IIC2513 - TECNOLOGÍAS Y APLICACIONES WEB

I-2016

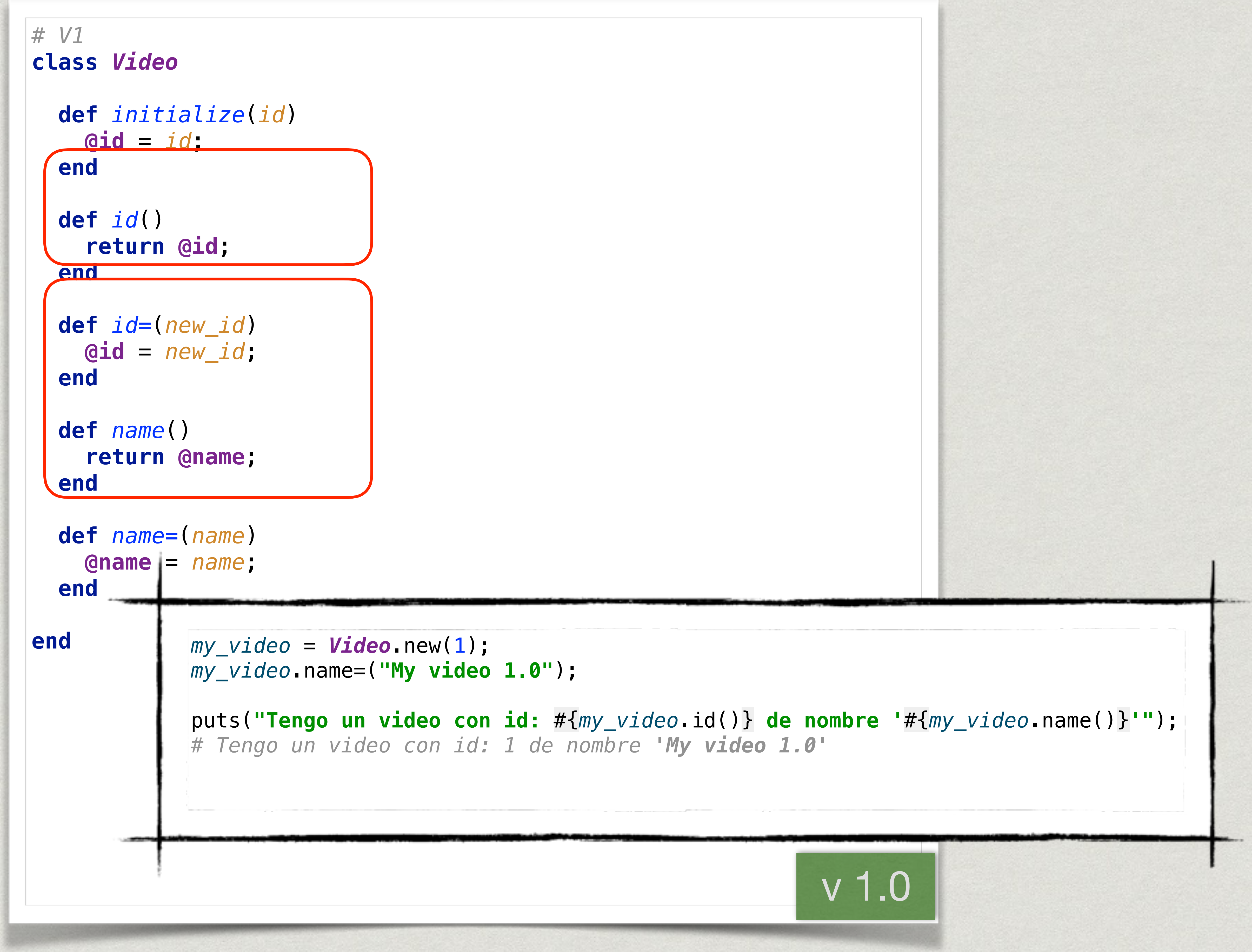
RUBY

Lenguaje de programación

Interpretado (no compilado)

Orientado a objetos

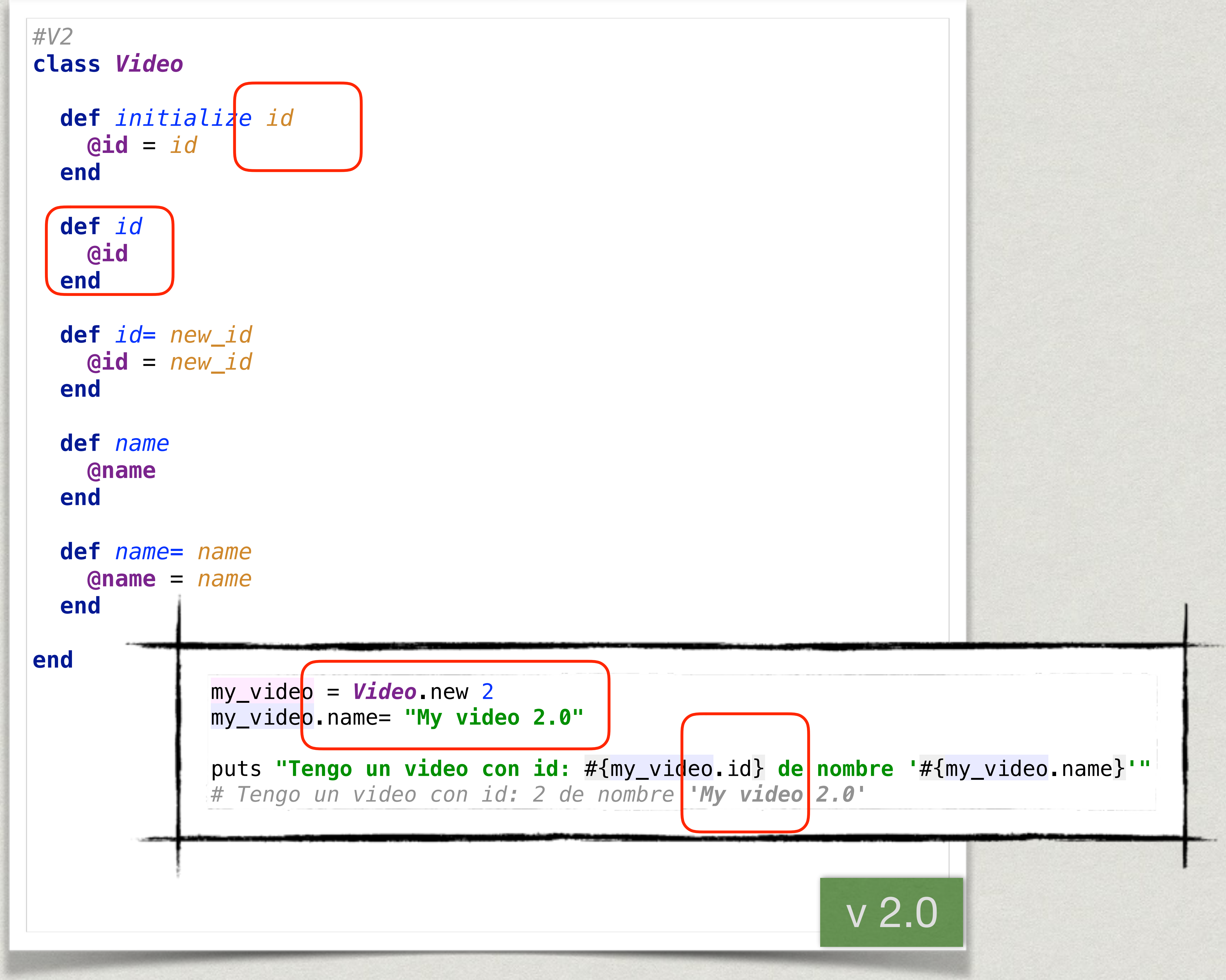
ANATOMÍA DE UNA CLASE



```ruby
# V1
class Video

  def initialize(id)
    @id = id;
  end

  def id()
    return @id;
  end

  def id=(new_id)
    @id = new_id;
  end

  def name()
    return @name;
  end

  def name=(name)
    @name = name;
  end

end
```

```ruby
my_video = Video.new(1);
my_video.name=("My video 1.0");

puts("Tengo un video con id: #{my_video.id()} de nombre '#{my_video.name()}'");
# Tengo un video con id: 1 de nombre 'My video 1.0'
```

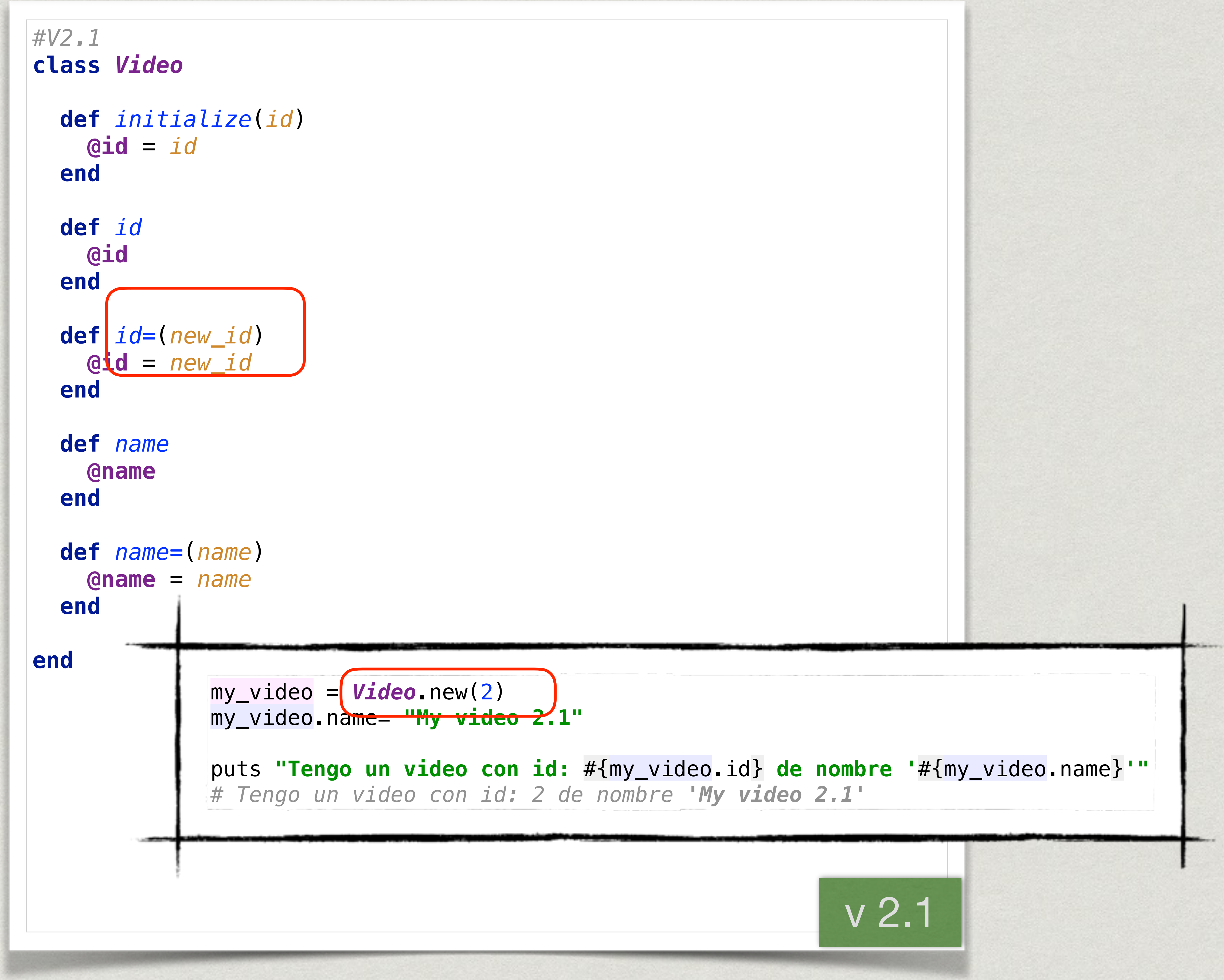
v 1.0



```ruby
#V2
class Video

  def initialize id
    @id = id
  end

  def id
    @id
  end

  def id= new_id
    @id = new_id
  end

  def name
    @name
  end

  def name= name
    @name = name
  end

end
```

```ruby
my_video = Video.new 2
my_video.name= "My video 2.0"

puts "Tengo un video con id: #{my_video.id} de nombre '#{my_video.name}'"
# Tengo un video con id: 2 de nombre 'My video 2.0'
```

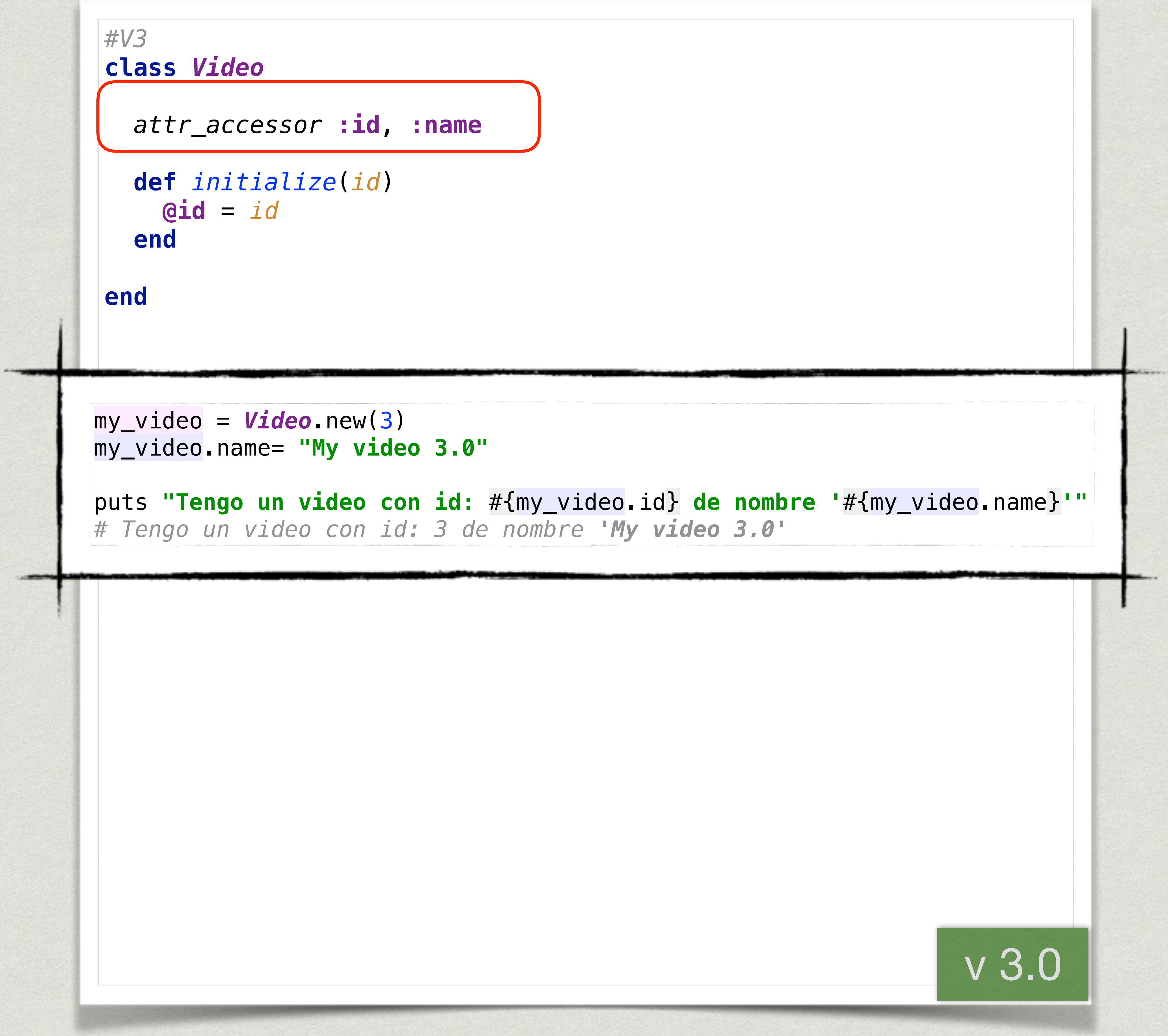
v 2.0



```ruby
#V2.1
class Video

  def initialize(id)
    @id = id
  end

  def id
    @id
  end

  def id=(new_id)
    @id = new_id
  end

  def name
    @name
  end

  def name=(name)
    @name = name
  end

end
```

```ruby
my_video = Video.new(2)
my_video.name= "My video 2.1"

puts "Tengo un video con id: #{my_video.id} de nombre '#{my_video.name}'"
# Tengo un video con id: 2 de nombre 'My video 2.1'
```

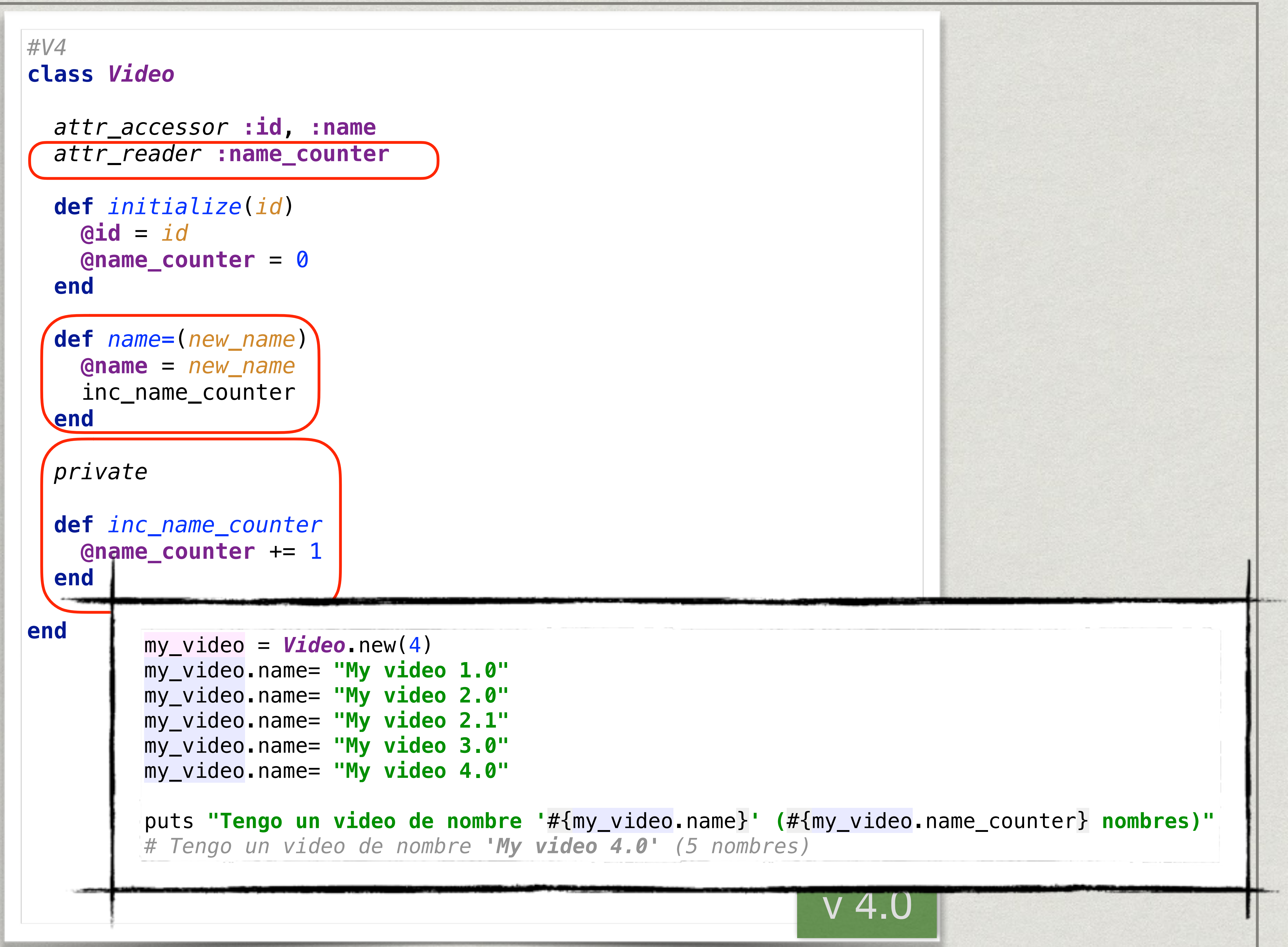
v 2.1



```
#V3
class Video

  attr_accessor :id, :name

  def initialize(id)
    @id = id
  end

end
```

```
my_video = Video.new(3)
my_video.name= "My video 3.0"

puts "Tengo un video con id: #{my_video.id} de nombre '#{my_video.name}'"
# Tengo un video con id: 3 de nombre 'My video 3.0'
```

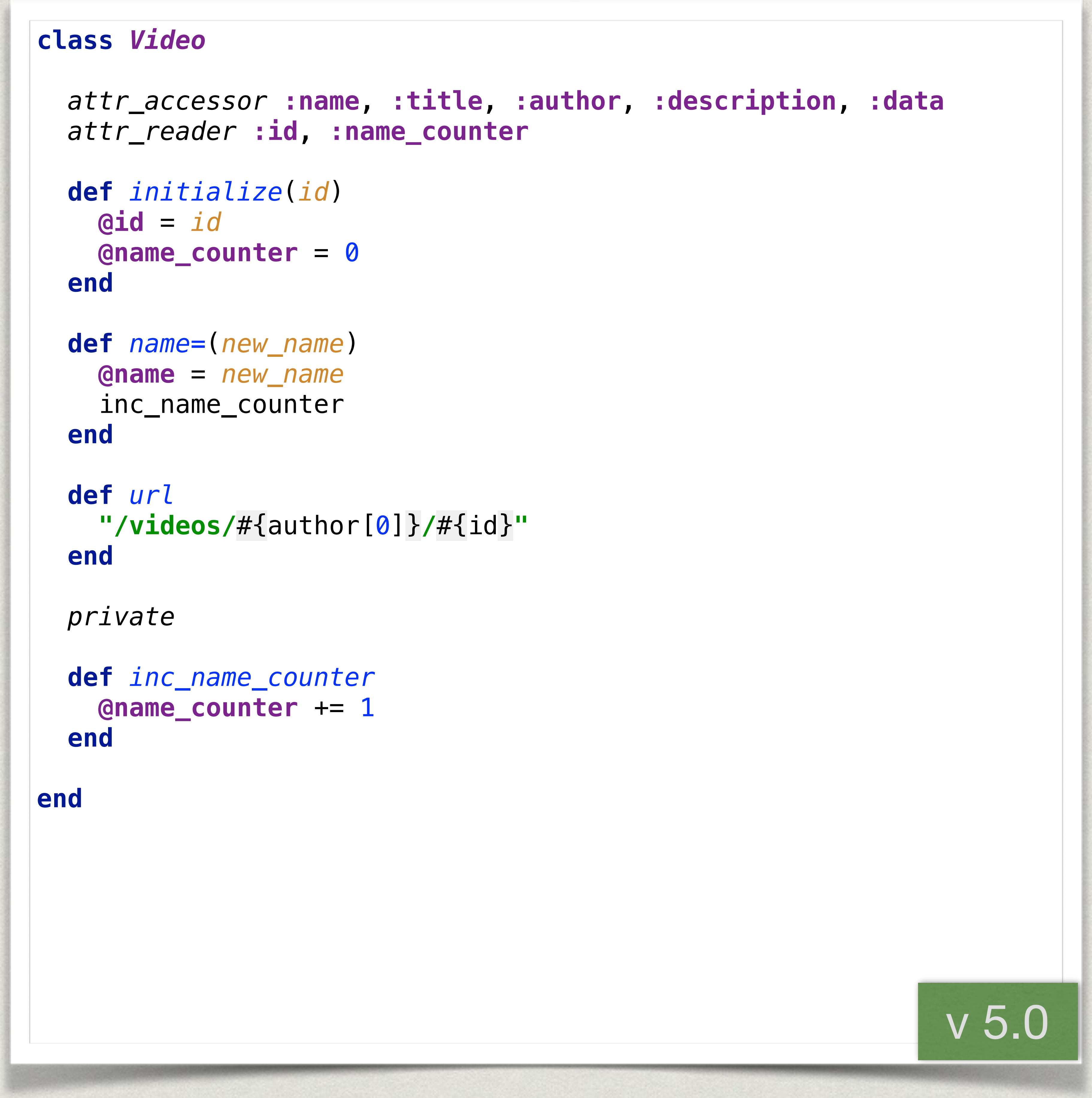
v 3.0



```ruby
#V4
class Video

  attr_accessor :id, :name
  attr_reader :name_counter

  def initialize(id)
    @id = id
    @name_counter = 0
  end

  def name=(new_name)
    @name = new_name
    inc_name_counter
  end

  private

  def inc_name_counter
    @name_counter += 1
  end

end
```

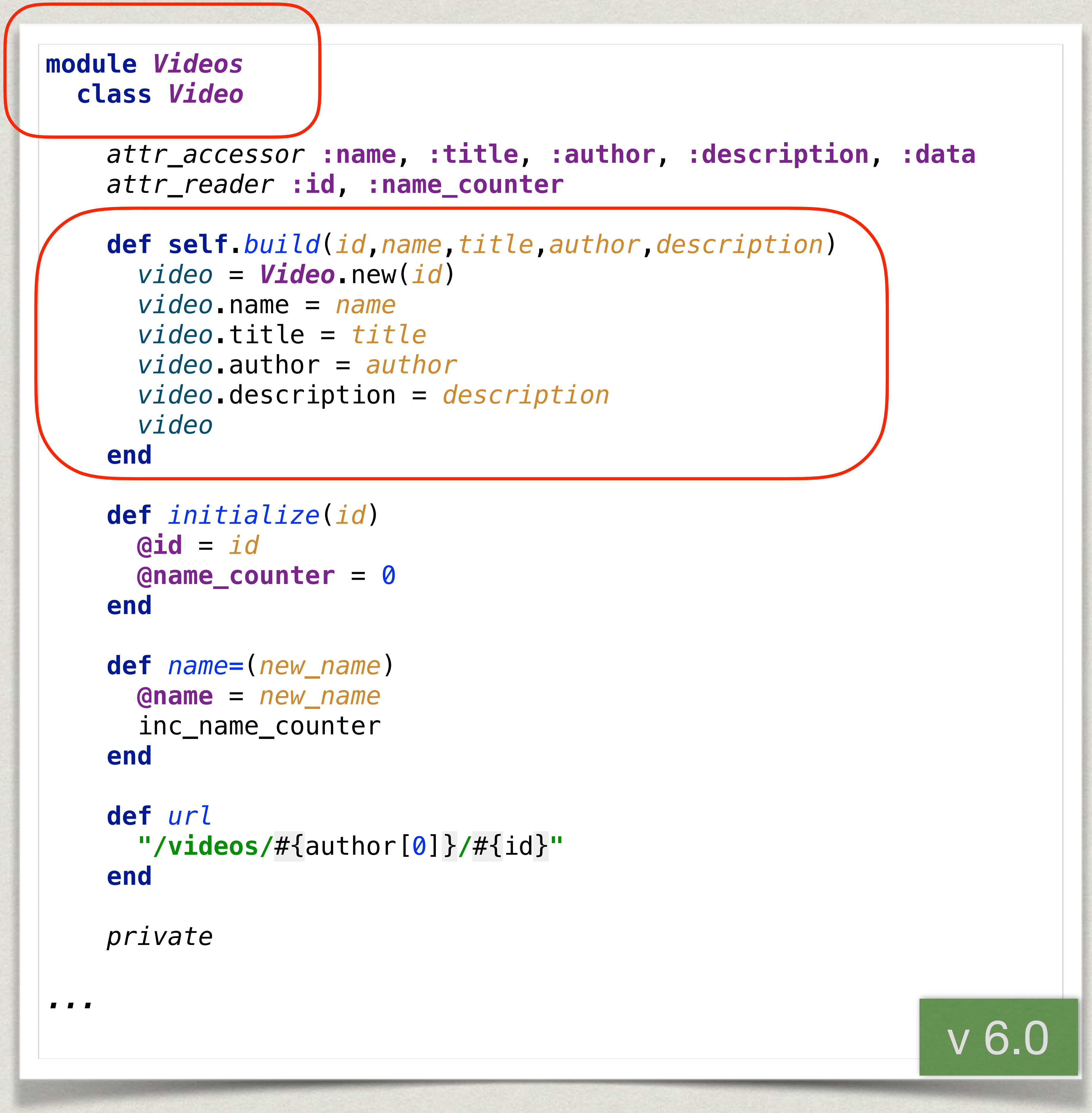
```ruby
my_video = Video.new(4)
my_video.name= "My video 1.0"
my_video.name= "My video 2.0"
my_video.name= "My video 2.1"
my_video.name= "My video 3.0"
my_video.name= "My video 4.0"

puts "Tengo un video de nombre '#{my_video.name}' (#{my_video.name_counter} nombres)"
# Tengo un video de nombre 'My video 4.0' (5 nombres)
```

v 4.0



```
class Video

  attr_accessor :name, :title, :author, :description, :data
  attr_reader :id, :name_counter

  def initialize(id)
    @id = id
    @name_counter = 0
  end

  def name=(new_name)
    @name = new_name
    inc_name_counter
  end

  def url
    "/videos/#{author[0]}/#{id}"
  end

  private

  def inc_name_counter
    @name_counter += 1
  end

end
```

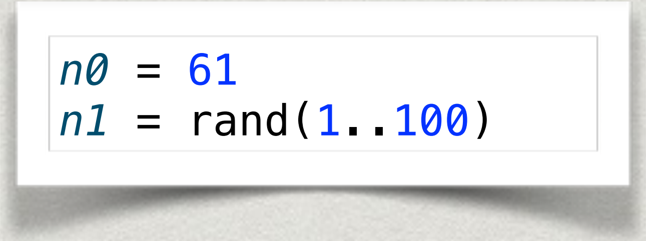
v 5.0



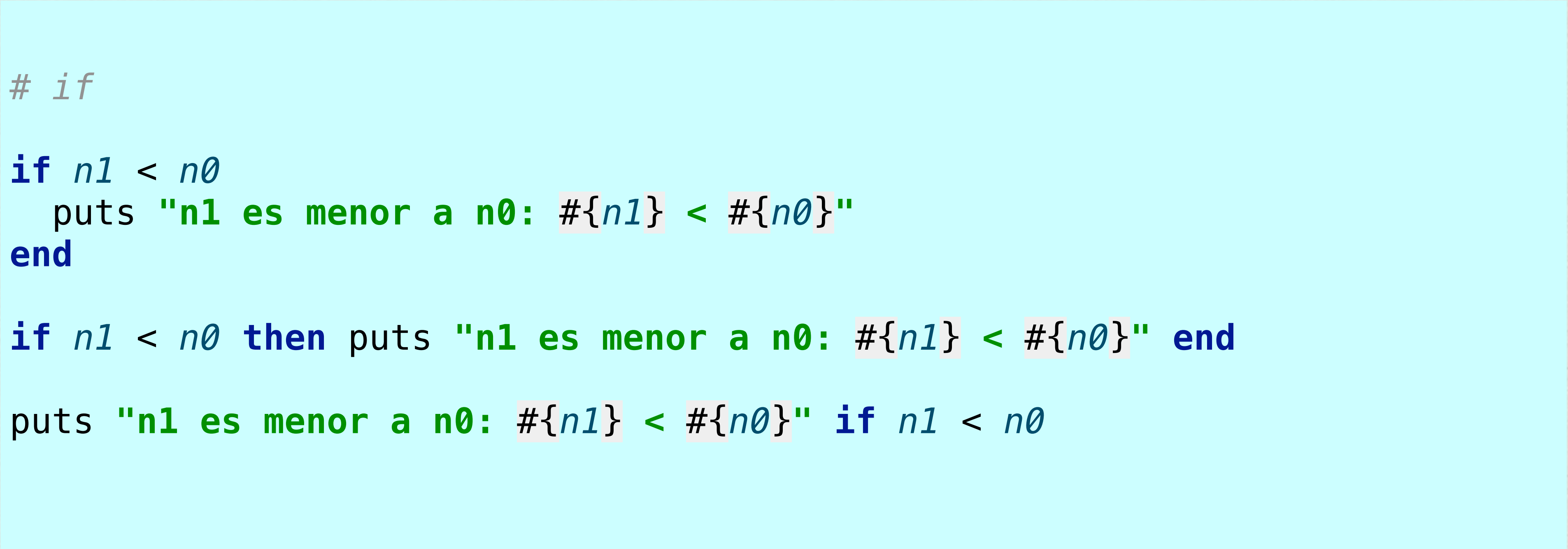
```ruby
module Videos
  class Video

    attr_accessor :name, :title, :author, :description, :data
    attr_reader :id, :name_counter

    def self.build(id,name,title,author,description)
      video = Video.new(id)
      video.name = name
      video.title = title
      video.author = author
      video.description = description
      video
    end

    def initialize(id)
      @id = id
      @name_counter = 0
    end

    def name=(new_name)
      @name = new_name
      inc_name_counter
    end

    def url
      "/videos/#{author[0]}/#{id}"
    end

    private

...
```

v 6.0

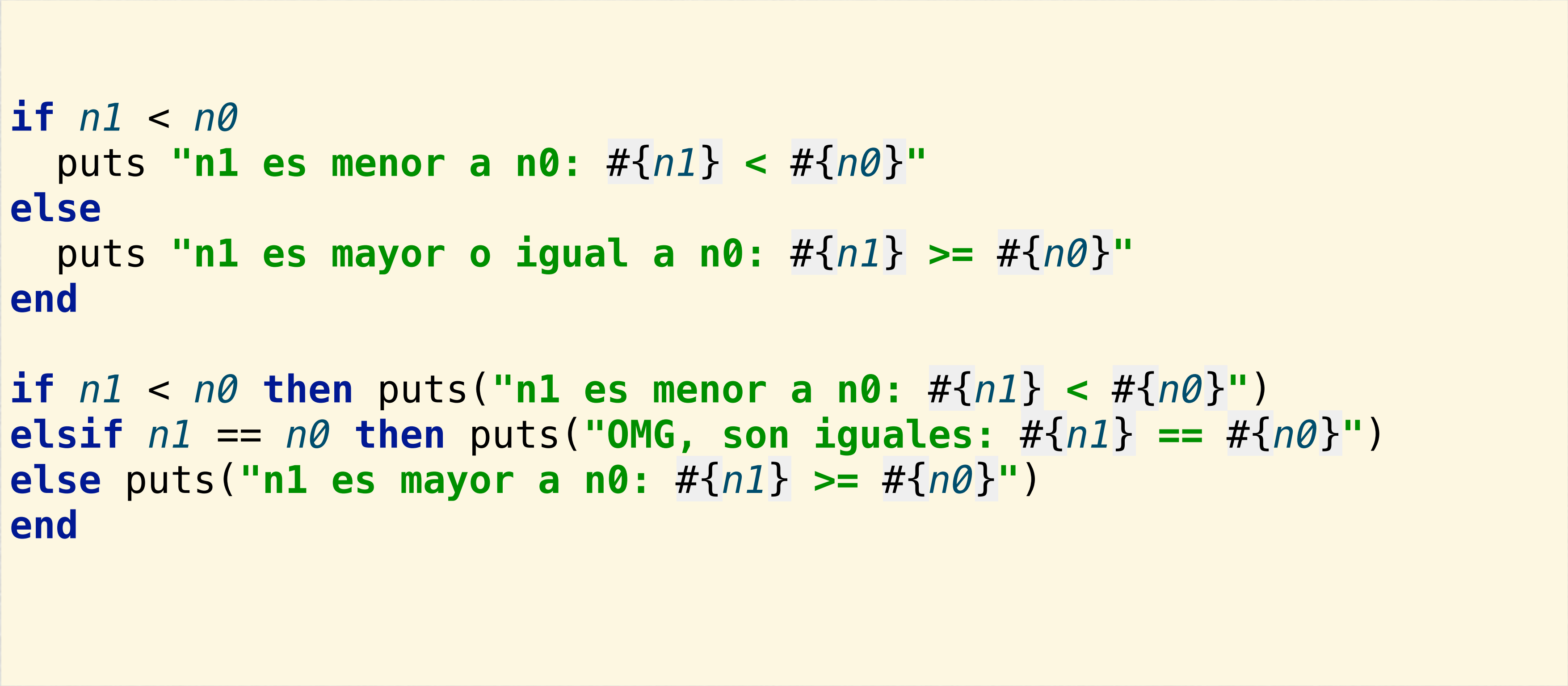
ALGO DE SINTAXIS



```ruby
n0 = 61
n1 = rand(1..100)
```

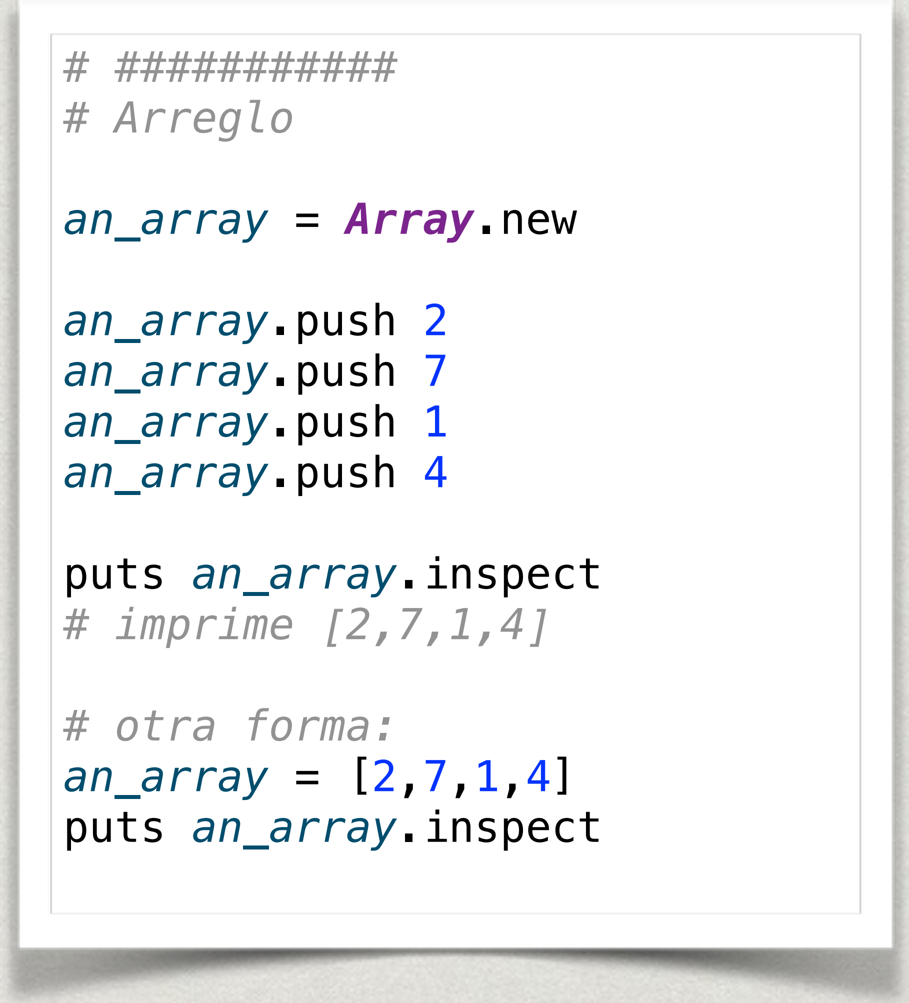


```ruby
# if

if n1 < n0
  puts "n1 es menor a n0: #{n1} < #{n0}"
end

if n1 < n0 then puts "n1 es menor a n0: #{n1} < #{n0}" end

puts "n1 es menor a n0: #{n1} < #{n0}" if n1 < n0
```

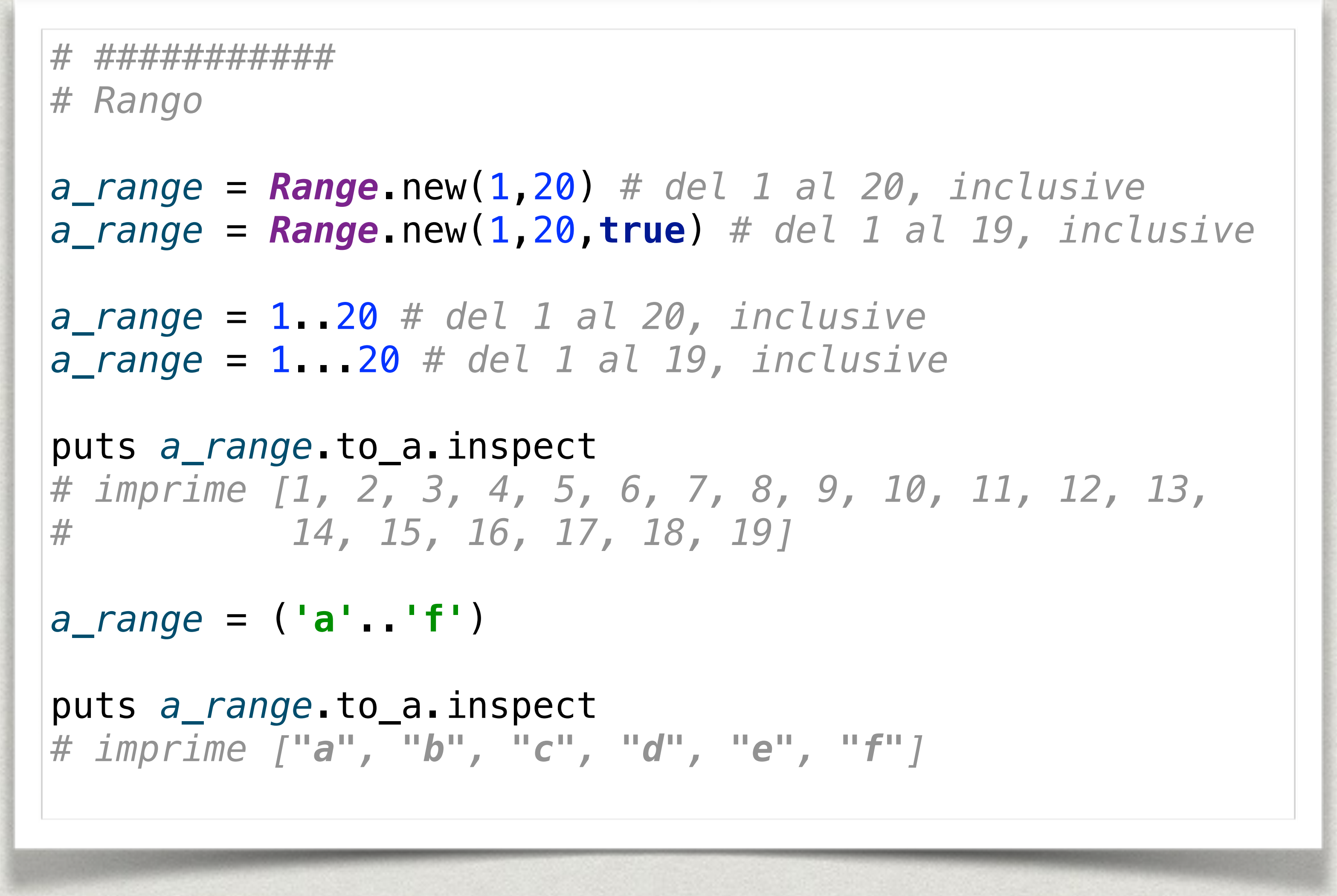


```ruby
if n1 < n0
  puts "n1 es menor a n0: #{n1} < #{n0}"
else
  puts "n1 es mayor o igual a n0: #{n1} >= #{n0}"
end

if n1 < n0 then puts("n1 es menor a n0: #{n1} < #{n0}")
elsif n1 == n0 then puts("OMG, son iguales: #{n1} == #{n0}")
else puts("n1 es mayor a n0: #{n1} >= #{n0}")
end
```

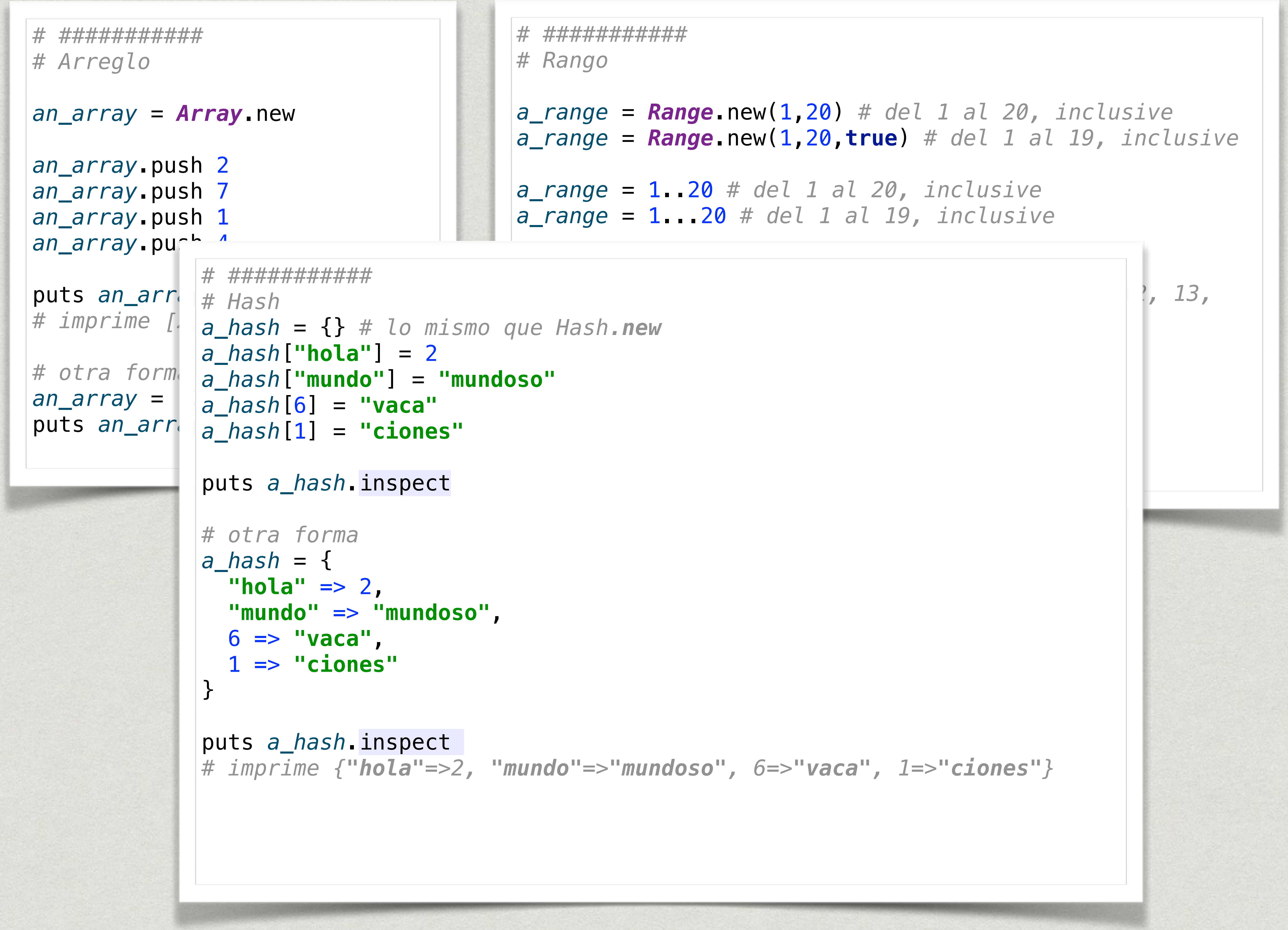
ALGUNAS ESTRUCTURAS



```
# ###########
# Arreglo

an_array = Array.new

an_array.push 2
an_array.push 7
an_array.push 1
an_array.push 4

puts an_array.inspect
# imprime [2,7,1,4]

# otra forma:
an_array = [2,7,1,4]
puts an_array.inspect
```



```
# ###########
# Rango

a_range = Range.new(1,20) # del 1 al 20, inclusive
a_range = Range.new(1,20,true) # del 1 al 19, inclusive

a_range = 1..20 # del 1 al 20, inclusive
a_range = 1...20 # del 1 al 19, inclusive

puts a_range.to_a.inspect
# imprime [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13,
#           14, 15, 16, 17, 18, 19]

a_range = ('a'..'f')

puts a_range.to_a.inspect
# imprime ["a", "b", "c", "d", "e", "f"]
```

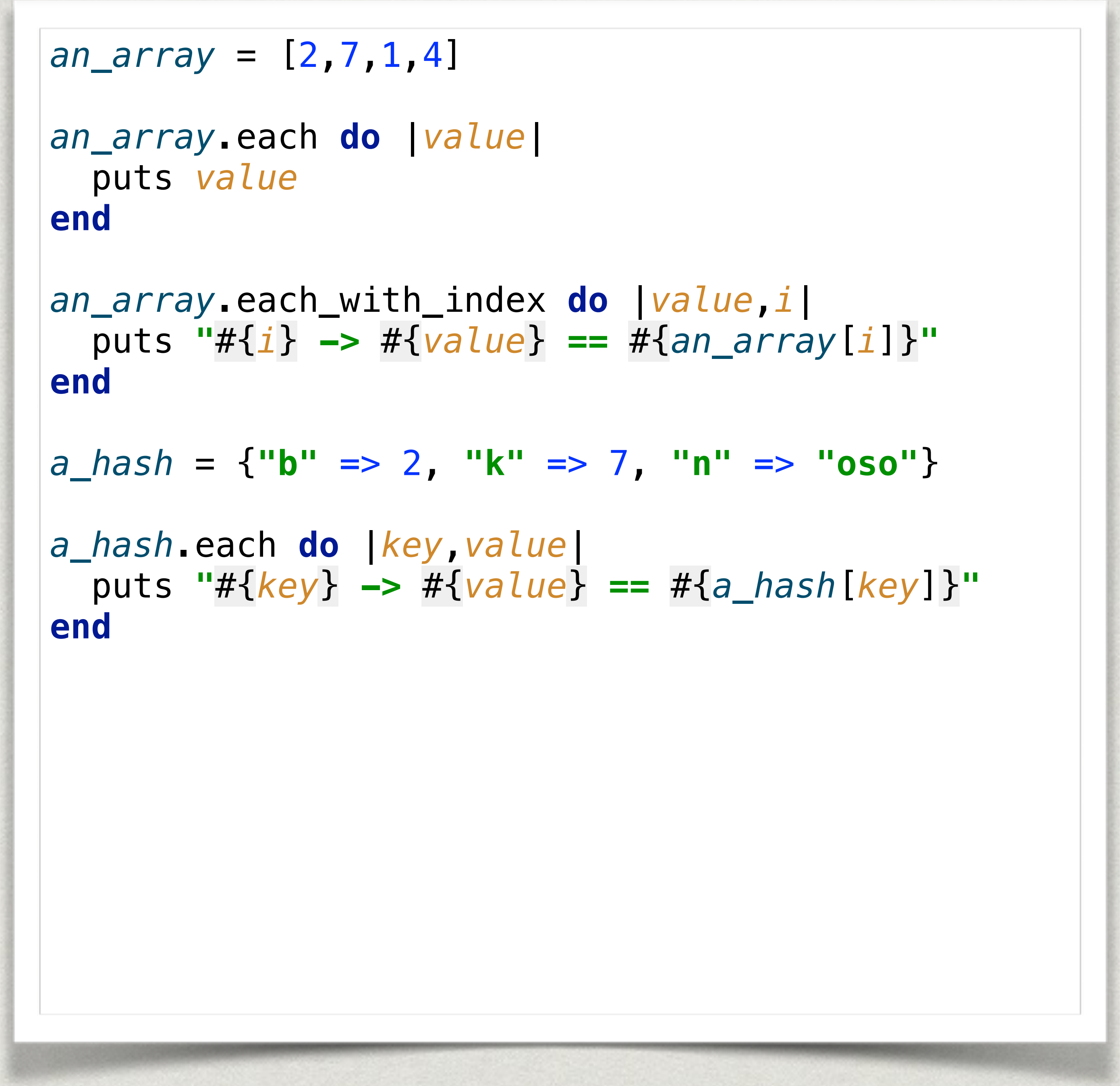


```
# ###########
# Arreglo

an_array = Array.new

an_array.push 2
an_array.push 7
an_array.push 1
an_array.push 4

puts an_arr
# imprime [

# otra forma
an_array =
puts an_arr
```

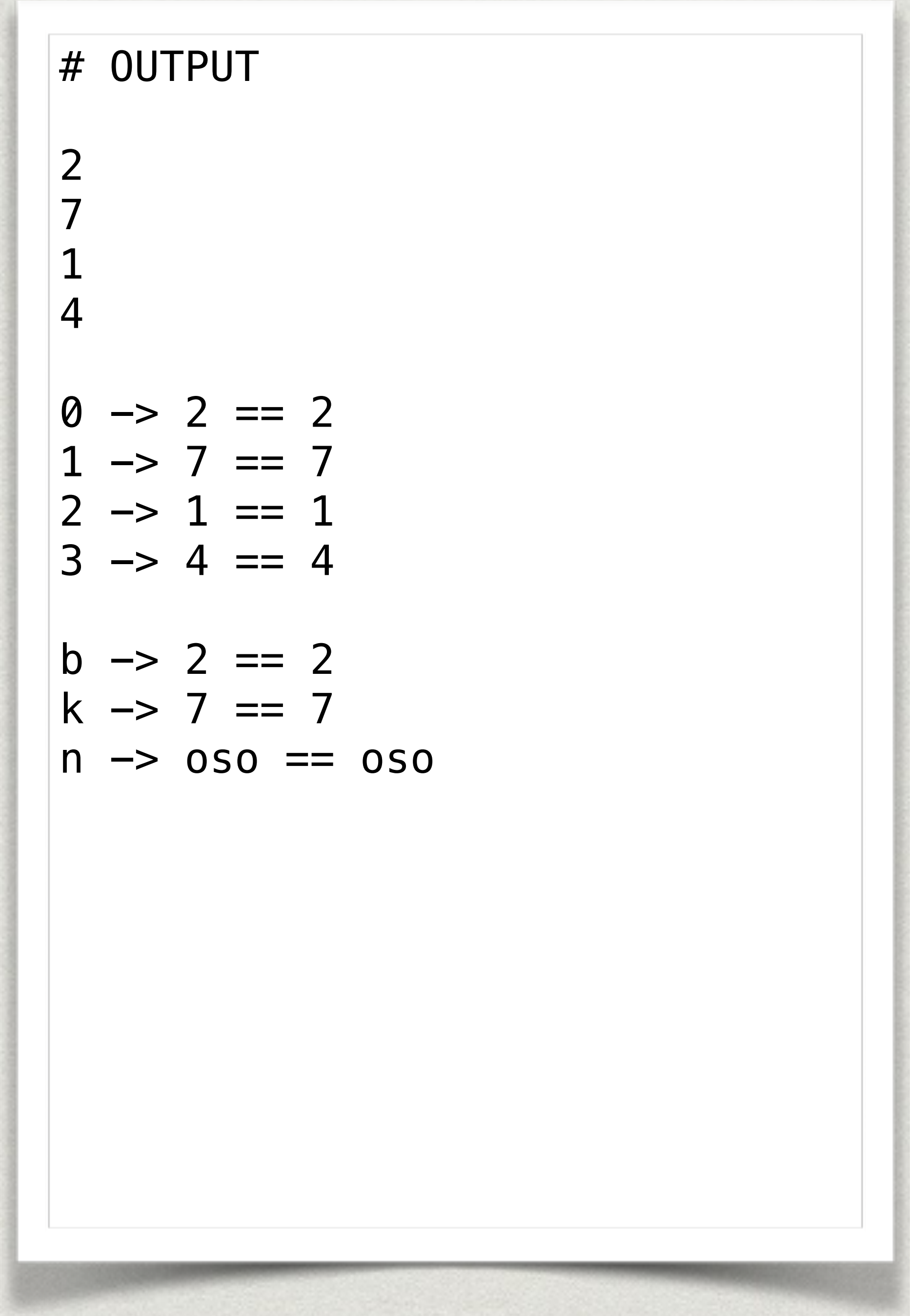
```
# ###########
# Rango

a_range = Range.new(1,20) # del 1 al 20, inclusive
a_range = Range.new(1,20,true) # del 1 al 19, inclusive

a_range = 1..20 # del 1 al 20, inclusive
a_range = 1...20 # del 1 al 19, inclusive

, 13,
```

```
# ###########
# Hash
a_hash = {} # lo mismo que Hash.new
a_hash["hola"] = 2
a_hash["mundo"] = "mundoso"
a_hash[6] = "vaca"
a_hash[1] = "ciones"

puts a_hash.inspect

# otra forma
a_hash = {
  "hola" => 2,
  "mundo" => "mundoso",
  6 => "vaca",
  1 => "ciones"
}

puts a_hash.inspect
# imprime {"hola"=>2, "mundo"=>"mundoso", 6=>"vaca", 1=>"ciones"}
```

ALGO MÁS DE SINTAXIS



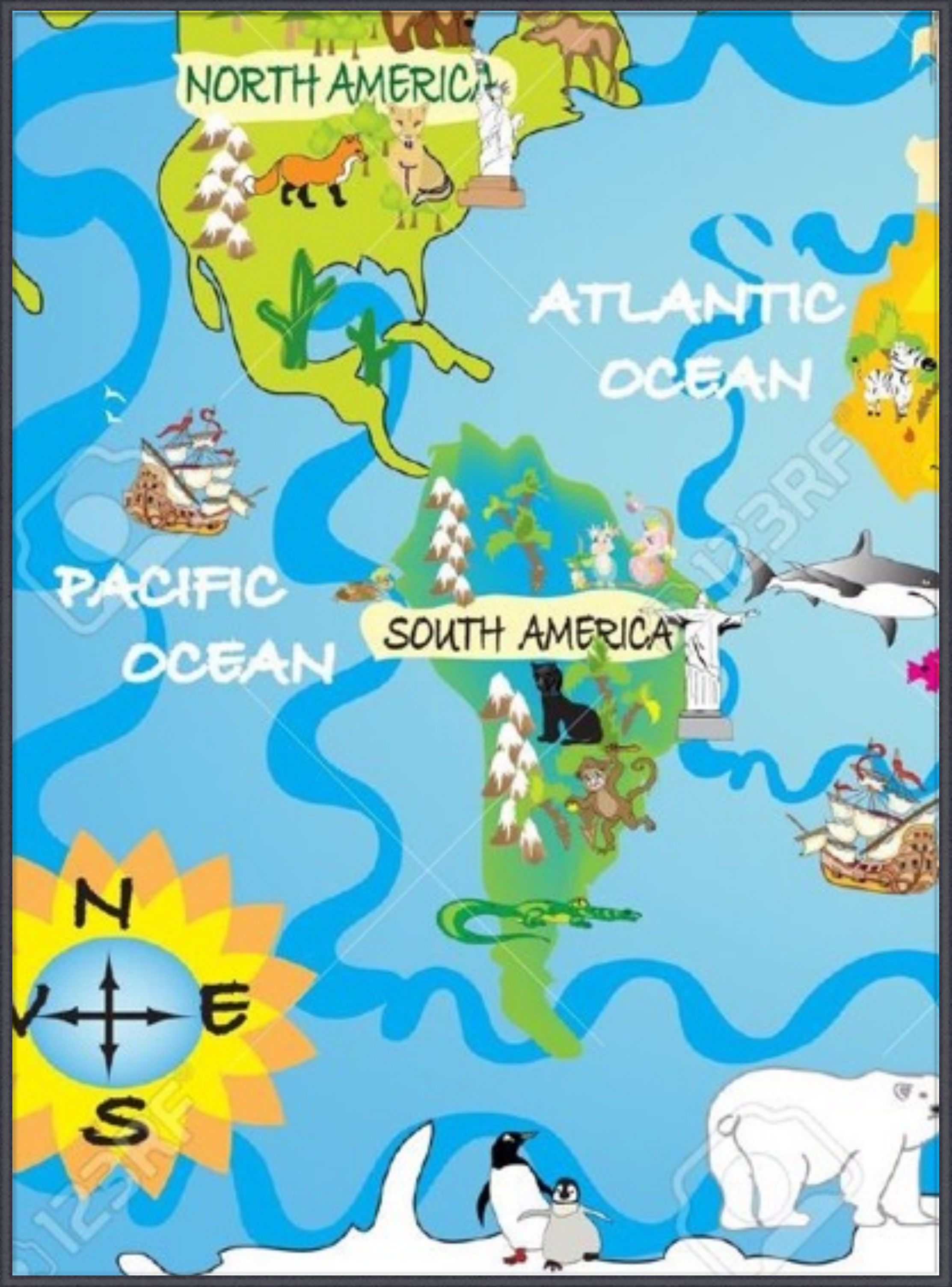
```ruby
an_array = [2,7,1,4]

an_array.each do |value|
  puts value
end

an_array.each_with_index do |value,i|
  puts "#{i} -> #{value} == #{an_array[i]}"
end

a_hash = {"b" => 2, "k" => 7, "n" => "oso"}

a_hash.each do |key,value|
  puts "#{key} -> #{value} == #{a_hash[key]}"
end
```

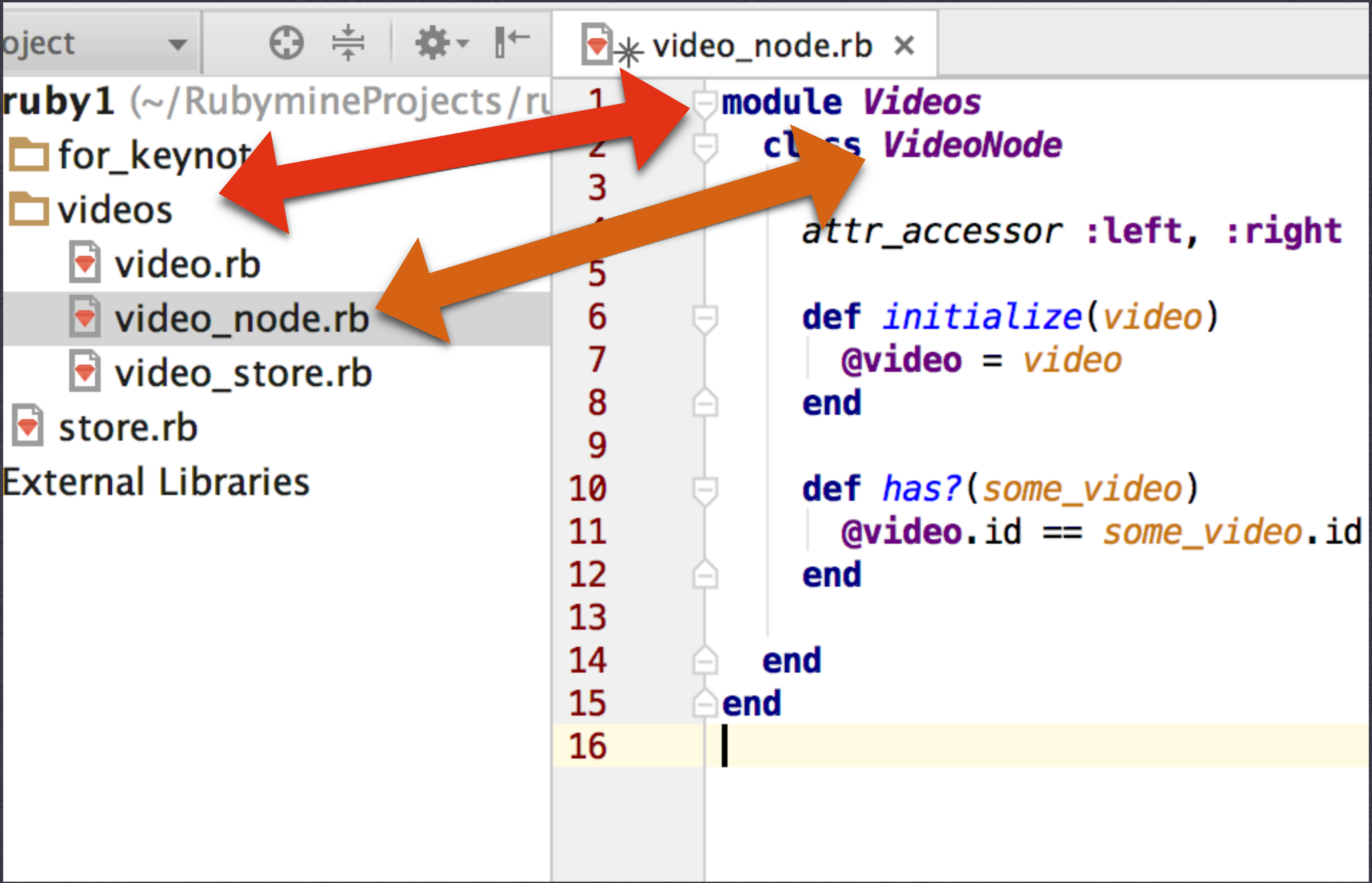


```
# OUTPUT

2
7
1
4

0 -> 2 == 2
1 -> 7 == 7
2 -> 1 == 1
3 -> 4 == 4

b -> 2 == 2
k -> 7 == 7
n -> oso == oso
```

ARCHIVOS

Y módulos y carpetas y require y otras cosas

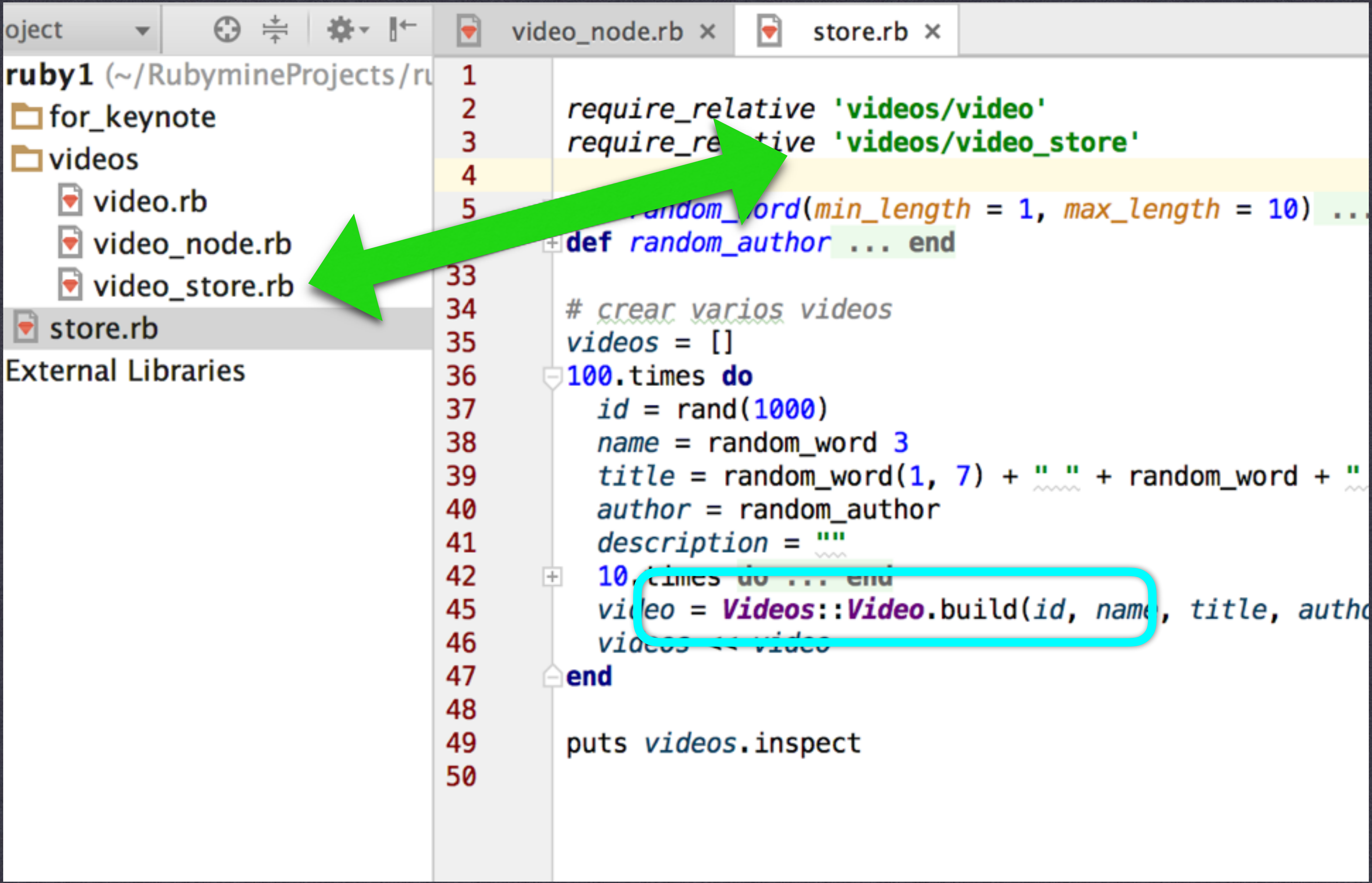




ruby1 (~/RubymineProjects/ru
- for_keynot
- videos
  - video.rb
  - video_node.rb
  - video_store.rb
- store.rb

External Libraries

```ruby
module Videos
  class VideoNode

    attr_accessor :left, :right

    def initialize(video)
      @video = video
    end

    def has?(some_video)
      @video.id == some_video.id
    end

  end
end

```

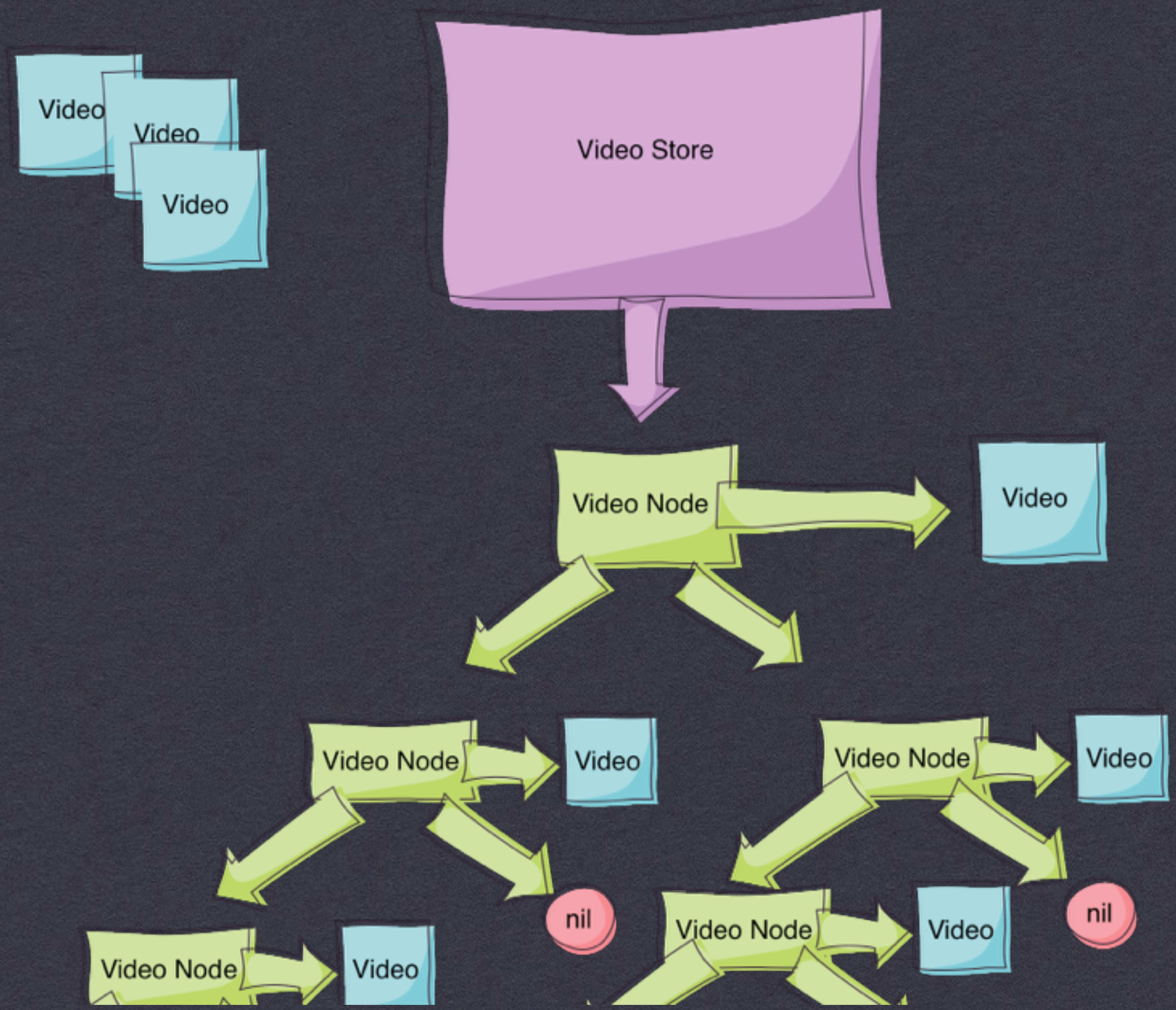


```

require_relative 'videos/video'
require_relative 'videos/video_store'

def random_word(min_length = 1, max_length = 10) ...
def random_author ... end

# crear varios videos
videos = []
100.times do
  id = rand(1000)
  name = random_word 3
  title = random_word(1, 7) + " " + random_word + "
  author = random_author
  description = ""
  10.times do ... end
  video = Videos::Video.build(id, name, title, autho
  videos << video
end

puts videos.inspect

```

Referencias

http://ruby-doc.org/

EJERCICIO


Video
Video
Video
Video Store
Video Node
Video
Video Node
Video
Video Node
Video
nil
nil
Video Node
Video
Video Node
Video

EJERCICIO

IIC2513 - TECNOLOGÍAS Y APLICACIONES WEB

I-2016