GET TO THE CODE

Debugging

- Figure out why the code is broken
- Fix it
- Repeat.

puts

- Most common debugging tool
- Very easy to use
- Gather information one chunk at a time

binding.pry

- Opens pry at that line in your code
- `gem install pry`
- Gather as much information as you need
- Test theories immediately

pry-rescue

- Automatic binding.pry on unhandled exceptions
- `gem install pry-rescue`
- Reduces feedback cycle times

NoMethodError

- Most common exception in ruby code?
- Caused by:
  - Typos
  - Forgetting the right method name
  - Using the wrong object

`ls --grep`

- Finds the correct method
- No need to google
- Doesn't rely on gems having docs ;)

```
[1] pry(main)> ls Base64 --grep encode
Base64.methods: encode64
strict_encode64  urlsafe_encode64
[2] pry(main)>
```

`ls --grep`

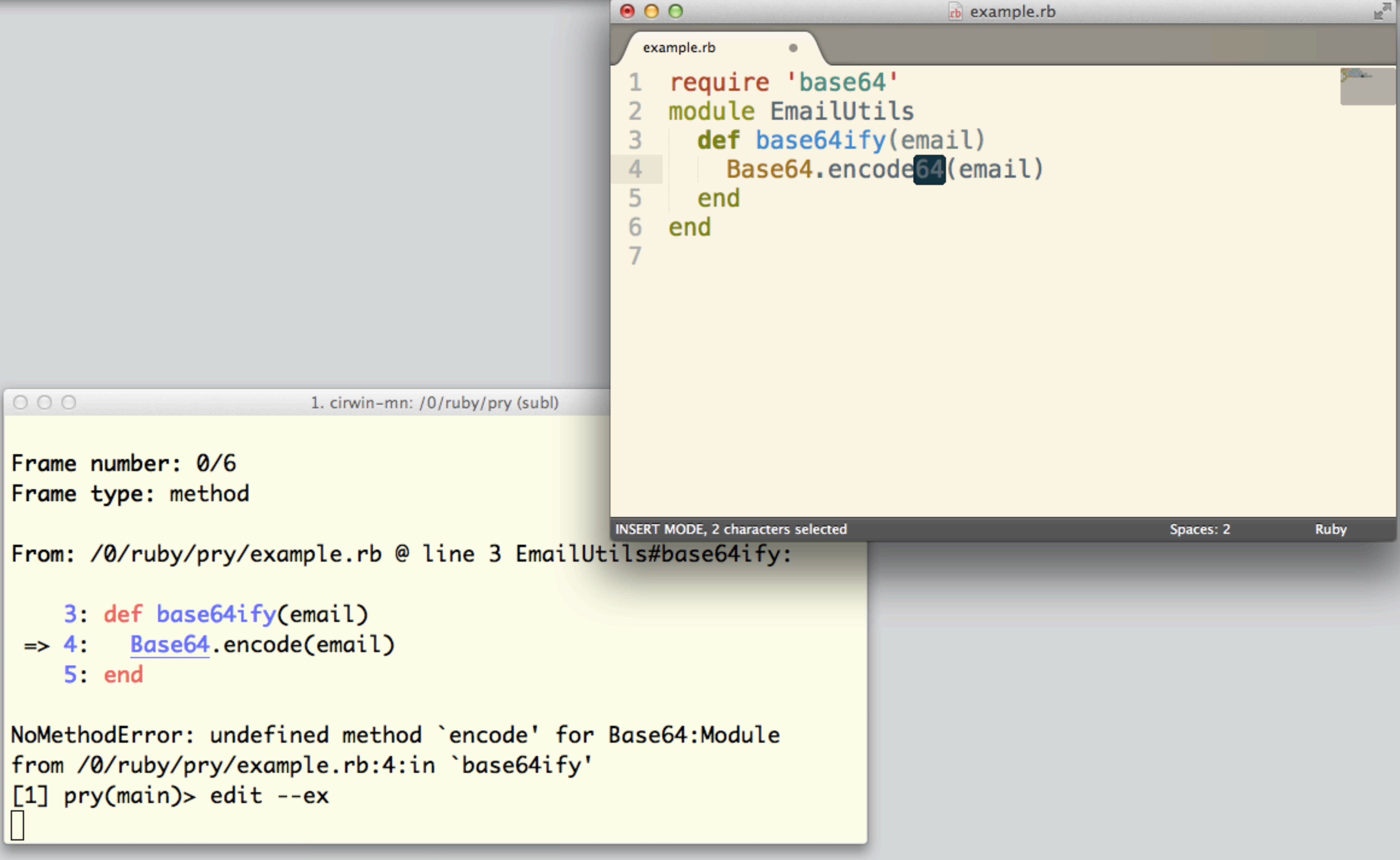
- Finds the correct method
- No need to google
- Doesn't rely on gems having docs ;)

`edit --ex`

- Opens your text editor
- Jumps to the exception
- Reloads the code when you're done

```
From: /0/ruby/pry/example.rb @ line 3:

    3: def base64ify(email)

 => 4:   Base64.encode(email)

    5: end

NoMethodError: undefined method
 `encode' for Base64:Module'

[1] pry(main)>
```

```
From: /0/ruby/pry/example.rb @ line 3:
    3: def base64ify(email)
 => 4:   Base64.encode(email)
    5: end
NoMethodError: undefined method
 `encode' for Base64:Module'
[1] pry(main)> edit --ex
```



```ruby
require 'base64'
module EmailUtils
  def base64ify(email)
    Base64.encode64(email)
  end
end
```

INSERT MODE, 2 characters selected   Spaces: 2   Ruby

```

Frame number: 0/6
Frame type: method

From: /0/ruby/pry/example.rb @ line 3 EmailUtils#base64ify:

    3: def base64ify(email)
 => 4:   Base64.encode(email)
    5: end

NoMethodError: undefined method `encode' for Base64:Module
from /0/ruby/pry/example.rb:4:in `base64ify'
[1] pry(main)> edit --ex
```

```
[1] pry(main)> edit --ex
    3: def base64ify(email)
 => 4:   Base64.encode64(email)
    5: end

[2] pry(main)> base64ify("hello world")
"aGVsbG8gd29ybGQ=\n"
```

up and down

- Moves pry up and down the call stack
- Figure out why you have the wrong object
- Discover why a gem doesn't work
- `gem install pry-stack_explorer`

```
From: /0/ruby/pry/example2.rb @ line 2:
    2: def make_safe(text)
 => 3:   text.gsub(/[^a-z]/i, '-')
    4: end
NoMethodError: undefined method `gsub'
  for nil:NilClass'
[1] pry(main)>
```

```
[1] pry(main)> up
From: /0/ruby/pry/example2.rb @ line 6:
    2: def safe_title(post)
 => 3:   make_safe(post[:title])
    4: end
[2] pry(main)> post
{"title" => "Hello Pry"}
```

up and down

- Moves pry up and down the call stack
- Figure out why you have the wrong object
- Discover why a gem doesn't work
- `gem install pry-stack_explorer`

$ and ?

- show-source and show-doc
- Instant documentation when you need it
- No need to `cd` into gem directories.

```
[1] pry(main)> $ safe_title
    2: def safe_title(post)
    3:   make_safe(post[:title])
    4: end
[2] pry(main)>
```

```
[2] pry(main)> ? safe_title
Convert the title of the blog into a
  string suitable for use in URLs.

param [Hash] post
return [String]
[3] pry(main)>
```

In conclusion

- Debugging requires gathering information
- Get into a `binding.pry` habit
- Explore pry's extra features
  - gem install pry-full
  - Type `help` inside pry

<EOF>

- @ConradIrwin (github, twitter, gmail, etc.)
- http://pryrepl.org/
- irc://freenode.net/#pry