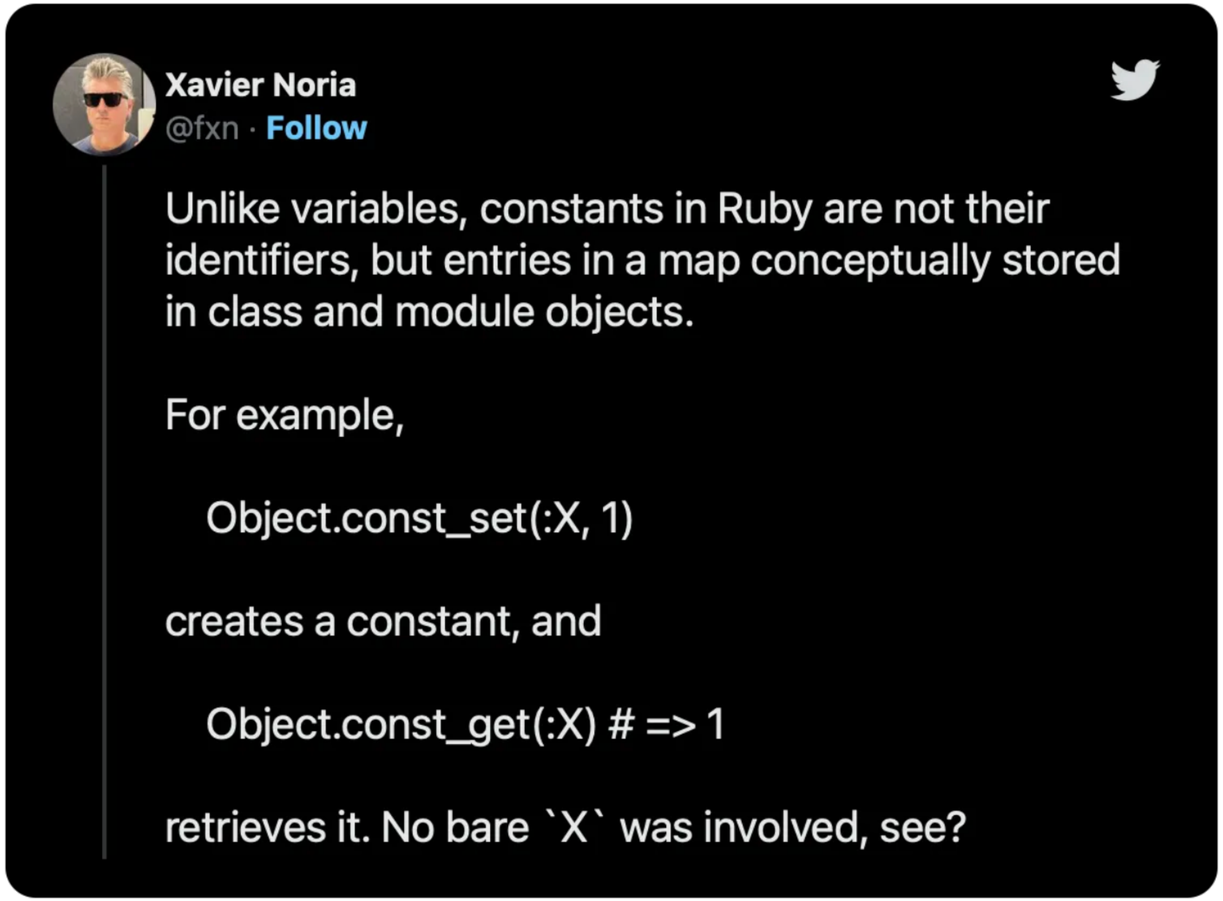
Short Ruby Newsletter

code content from Issue #9

for full newsletter visit newsletter.shortruby.com

curated by @lucianghinda

Xavier Noria shared a short explanation about how constants works in Ruby:



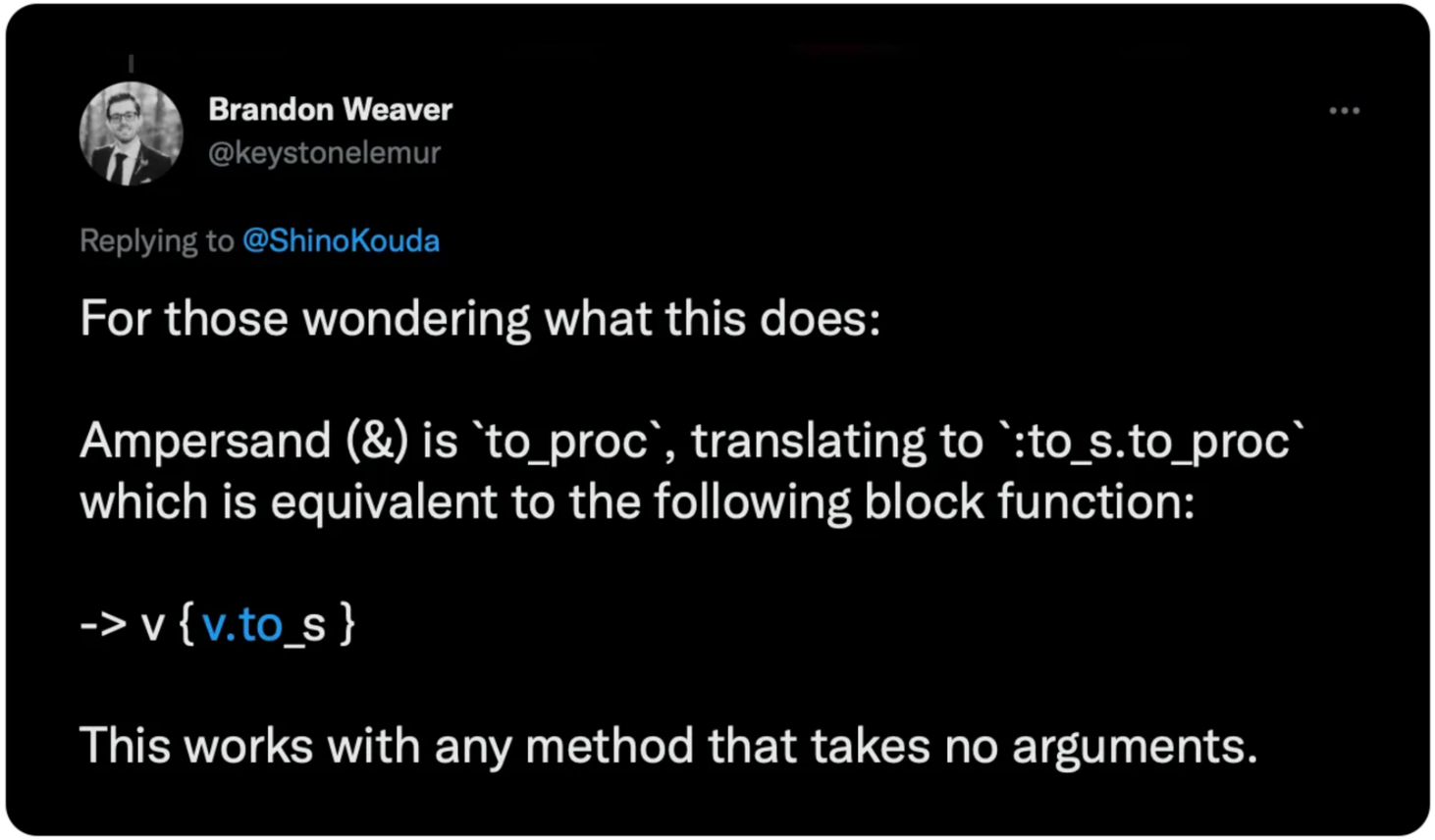
Source: @fxn on Twitter

Thiago Massa shared a new example of pattern matching with hashes:

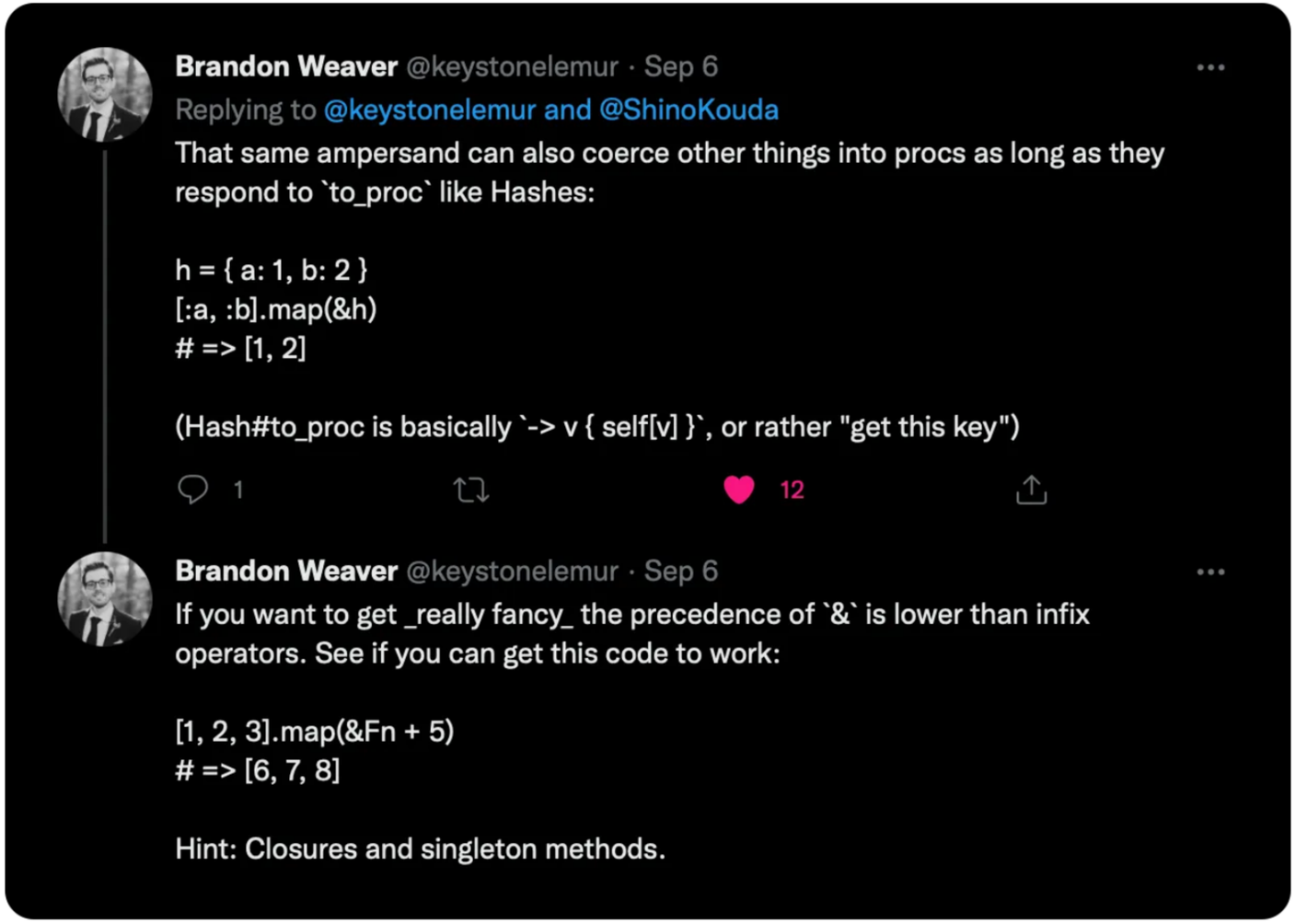
```ruby
# Creating a complex hash, with nested keys
my_complex_hash = {
  users: [
    {name: "Yukihiro Matsumoto", age: 57},
    {name: "Kabosu the Shiba Inu", age: 16},
    {name: "Thiago Massa", age: 33}
  ]
}

# What if we want to find the age of Matz?
my_complex_hash => {
 users: [*_, {name: "Yukihiro Matsumoto", age: matz_age}, *_]
}
matz_age # => 57

# 😱 WTF! What happened there?

# The _ just means we want to ignore the output and not want to assign it to a variable.

# We are using the Find pattern. (experimental feature on Ruby 3.1)
# By specifying *_ in the beginning and end we can search through the data structure...
# Similar to a Regexp! 🤯 So cool!

# Coming next we'll look into variable pinning in Pattern Matching 🙂
```

Source: @th1agofm on Twitter

Shino Kouda shared a short code example of .map(&:to_s) and Brandon Weaver explain more in depth how this works:



Source: @keystonelemur on Twitter



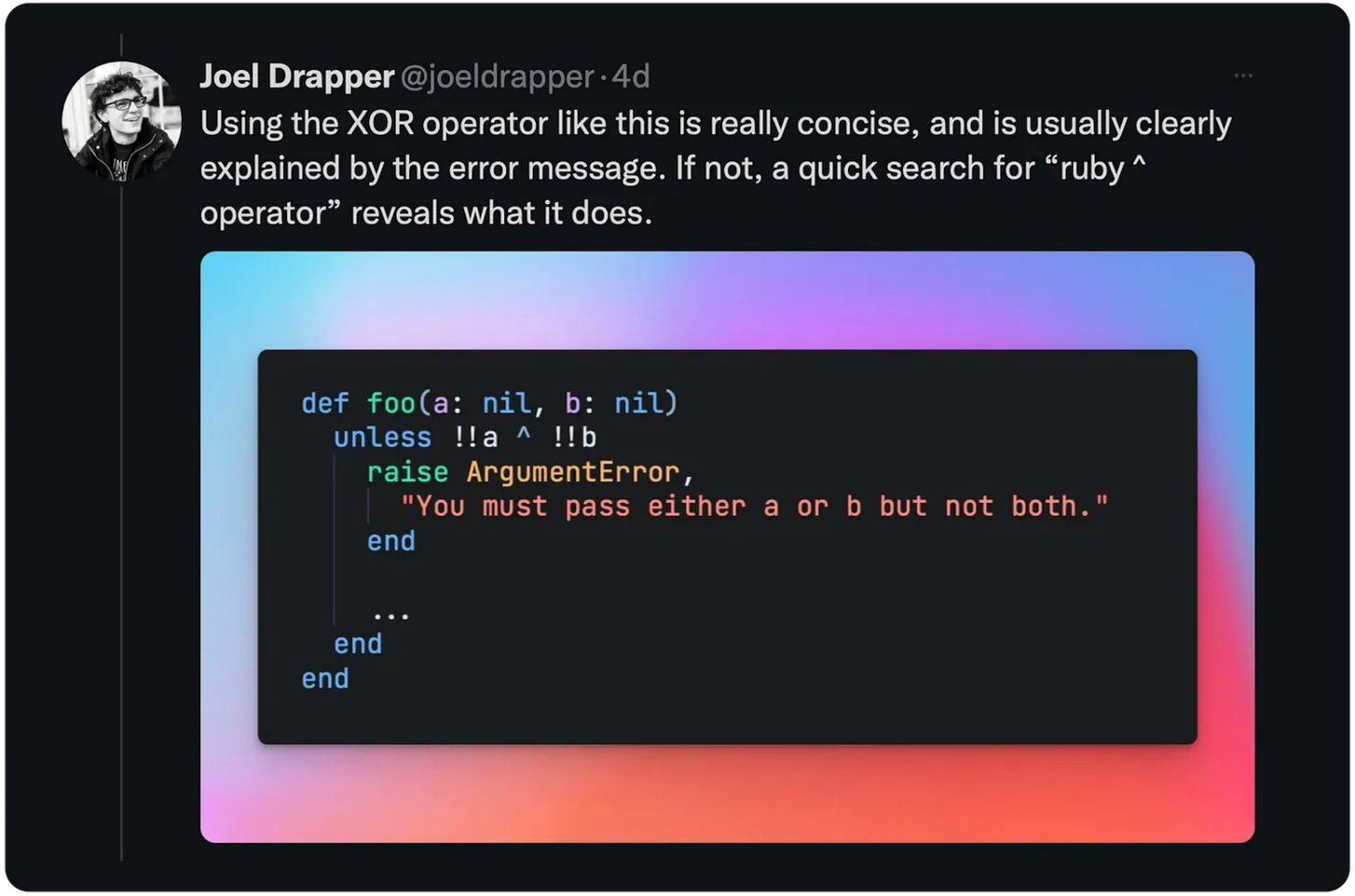
Source: @keystonelemur on Twitter

Thiago Massa shared an example of how to use the Pin operator (^) patten matching:

```
# The pin operator (^) is used in pattern matching to evaluate a variable instead of assigning it.

# That sounds complicated 🤔 We can understand it better with an example of its usage.

# Let's suppose we have an Array with some School grades
school_grades = [10, 9, 0]

# With pattern matching, we want to see if the first grade is a 10.
school_grades => [10, *]
# It matched successfuly! No exception.

# What if now we want to base our lookup from a dynamic variable instead of constant 10?
expected_first_grade = 5
school_grades => [expected_first_grade, *]
expected_first_grade # => 10

# Oh no. 😩
# We have overwritten expected_first_grade with 10 instead of looking it up.

# Let's try again now and use the pin(^) operator before our expected_first_grade variable.
expected_first_grade = 5
school_grades => [^expected_first_grade, *]
# [10, 9, 0]: 5 === 10 does not return true (NoMatchingPatternError)

# Nice! ✨ It used the content of the variable as we initially desired and didn't match!

# On my next Tweet we'll learn practical ideas on where to use pattern matching.
```

Source: @th1agofm on Twitter

Joel Drapper shared an example of another underused operator, XOR:



Source: @joeldrapper on Twitter

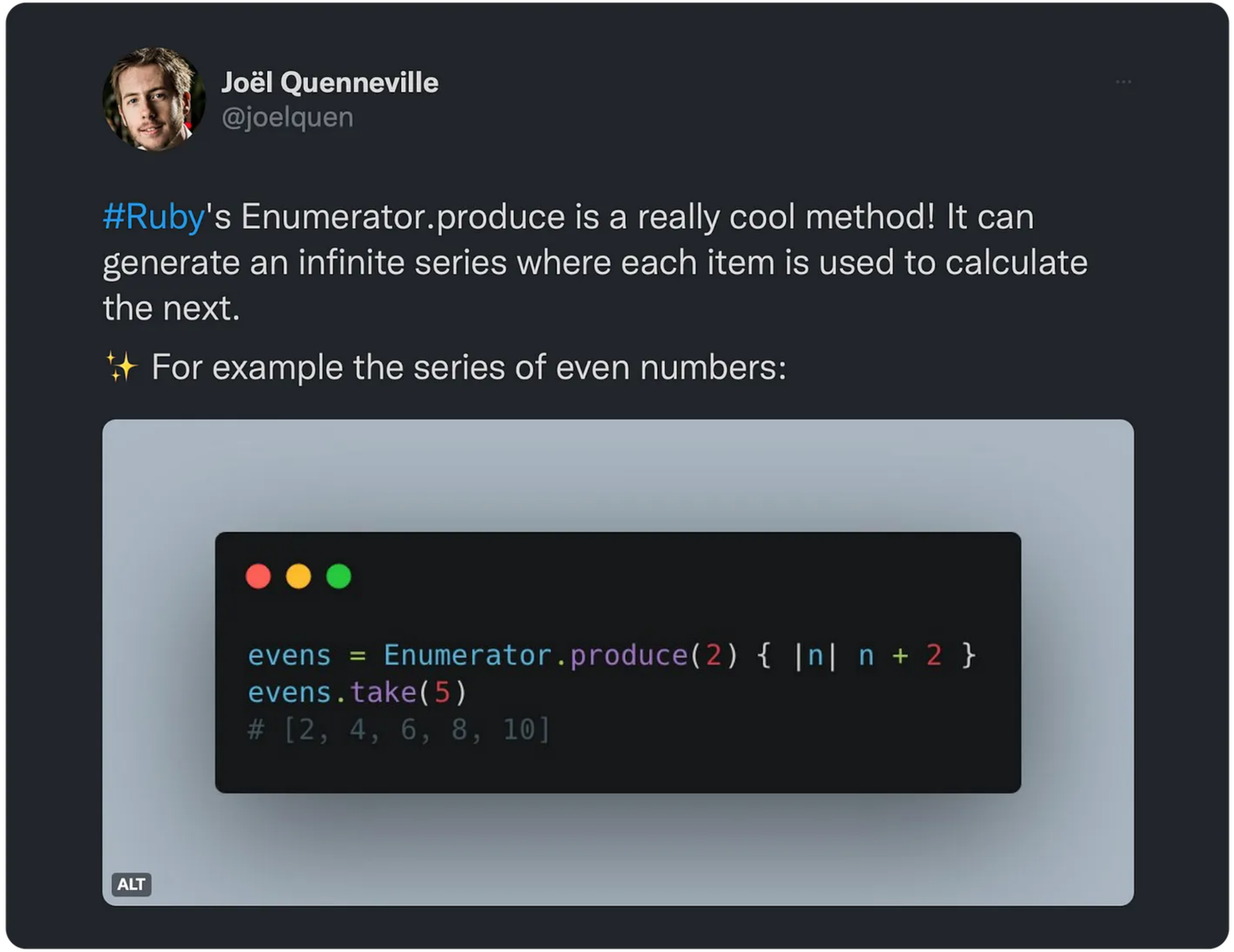
Lucian Ghinda shared another example of using XOR:

Source: @lucianghinda on Twitter

Kirill Shevchenko shared shared two code samples explaining how |= is creating a new array instead of adding to the existing one:

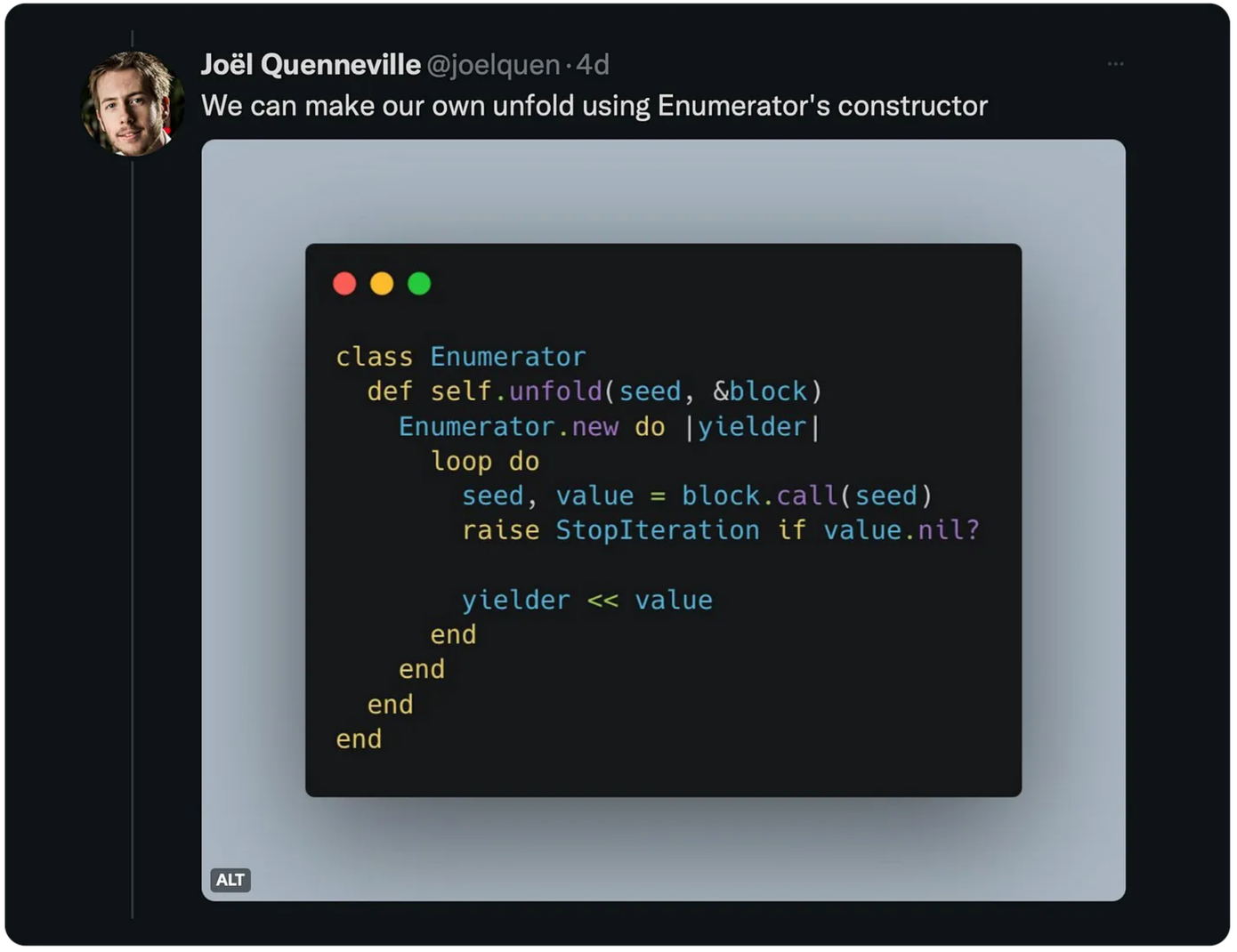
Source: @kirill_shevch on Twitter

Joël Quenneville shared code about how to generate infinite series with Enumerator:



Source: @joelquen on Twitter

As the produce cannot be used to generate series where the input from each step needs the output of the previous step, then he showed how to add unfold to the Enumerator to achieve more:



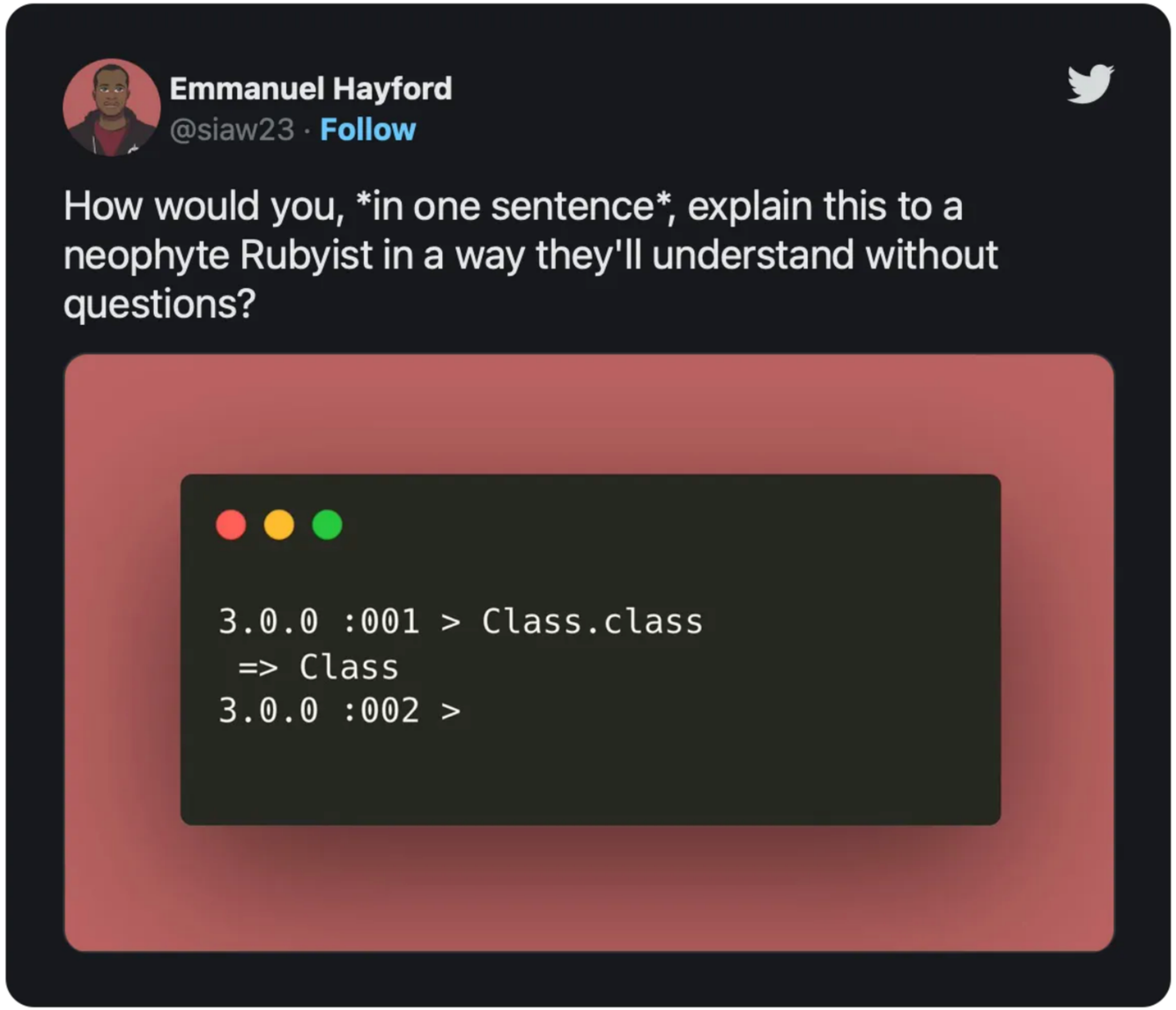
Source: @joelquen on Twitter

And here is how he showed how to use that:

```
squares = Enumerator.unfold(2) { |seed| [seed + 1, seed * seed] }
squares.take(5)
=> [4, 9, 16, 25, 36,]
```

Source: @joelquen on Twitter

Emmanuel Hayford asked a smart question: How to explain Class.class being Class:



Source: @siaw23 on Twitter

Here are some of the examples shared in that thread by Brandon Weaver, Lucian Ghinda, John Cheek, Ariel Caplan, Eric Halverson:

> All Ruby classes are objects of the type Class, including amusingly Class itself
> (source: @keystonelemur)

> The constant "Class" is also an object (instance of Class) yet somehow is also a class
> (source: @lucianghinda)

> You can instantiate a class with `.new`. Methods are looked up on an object's class, so `Array. new` finds the `new` method in `Array.class # => Class`. Further, they have set `Class.class # => Class`, so that you can programmatically create classes with `Class. new`"
> (source: @josh_cheek)

> The Class class is, like any class, an object whose methods are defined on that object's class, the Class class.
> (source: @amcaplan)

> Everything in Ruby is an object and Class is the mother of all objects including itself. 🙂
> (source: @elhalvers)

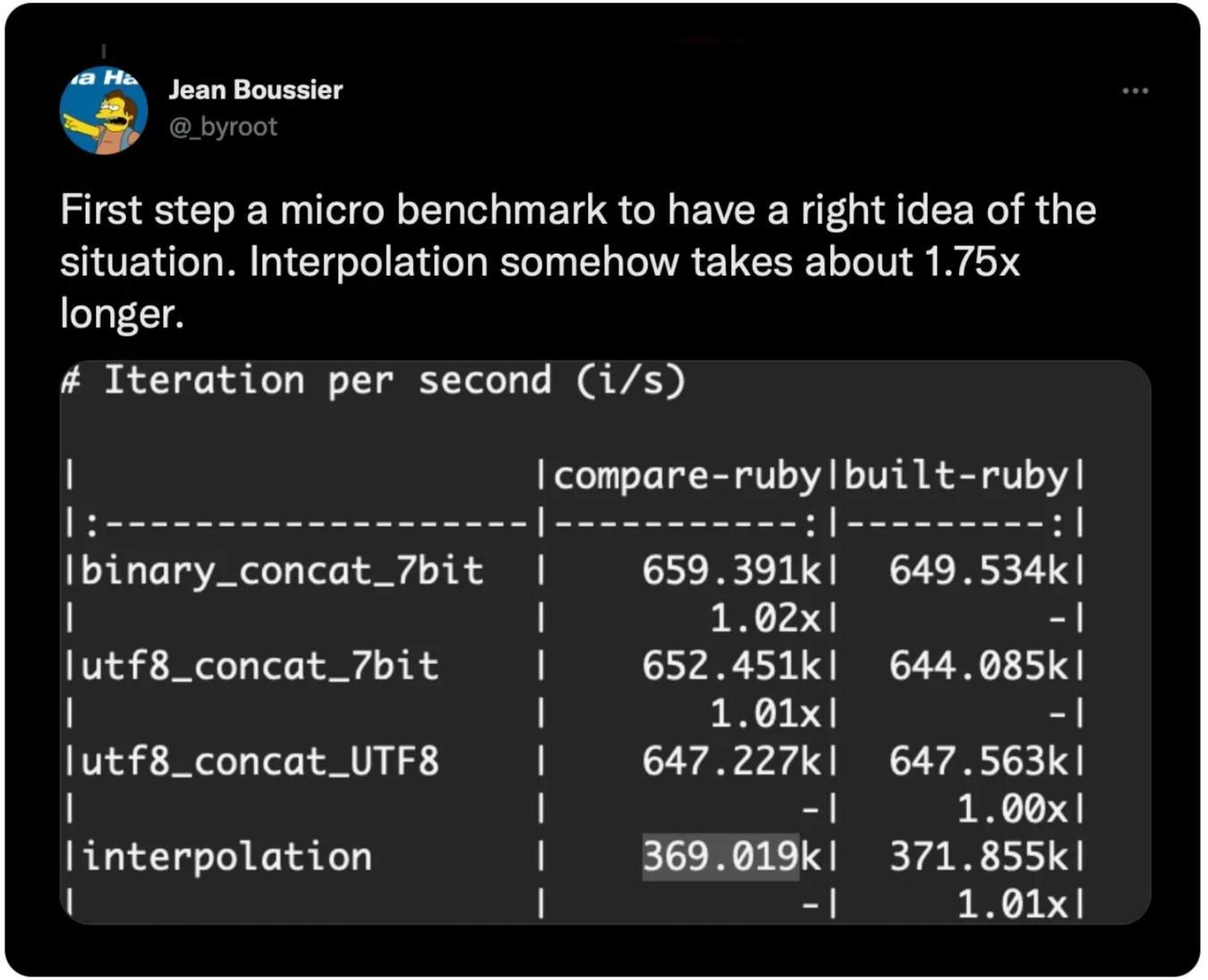
Brandon Weaver shared an experimental piece of code to implement proc_line:

Source: @keystonelemur on Twitter

Brandon Weaver shared an experimental piece of code to implement proc_line:

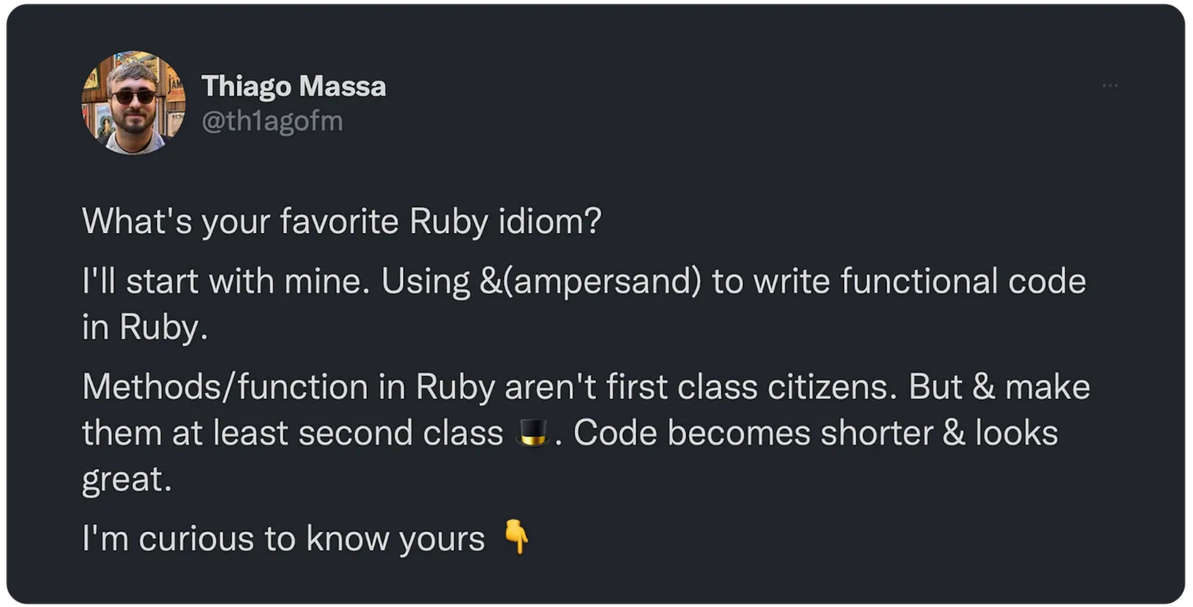
Source: @keystonelemur on Twitter

Jean Boussier shared their process to improve performance on string interpolation. Amazing read!



Source: @_byroot on Twitter

Thiago Massa asked a question and shared a code sample about ampersand Ruby idiom:



Source: @th1agofm on Twitter

And along with this question Thiago shared a code sample:

```ruby
# From:
[1,2,3].map {|n| n.odd? }
# => [true, false, true]

# To:
[1,2,3].map(&:odd?)
# => [true, false, true]

# It isn't exclusive for ruby methods
# You can also use on your own methods...

def my_method(_)
  10
end

[1,2,3].map(&method(:my_method))
# => [10, 10, 10]

# How it works?
# & calls #to_proc on the method converting it to a Proc
# Procs work pretty much like anonymous functions in Ruby
```

Source: @th1agofm on Twitter

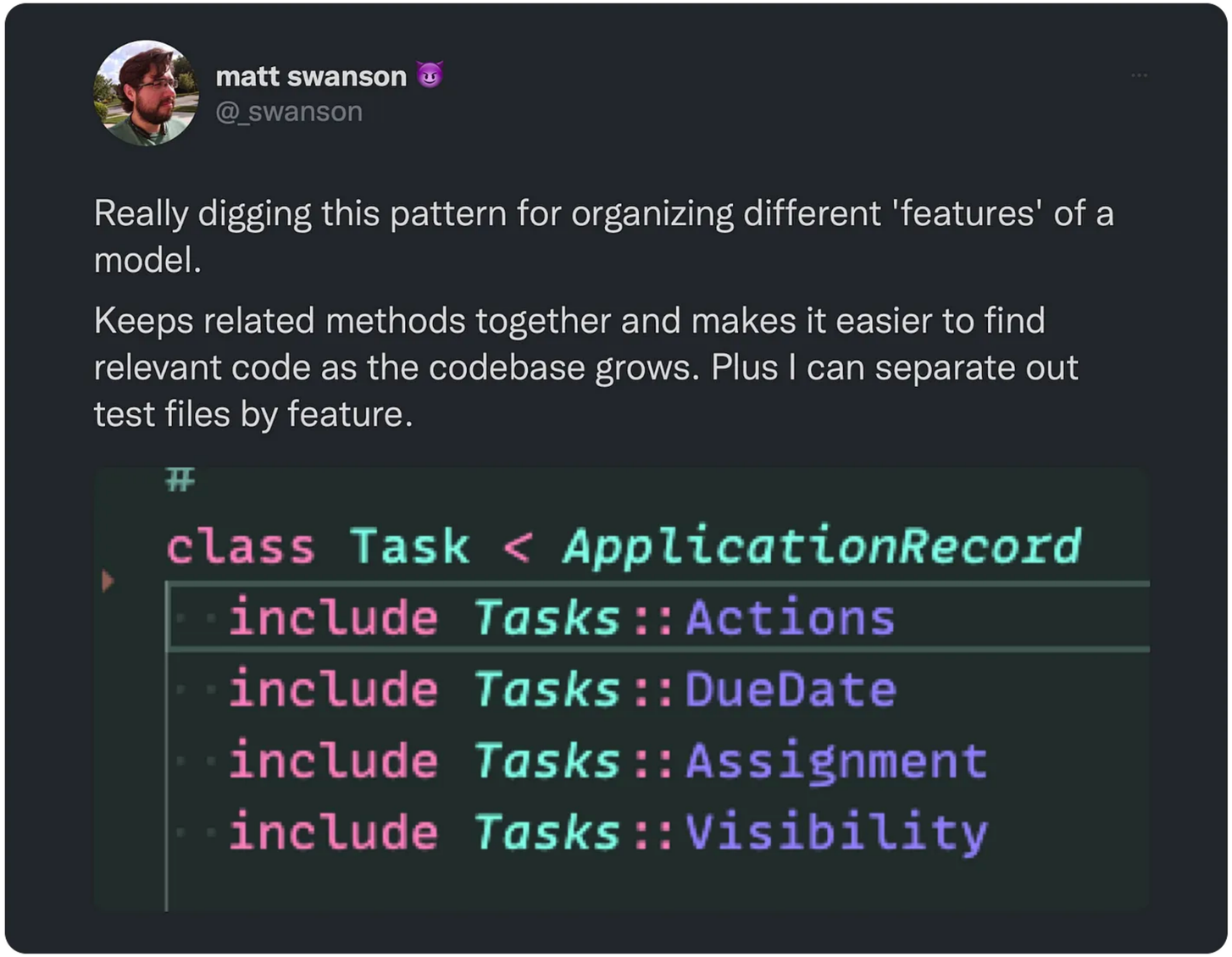
Joel Drapper shared a new update for Phlex.fun showing support for conditional CSS classes:

Source: @joeldrapper on Twitter

Cj Avilla shared code showing how to use right assisgnment / pattern matching:

Source: @cjav_dev on Twitter

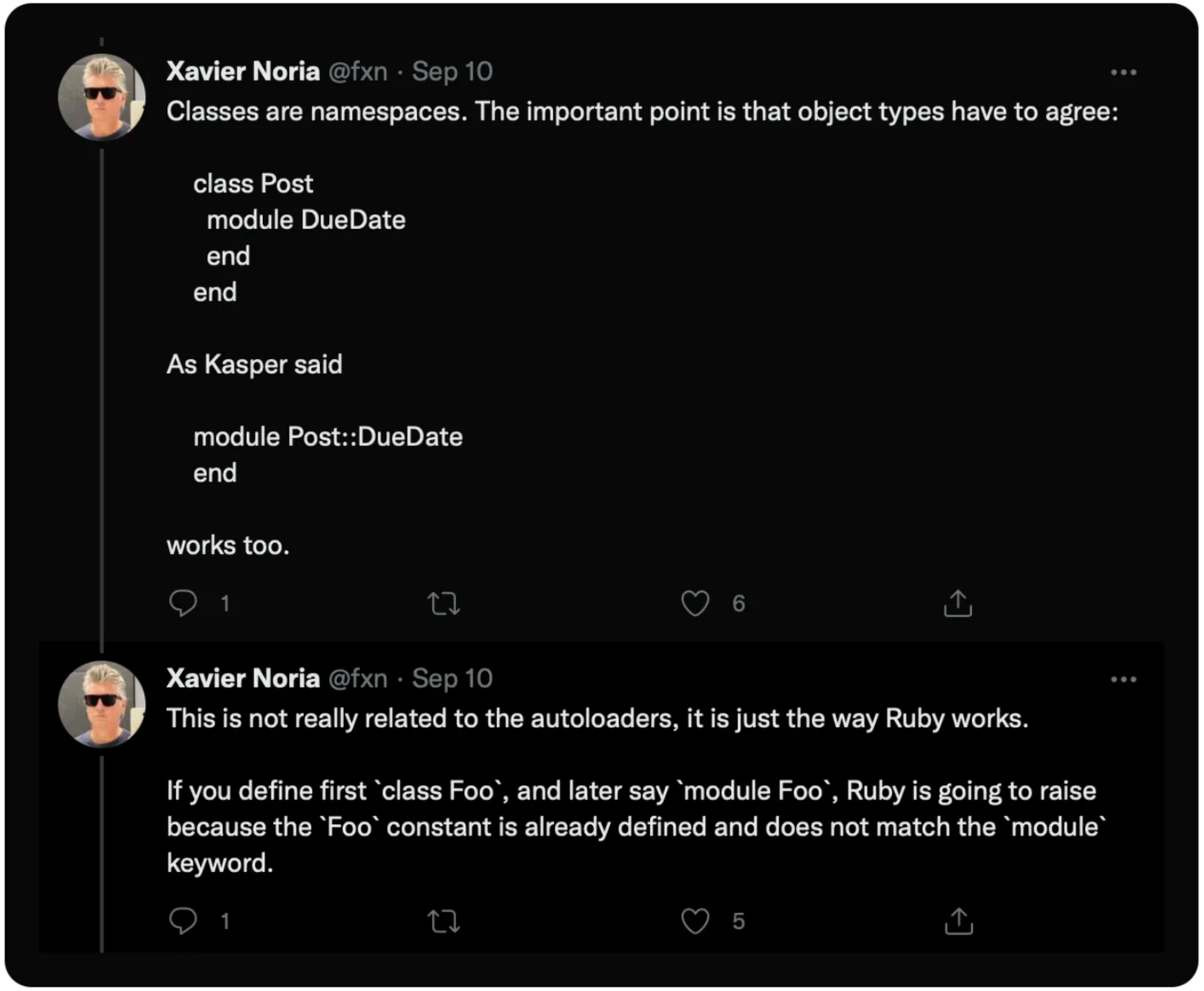
Matt Swanson shared code about how to use modules to organize features:



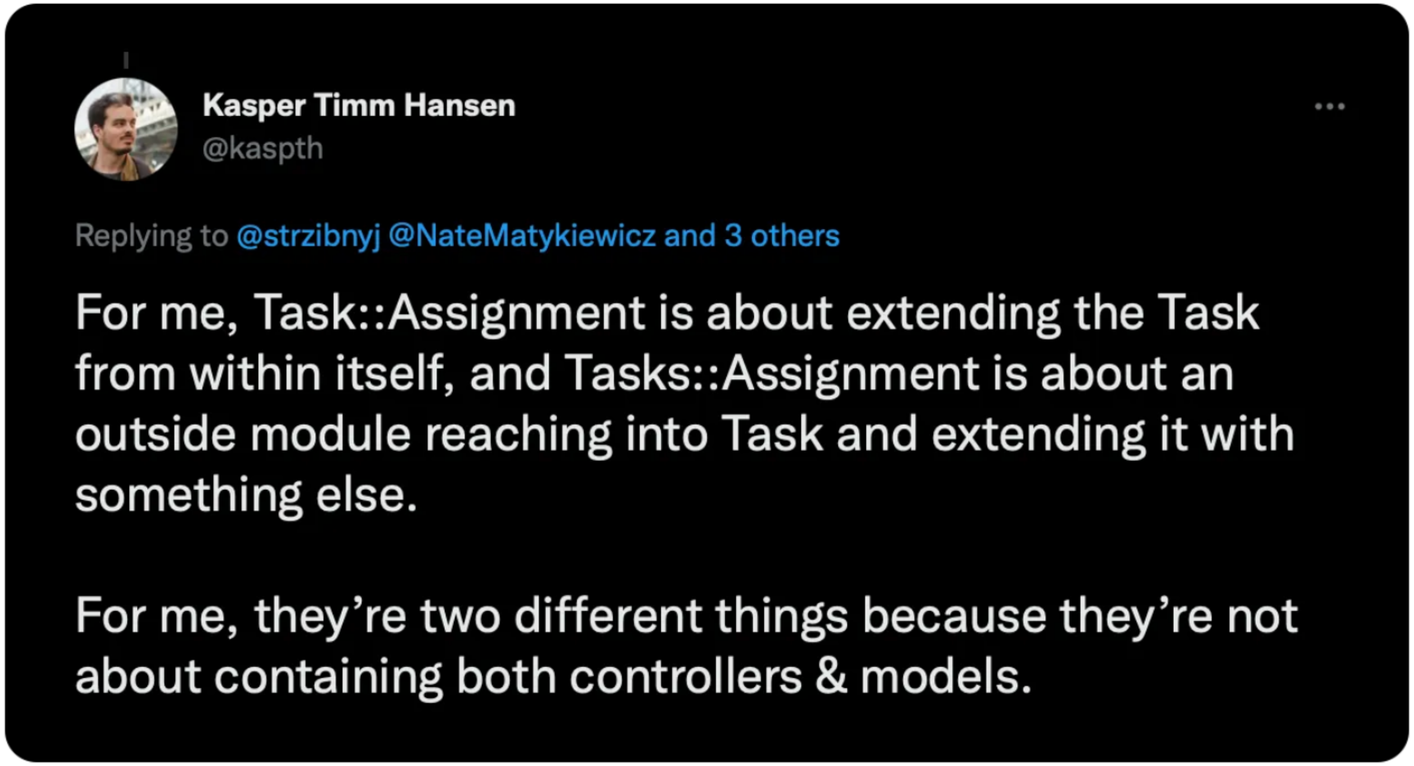
Source: @_swanson on Twitter

Here are some interesting replies, but I really recommend you to read the entire conversation (for people not using Twitter use this)as it has great sugestions and some great explanation for various styles:

```
# https://twitter.com/rockatanescu proposed:
class Task < ActiveRecords #in app/models/task.rb
  include Actions
end

module Task::Actions # in app/models/task/actions.rb
end

# https://twitter.com/gjtorikian proposed:
class Message < ActiveRecord
  include Relationships::Mesage
end

module Rerlationships::Message # in app/models/relationships/message.rb
end

# https://twitter.com/ahmedabdel3aaal proposed
class BusinessProfile < ActiveRecord
  include BusinessProfile::Scopes
end

module BusinessProfile::Scopes # in app/models/business_profile/scope.rb
end
```



Source: @fxn on Twitter

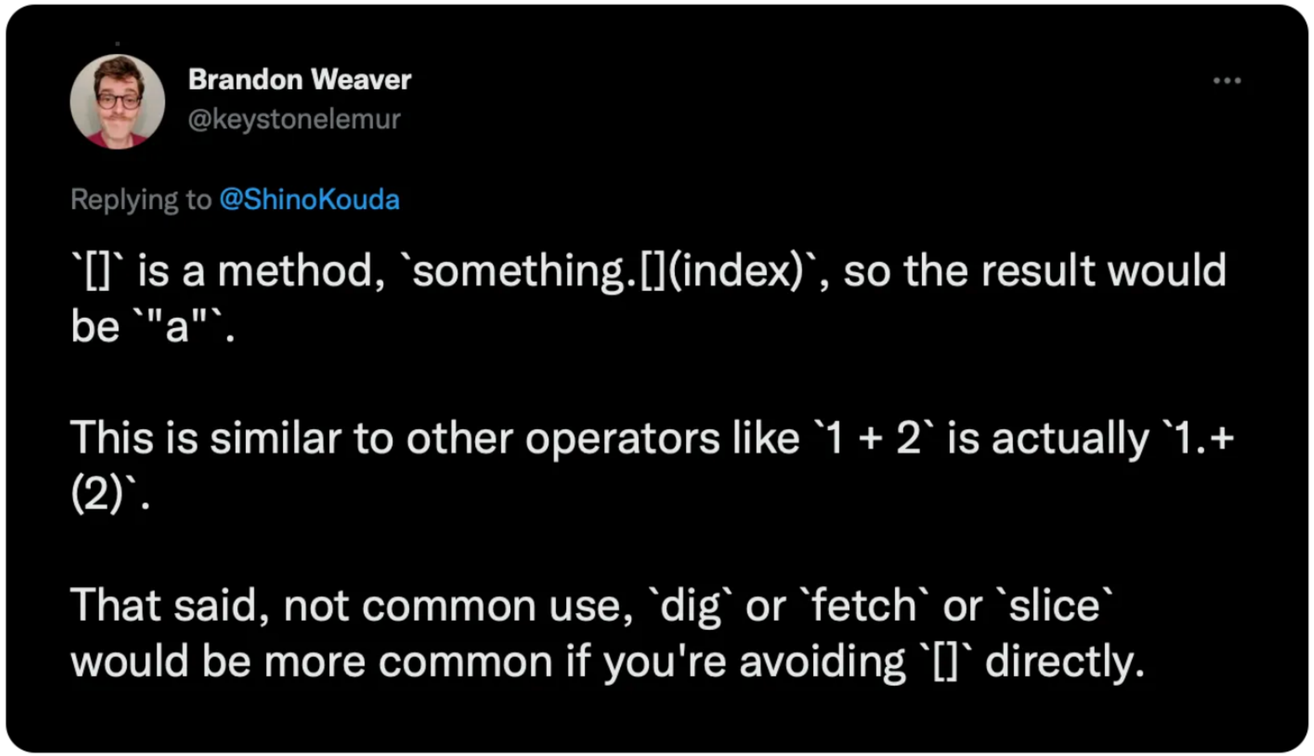


Source: @fxn on Twitter

Shino Kouda asked what is the result of executing:

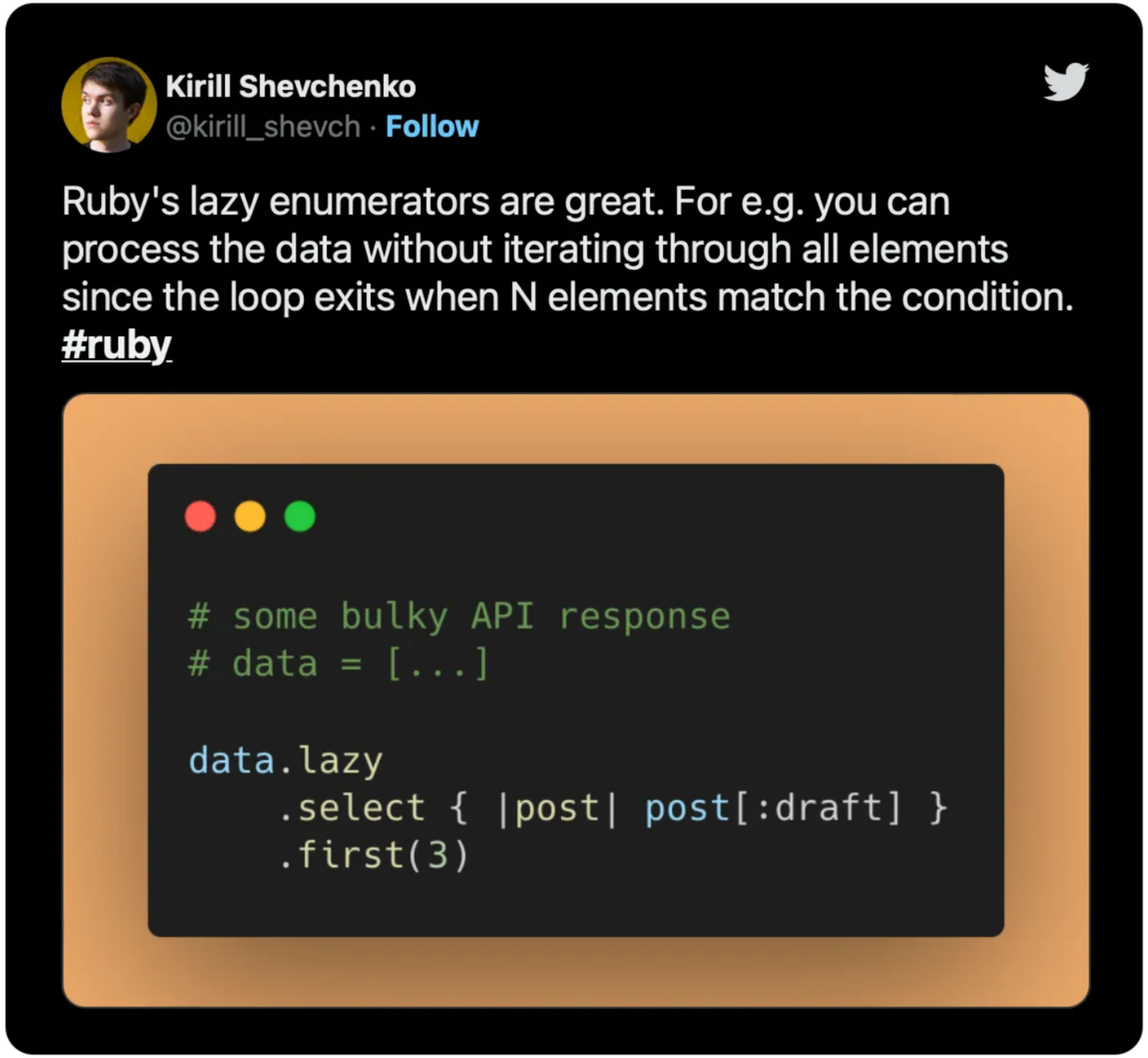
```
["a","b","c","d"].[](0)
```

Brandon Weaver shared explaining how [] works for arrays and hashes:



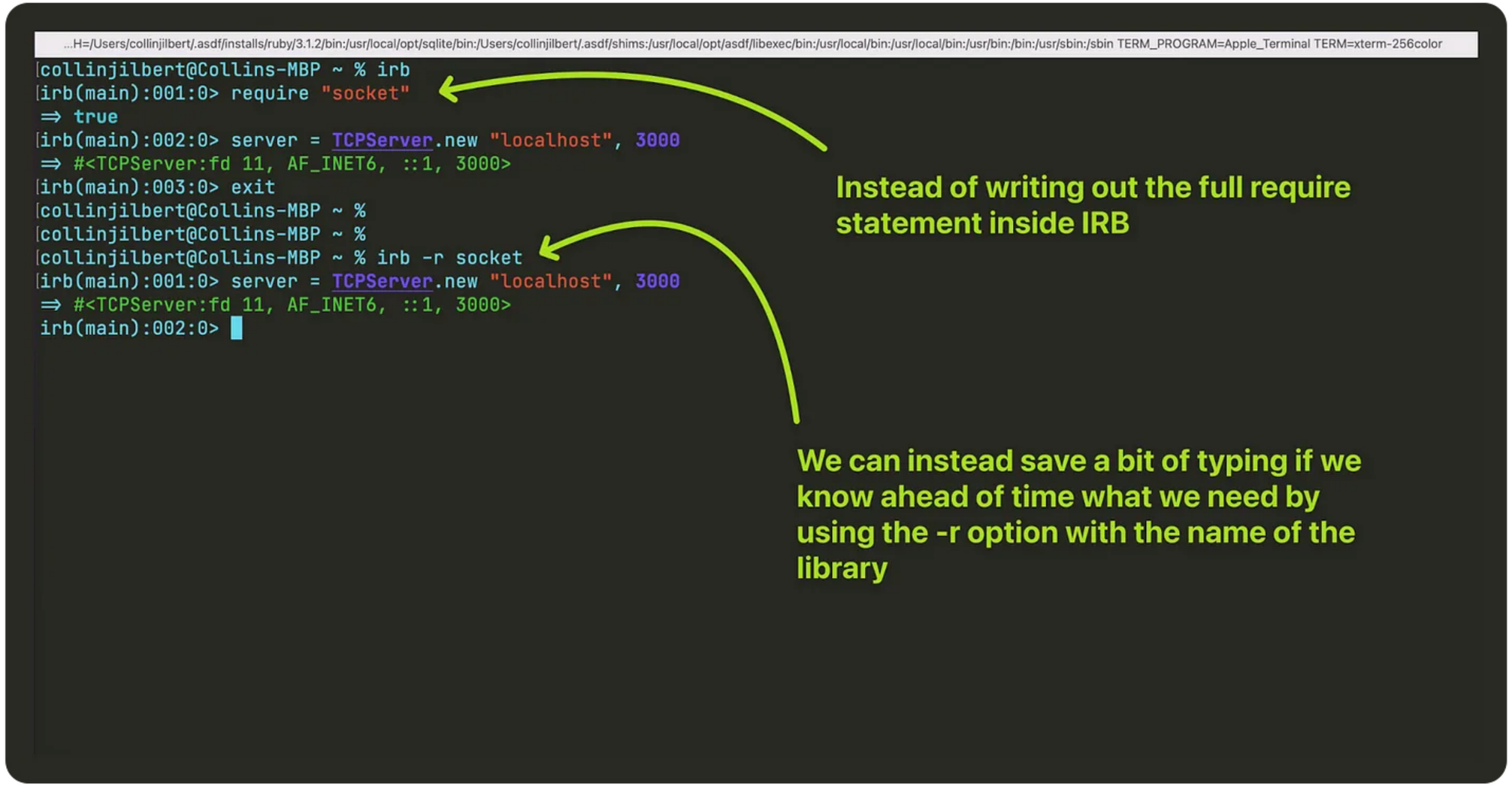
Source: @keystonelemur on Twitter

Kirill Shevchenko shared code sample about how to use lazy enumerators:



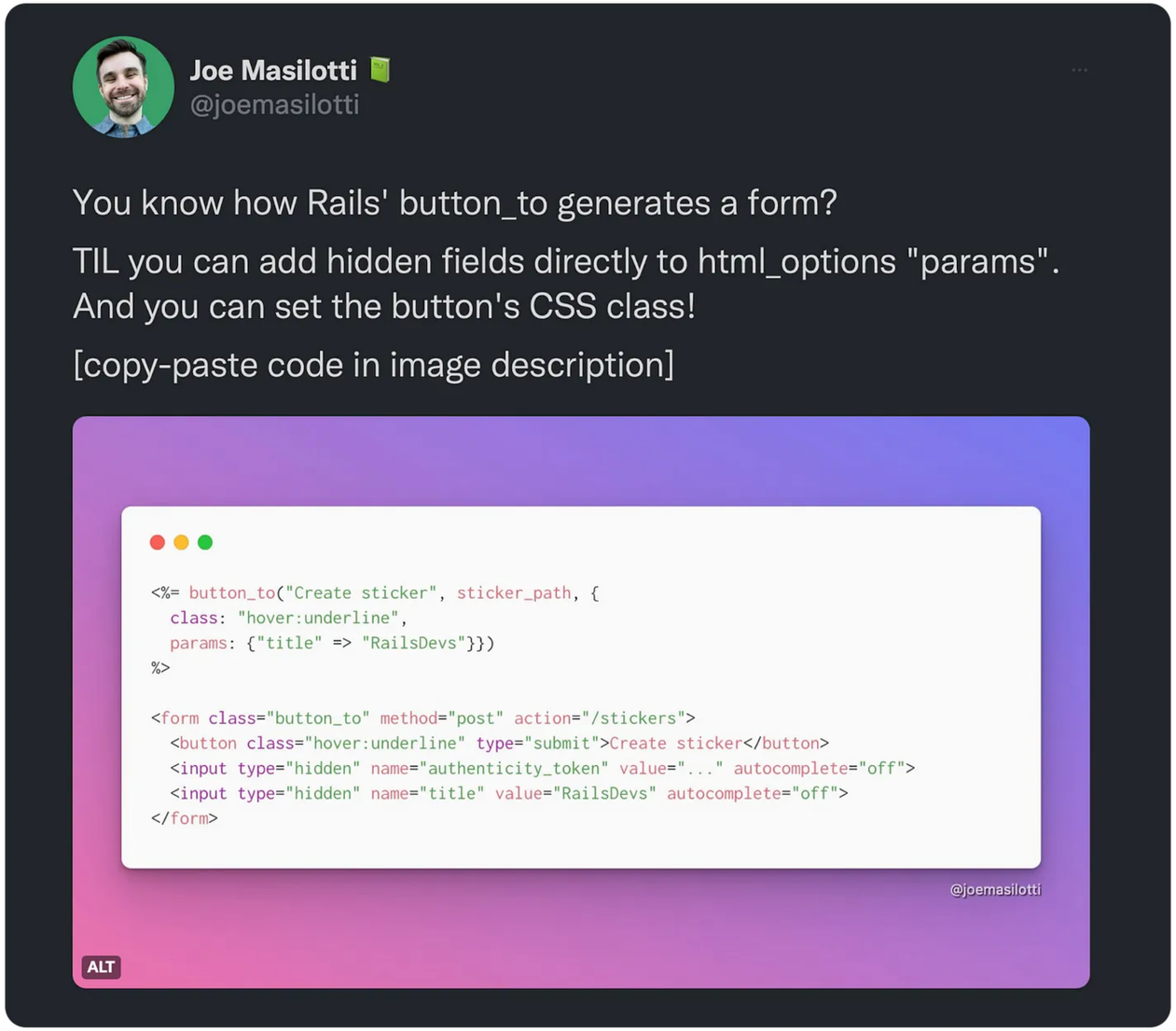
Source: @kirill_shevch on Twitter

Collin shared a tip about how to require a library when starting IRB:



Source: @collin_jilbert on Twitter

Joe Masilotti shared a sample code showing how to add hidden field to the button_to helper:



Source: @joemasilotti on Twitter

If you want to read the entire newsletter please go to

https://newsletter.shortruby.com