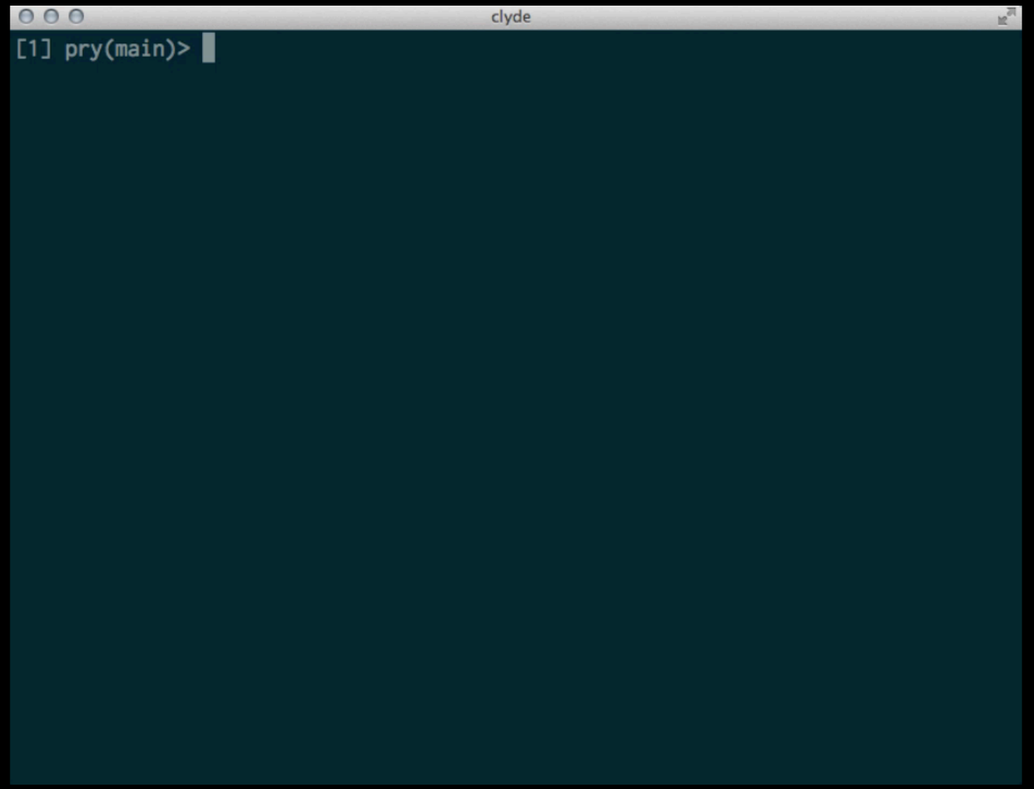
⭐ Watch the Video ⭐

Source Browsing / Editing



⭐ Watch the Video ⭐

Line Editing



⭐ Watch the Video ⭐


HALP!

GETTING *Help*

```
[1] pry(main)> help
Help
  help               Show a list of commands or information about a specific command.

Context
  cd                 Move into a new context (object or scope).
  ls                 Show the list of vars and methods in the current scope.
  wtf?               Show the backtrace of the most recent exception.

Editing
  edit               Invoke the default editor on a file.

Introspection
  show-doc           Show the documentation for a method or class.
  show-source        Show the source for a method or class.

Input and output
  .<shell command>   All text following a '.' is forwarded to the shell.
  cat                Show code from a file, pry's input buffer, or the last exception.
```

GETTING *Help*

```
[1] pry(main)> help wtf?
Usage: wtf[?|!]

Show's a few lines of the backtrace of the most recent exception (also available
as `_ex_.backtrace`). If you want to see more lines, add more question marks or
exclamation marks.

wtf?
wtf?!???!?!?
```


CUSTOMIZATION

```
vim ~/.pryrc
```

REPEAT LAST COMMAND

```ruby
# Hit Enter to repeat last command
Pry::Commands.command(/^$/, 'repeat last command') do
  last_command = Pry.history.to_a.last
  unless ['c', 'continue', 'q', 'exit-program', 'quit'].include? last_command
    _pry_.run_command last_command
  end
end
```

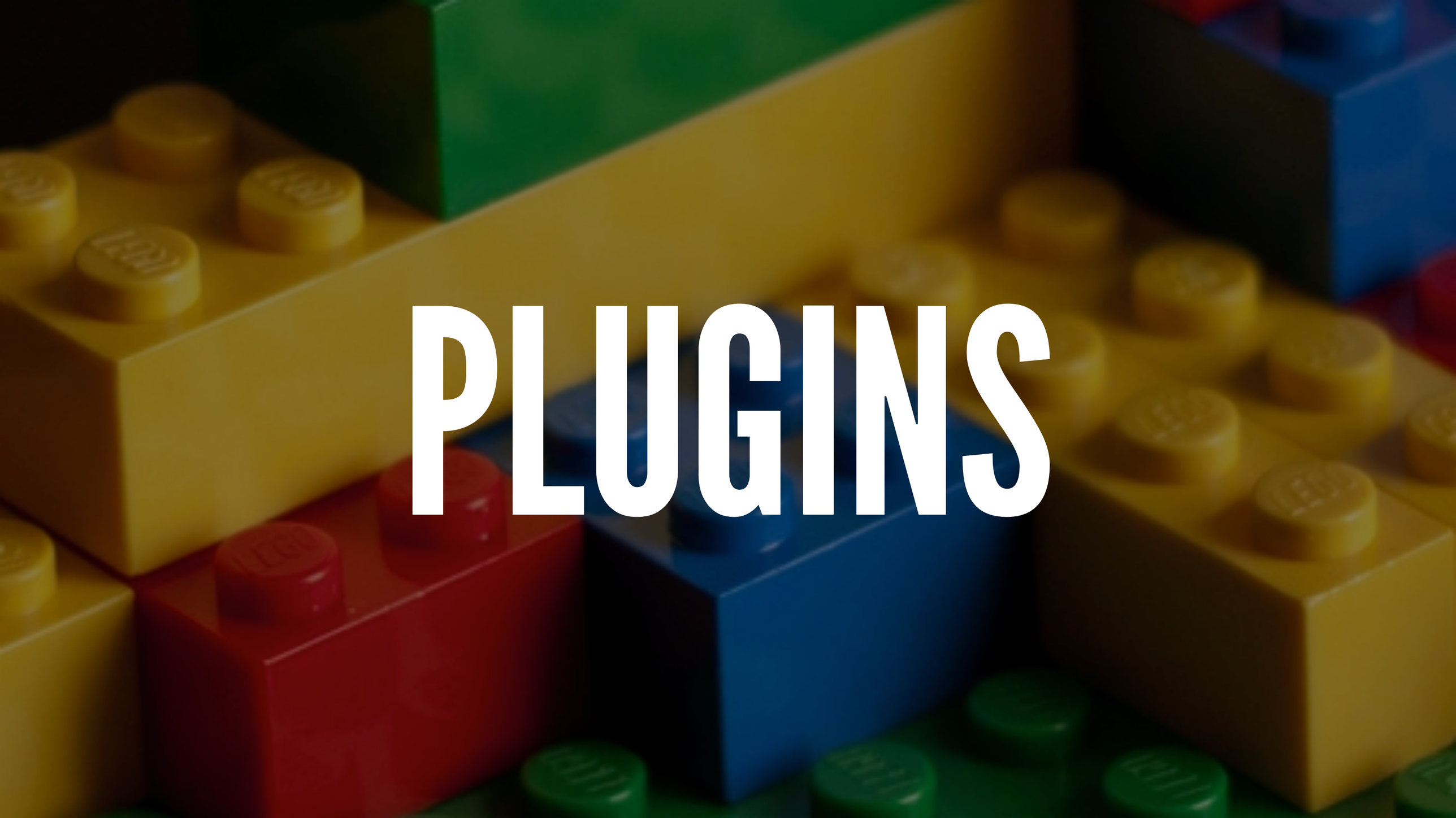
ALIASES

```
# Single letter and pdb like aliases
if defined?(PryByebug)
  Pry.commands.alias_command 'c', 'continue'
  Pry.commands.alias_command 's', 'step'
  Pry.commands.alias_command 'n', 'next'
  Pry.commands.alias_command 'return', 'finish'
  Pry.commands.alias_command 'r', 'finish'
  Pry.commands.alias_command 'f', 'finish'
  Pry.commands.alias_command 'q', 'exit-program'
  Pry.commands.alias_command 'quit', 'exit-program'
end
```

OTHER STUFF

```
# Turn off the automatic pager
Pry.config.pager = false

# Make the prompt silly
# [1] oh no!(main)>
Pry.config.prompt_name = 'oh no!'
```


PLUGINS

PRY-AWESOME_PRINT

Automatically `ap` return value

Use semi-colon to silence return value

```
[1] pry(main)> x = {foo: :bar, baz: :bang}
=> {:foo=>:bar, :baz=>:bang}
[2] pry(main)> x = {foo: :bar, baz: :bang};
[3] pry(main)>
```

PRY-BYEBUG

Make Pry act like an actual debugger

Adds `next`, `step`, `continue`, etc.

PRY-STACK_EXPLORER

Show the call stack

Go up and down the call stack

PRY-CLIPBOARD

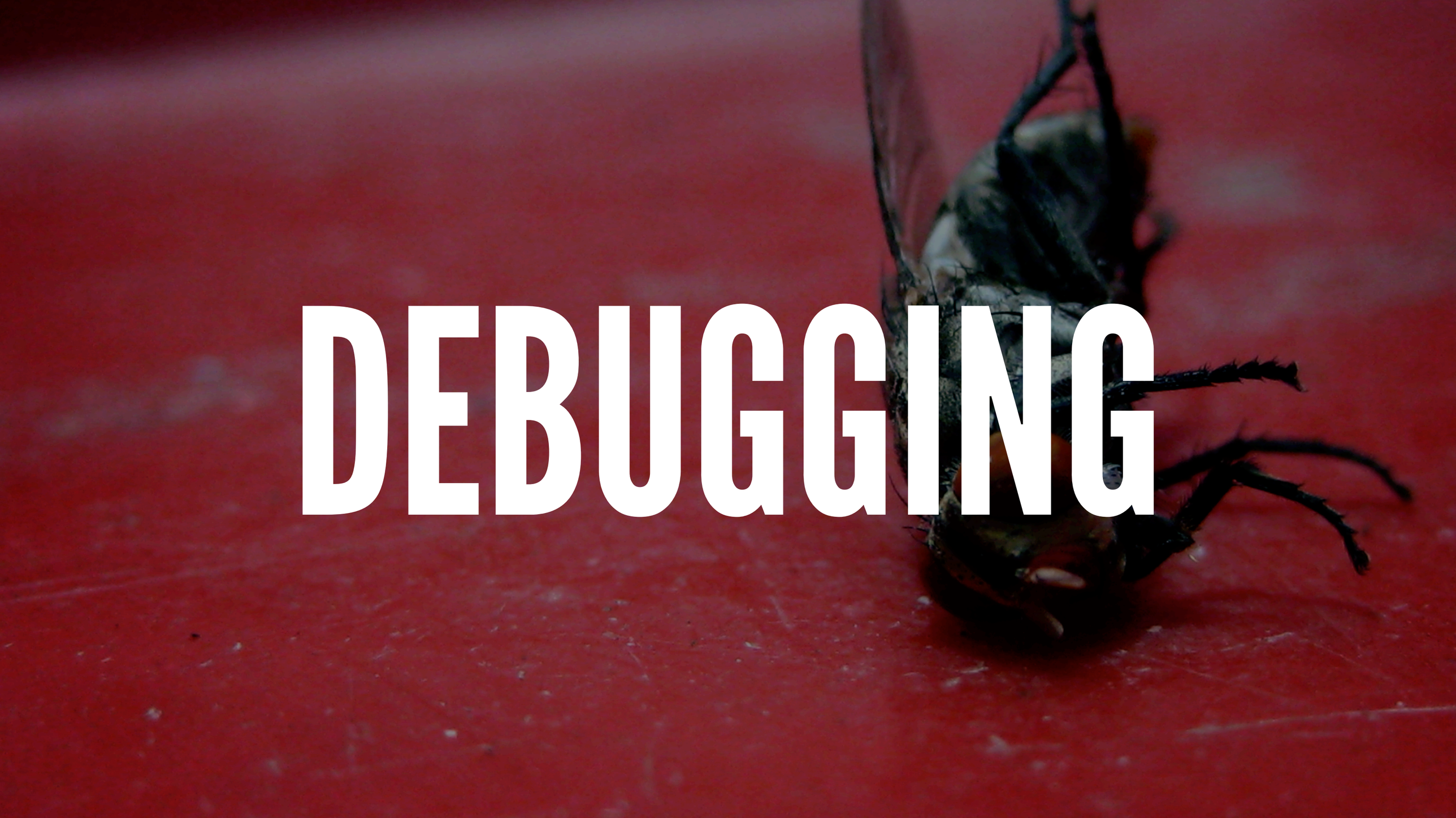
Quickly copy history or output to the clipboard

Works cross platform via the `clipboard` gem

PRY-RAILS

Automatically use Pry instead of IRB for `rails console`

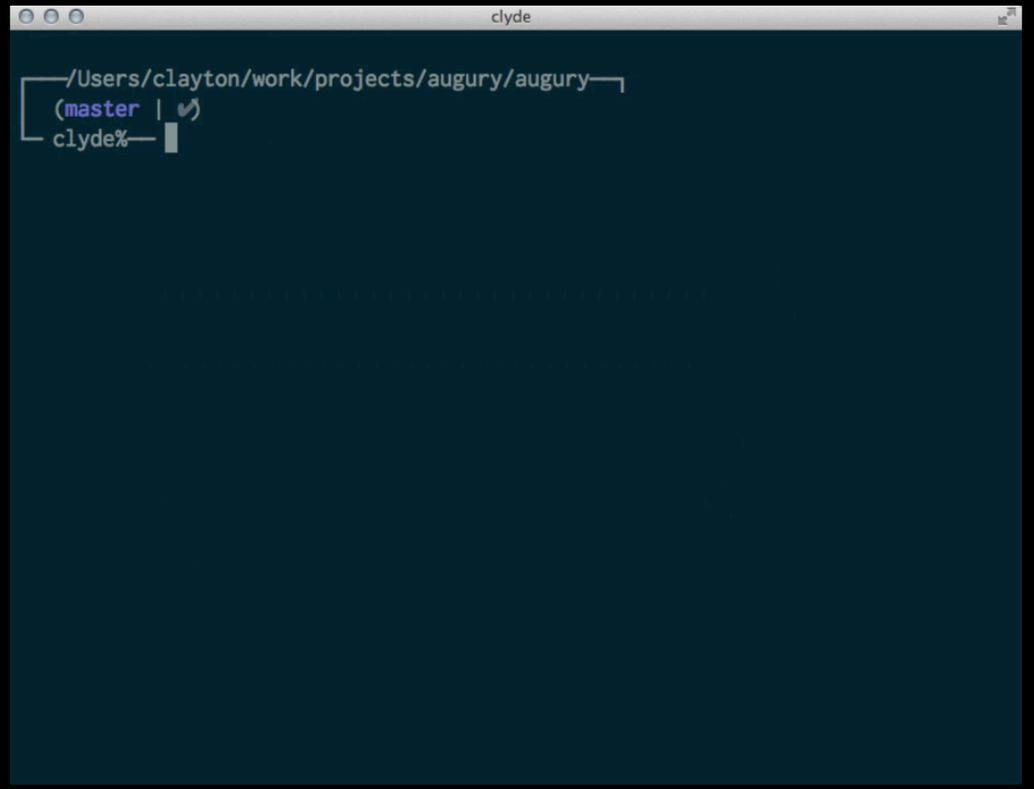
Commands to inspect models and routes


DEBUGGING

RUNTIME INVOCATION

```
require 'pry'
binding.pry
```

Debugging



★ Watch the Video ★

THE END

LINKS

indy.rb
Pry Website
Repeat last command
Command Aliases

Plugins

pry-awesome_print
pry-byebug
pry-stack_explorer
pry-clipboard
pry-rails

PHOTO CREDITS

And now for my first trick... (@vtornick)
Halp! (@daveblog)
Customize (@hz536n)
Plugins (@antpaniagua)
Debugging (@eduardox)
Links (@volvob12b)
⬅ This Picture (@rhodes)

Made possible by Creative Commons