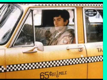
A Journey Through Abstraction

With Holt and Corinne



Overview

More output less code

Define Method vs. Define Singleton Method

Metaprogramming with boats

Stripping code to make it dynamic

Problems we encountered

Cautionary tales

Define method vs define singleton method

define_method creates methods for instance objects

define_singleton_method creates methods for the class

In case you forgot what that means...

define_method ---> “method_name”

define_singleton_method ---> “*self*.method_name”

Remember this?

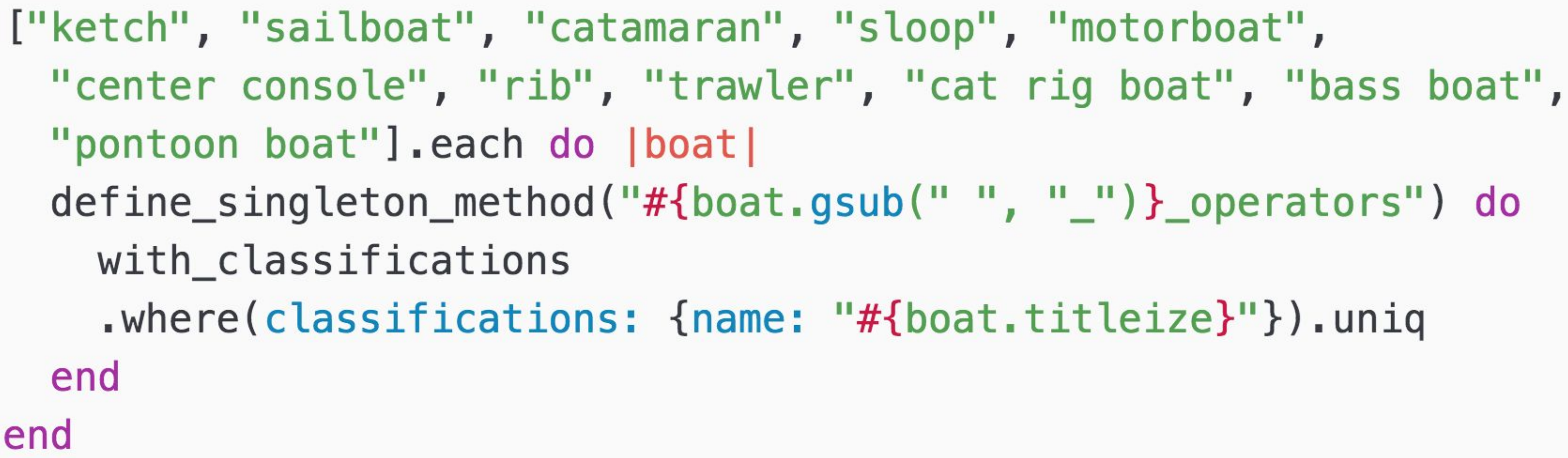
```ruby
class Captain < ActiveRecord::Base
  has_many :boats
  has_many :boat_classifications, through: :boats
  has_many :classifications, through: :boat_classifications

  def self.with_classifications
    joins(:classifications)
  end

  def self.catamaran_operators
    with_classifications
     .where(classifications: {name: 'Catamaran'}).uniq
  end

  def self.sailboat_operators
    with_classifications
     .where(classifications: {name: 'Sailboat'}).uniq
  end
end
```

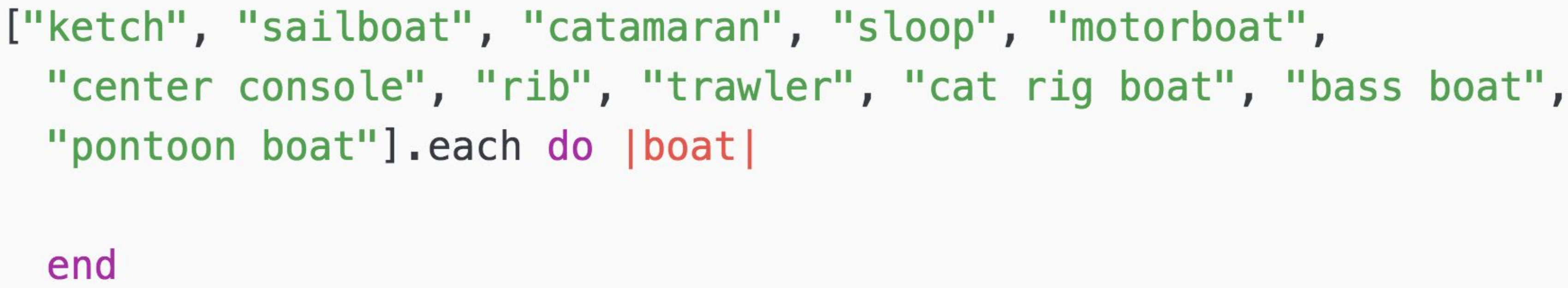
The slightly more terrifying method



```
["ketch", "sailboat", "catamaran", "sloop", "motorboat",
  "center console", "rib", "trawler", "cat rig boat", "bass boat",
  "pontoon boat"].each do |boat|
  define_singleton_method("#{boat.gsub(" ", "_")}_operators") do
    with_classifications
    .where(classifications: {name: "#{boat.titleize}"}).uniq
  end
end
```

Step One

The Array

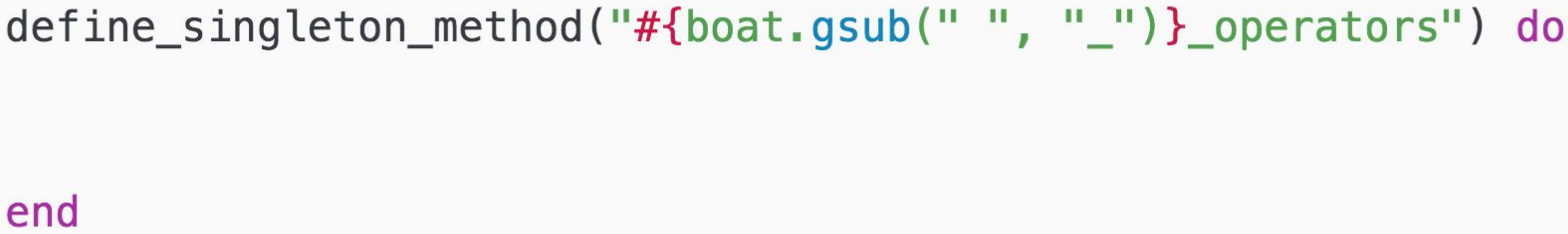


```
["ketch", "sailboat", "catamaran", "sloop", "motorboat",
  "center console", "rib", "trawler", "cat rig boat", "bass boat",
  "pontoon boat"].each do |boat|

  end
```

Step Two

The Singleton

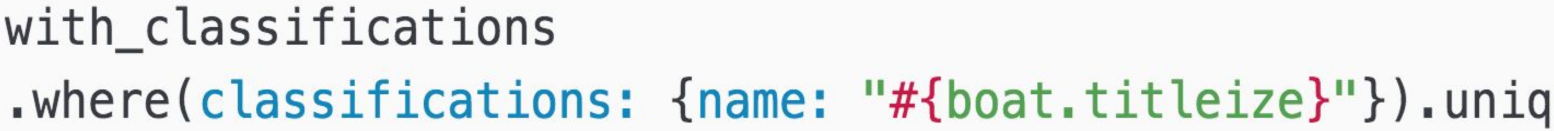


```ruby
define_singleton_method("#{boat.gsub(" ", "_")}_operators") do

end
```

Step Three

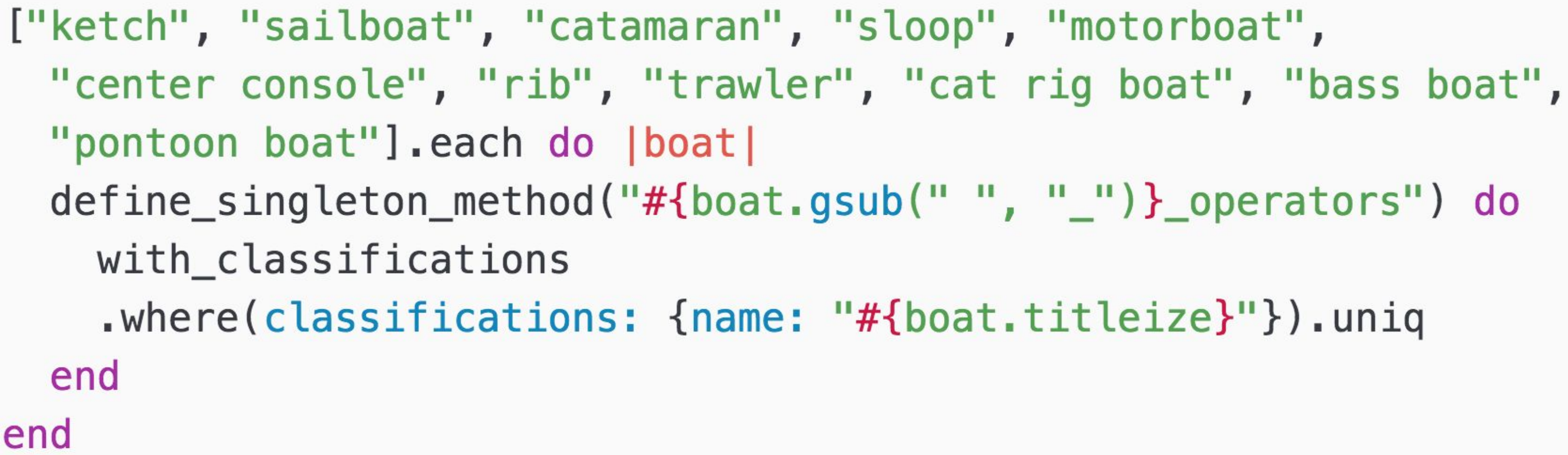
Original Method



```
with_classifications
.where(classifications: {name: "#{boat.titleize}"}).uniq
```

Step Four

Bringing It All Together



```ruby
["ketch", "sailboat", "catamaran", "sloop", "motorboat",
  "center console", "rib", "trawler", "cat rig boat", "bass boat",
  "pontoon boat"].each do |boat|
  define_singleton_method("#{boat.gsub(" ", "_")}_operators") do
    with_classifications
    .where(classifications: {name: "#{boat.titleize}"}).uniq
  end
end
```

METHODS WE CAN NOW CALL IN CAPTAIN CLASS

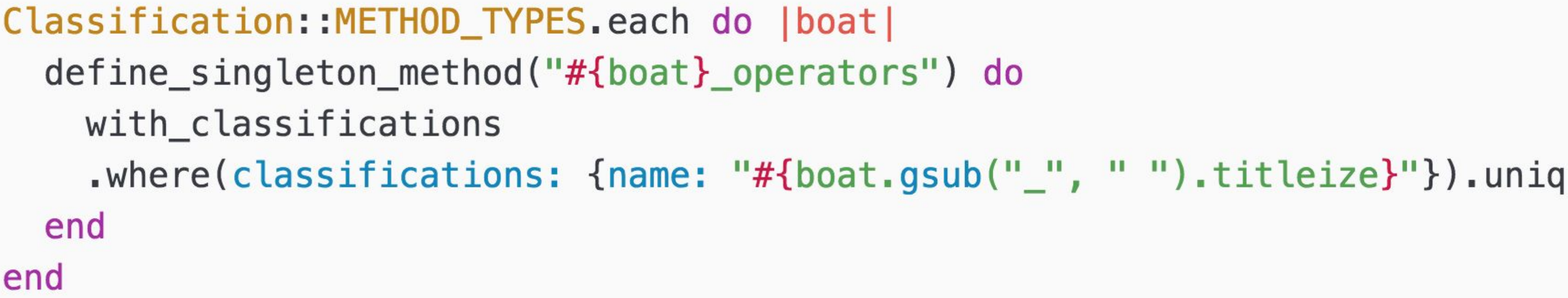
```
self.ketch_operators
self.catamaran_operators
self.sailboat_operators
self.sloop_operators
self.motorboat_operators
self.center_console_operators
```

```
self.rig_operators
self.trawler_operators
self.cat_rig_boat_operators
self.bass_boat_operators
self.pontoon_boat_operators
```

WHOA! WE'LL NEVER HAVE TO WORRY ABOUT CAPTAINS AGAIN!

“Ultra Meta-Magic”

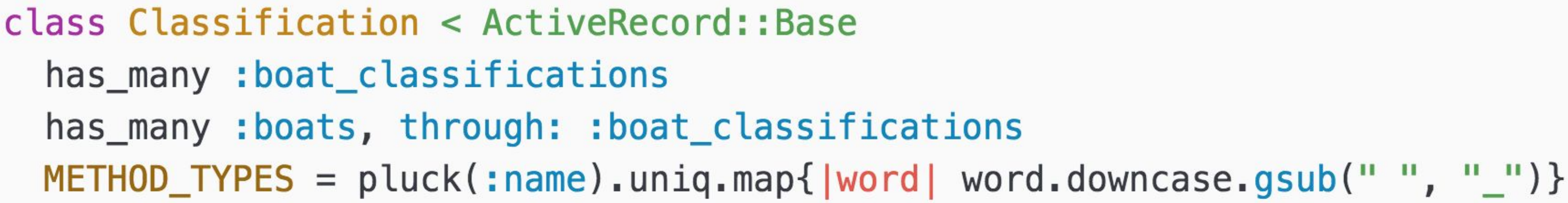
Scope Resolution Operators and gsub



```
Classification::METHOD_TYPES.each do |boat|
  define_singleton_method("#{boat}_operators") do
    with_classifications
    .where(classifications: {name: "#{boat.gsub("_", " ").titleize}"}).uniq
  end
end
```

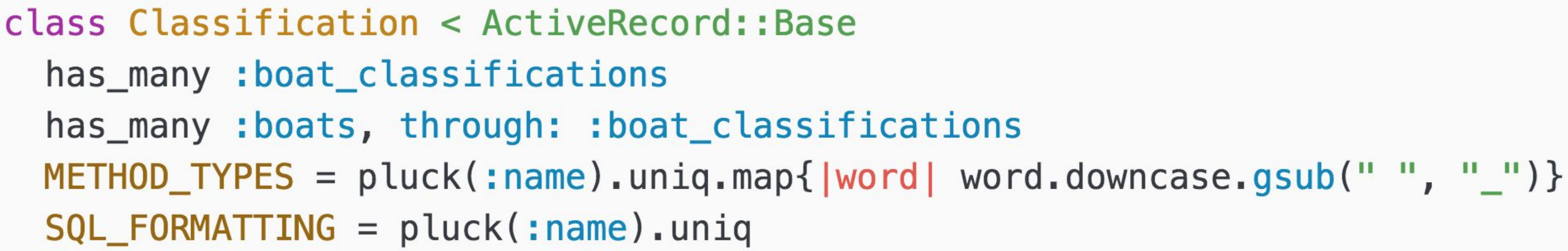
Where did all the code go?!

Don’t forget to declare your constant(s)!



```ruby
class Classification < ActiveRecord::Base
  has_many :boat_classifications
  has_many :boats, through: :boat_classifications
  METHOD_TYPES = pluck(:name).uniq.map{|word| word.downcase.gsub(" ", "_")}
```

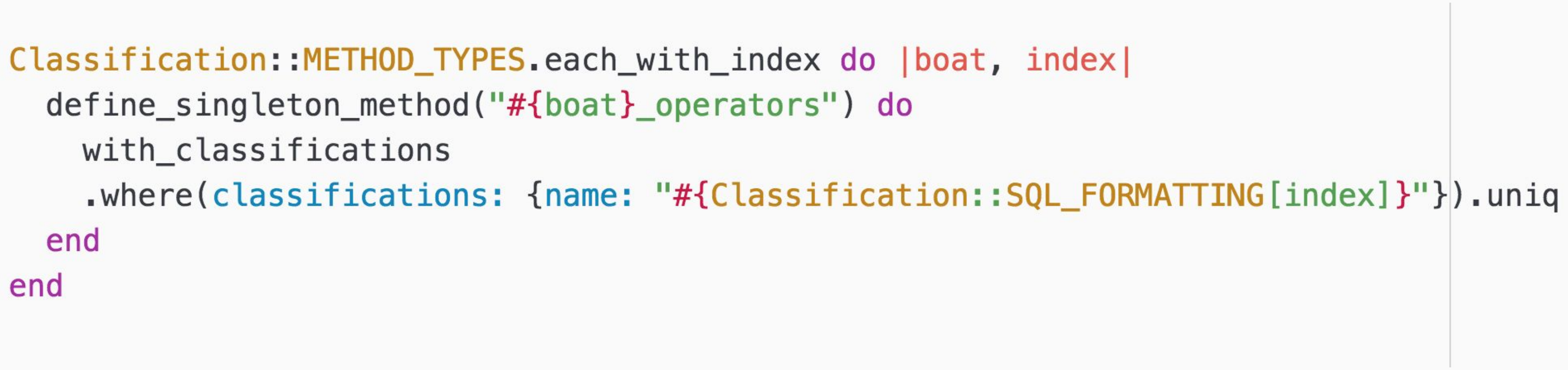
-OR-



```ruby
class Classification < ActiveRecord::Base
  has_many :boat_classifications
  has_many :boats, through: :boat_classifications
  METHOD_TYPES = pluck(:name).uniq.map{|word| word.downcase.gsub(" ", "_")}
  SQL_FORMATTING = pluck(:name).uniq
```

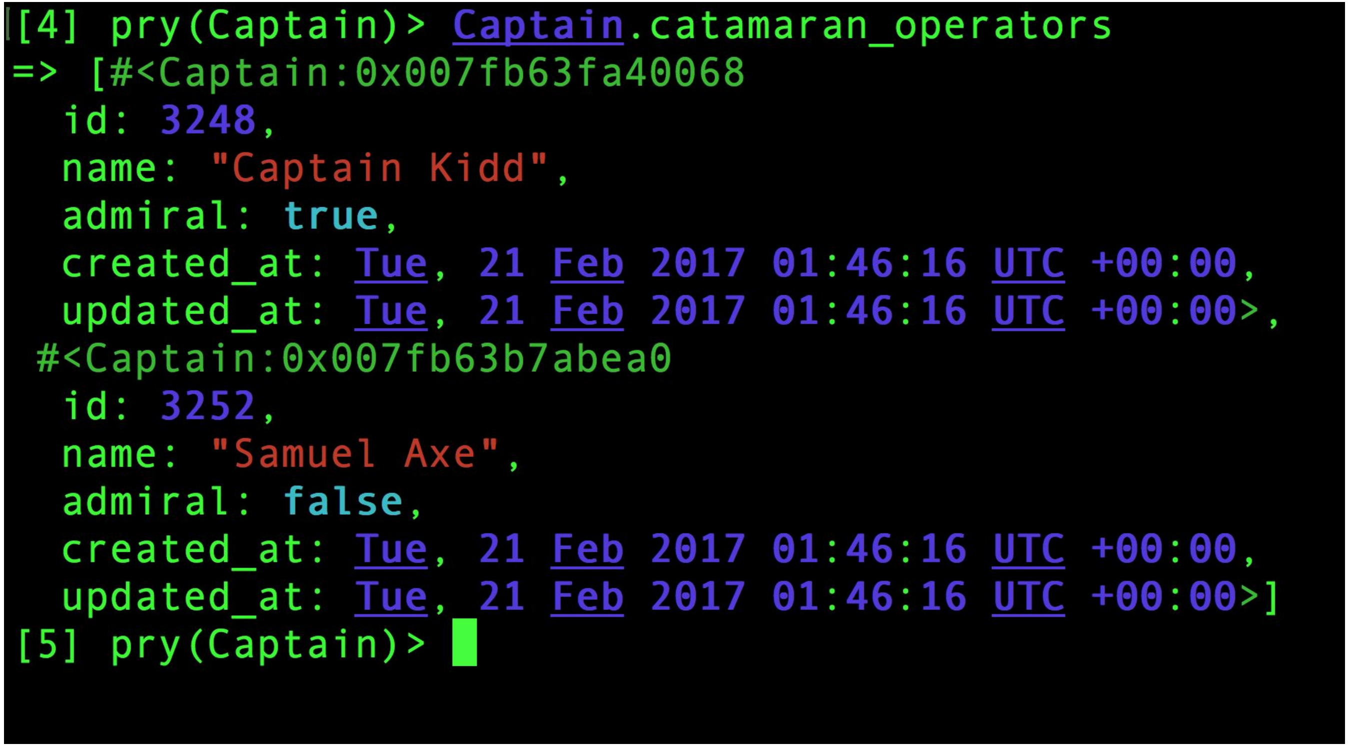
But wait, there's more!

Scope Resolution Operators x2



```
Classification::METHOD_TYPES.each_with_index do |boat, index|
  define_singleton_method("#{boat}_operators") do
    with_classifications
    .where(classifications: {name: "#{Classification::SQL_FORMATTING[index]}"}).uniq
  end
end
```

The Singleton Method In Action!



```
[4] pry(Captain)> Captain.catamaran_operators
=> [#<Captain:0x007fb63fa40068
  id: 3248,
  name: "Captain Kidd",
  admiral: true,
  created_at: Tue, 21 Feb 2017 01:46:16 UTC +00:00,
  updated_at: Tue, 21 Feb 2017 01:46:16 UTC +00:00>,
 #<Captain:0x007fb63b7abea0
  id: 3252,
  name: "Samuel Axe",
  admiral: false,
  created_at: Tue, 21 Feb 2017 01:46:16 UTC +00:00,
  updated_at: Tue, 21 Feb 2017 01:46:16 UTC +00:00>]
[5] pry(Captain)> 
```

Things to take with you on your own journeys

Singleton exists

Where/when are you seeding your database?

Rails C

Scope Resolutions Operators and Constants

Need more sleep? Make your code do all the work!