

@oheydrew

A BREAKDOWN OF METHOD ARGUMENTS

ARGUING WITH RUBY 













What does '*professional
code*' look like?



What does *self-documenting*
code look like?

ARGS!



Weigh Anchor! Hoist The Mainsails! AARGGS! 🏴‍☠️⚔️

```
def pirate_greet(name)
  puts("Ahoy there, #{name}! ...ARG!".upcase)
end
```

```
$ pirate_greet("Roro Melbourne")
```

```
# => "AH0Y THERE, R0R0 MELBOURNE! ...ARG!"
```

Positional ARGS!

```
class Pirate
  def say(phrase_one, phrase_two, phrase_three)
    puts("Ahoy there, #{phrase_one}! #{phrase_two}!
        #{phrase_three}! ARRRRRRG!".upcase)
  end
end
```

Positional ARGS!

```
class Pirate
  def say(phrase_one, phrase_two, phrase_three)
    puts("Ahoy there, #{phrase_one}! #{phrase_two}!
        #{phrase_three}! ARRRRRRG!".upcase)
  end
end

$ totally_normal_pirate = Pirate.new()
$ totally_normal_pirate.say("Matey",
                           "Hoist the main sails",
                           "Walk the plank, ye scurvy dogs")

# => "AHOY THERE, MATEY! HOIST THE MAIN SAILS! WALK THE
      PLANK, YE SCURVY DOGS! ARRRRRRG!"
```


Positional ARGS!

- The position of the arguments **matters**
- The number of arguments **matters**

Passing args as an Array

```
[ item1, item2, item3 ]
```

The 'Splat' Operator (*) (*args)

Automatically transforms arguments into an array

***args** - The 'Splat' Operator (*)

```
class Pirate
  def say_many(*phrases)
    full_sentence = phrases.map do |phrase|
      "#{phrase}! ".upcase
    end
    puts("AH0Y THERE! #{full_sentence.join(" ")} ARRRRRG!")
  end
end
```

args** - The **'Splat'** Operator (**)

```
class Pirate
  def say_many(*phrases)
    full_sentence = phrases.map do |phrase|
      "#{phrase}! ".upcase
    end
    puts("AHoy THERE! #{full_sentence.join(" ")} ARRRRRG!")
  end
end
```

```
$ smart_pirate = Pirate.new()
$ smart_pirate.say_many("Avast", "Pass in all the args",
                        "However ye darned well please!",
                        "Well I'll be the kraken's uncle!",
                        "Ain't that a fancy jig?")
```

```
# => AHoy THERE! AVAST! PASS IN ALL THE ARGS! HOWEVER YE
DARNED WELL PLEASE!! WELL I'LL BE THE KRAKEN'S UNCLE!! AIN'T
THAT A FANCY JIG?! ARRRRRG!
```


Splat operator caveats

- Great for unknown quantities (ie processing database records)
- Not so great for clarity
- Not so great when dealing with multiple types of data

Passing args as a hash

```
{ key: value }
```

* = 'Splat'

** = 'Double Splat'



The 'Double Splat' Operator (*) (**args)

Automatically transforms arguments into a hash

****args** - The 'Double-Splat' Operator (******)

```
class Pirate
  def initialize(**pirate_args)
    @name = pirate_args[:name]
    @vessel = pirate_args[:vessel]
    @fearsome_quote = pirate_args[:fearsome_quote]
  end
end

$ dotcom = Pirate.new(name: "Kim Dotcom",
                      vessel: "megaupload.com",
                      fearsome_quote: "I'm not a pirate- I'm
                                      an innovator.")

# => #<Pirate:0x00007f7fd40a18a0 @name="Kim Dotcom",
@vessel="megaupload.com", @fearsome_quote="I'm not a pirate-
I'm an innovator.">
```




Ruby Keyword Arguments (`keyword:`)

The holy grail of ruby method arguments



Keyword Arguments

```
class Pirate
  def initialize(name:, vessel:, fearsome_quote:)
    @name = name
    @vessel = vessel
    @fearsome_quote = fearsome_quote
  end
end

$ feathersword = Pirate.new(name: "Captain Feathersword",
                             vessel: "The Goodship Feathersword",
                             fearsome_quote: "Well, blow me down!")

# => #<Pirate:0x007fa4271769c0 @name="Captain Feathersword",
@vessel="The Goodship Feathersword", @fearsome_quote="Well,
blow me down!">
```

Keyword Arguments

```
class Pirate
  def initialize(name:, vessel:, fearsome_quote:)
    @name = name
    @vessel = vessel
    @fearsome_quote = fearsome_quote
  end
end
```

```
$ gonzo = Pirate.new(name: "The Great Gonzo",
                     fearsome_quote: "Off to Zanzibar, to
                                     fight the Zanzibarbarians!")
```

```
# => ArgumentError: missing keyword: vessel
```

Keyword Operator Defaults

```
class Pirate
  def initialize(name:, vessel: "Land lubber", fearsome_quote:)
    @name = name
    @vessel = vessel
    @fearsome_quote = fearsome_quote
  end
end
```

```
$ monkey_island = Pirate.new(name: "Guybrush Threepwood",
                              fearsome_quote: "You fight like a
                              dairy farmer!")
```

```
# => #<Pirate:0x00007feed8858418 @name="Guybrush Threepwood",
@vessel="Land lubber", @fearsome_quote="You fight like a dairy
farmer!">
```


Bringing it all together ✨🌈

Keyword operators clarify method calls *and* definitions

```
calculate_total(52178, 6.0, 12.50, 1.5)
```

```
calculate_total(item_id: 52178, qty: 6.0, price: 12.50, tax: 1.5)
```

Sometimes just simple naming can help clarify your code

```
def process(*args)
```

```
def process(*record)
```

Understanding method ar

```
validates(:password, pres
```



...args?

Thankyou!



```
def contact_drew(crowsnest_handle: @oheydrew,  
                 sea_port: www.oheydrew.me)
```