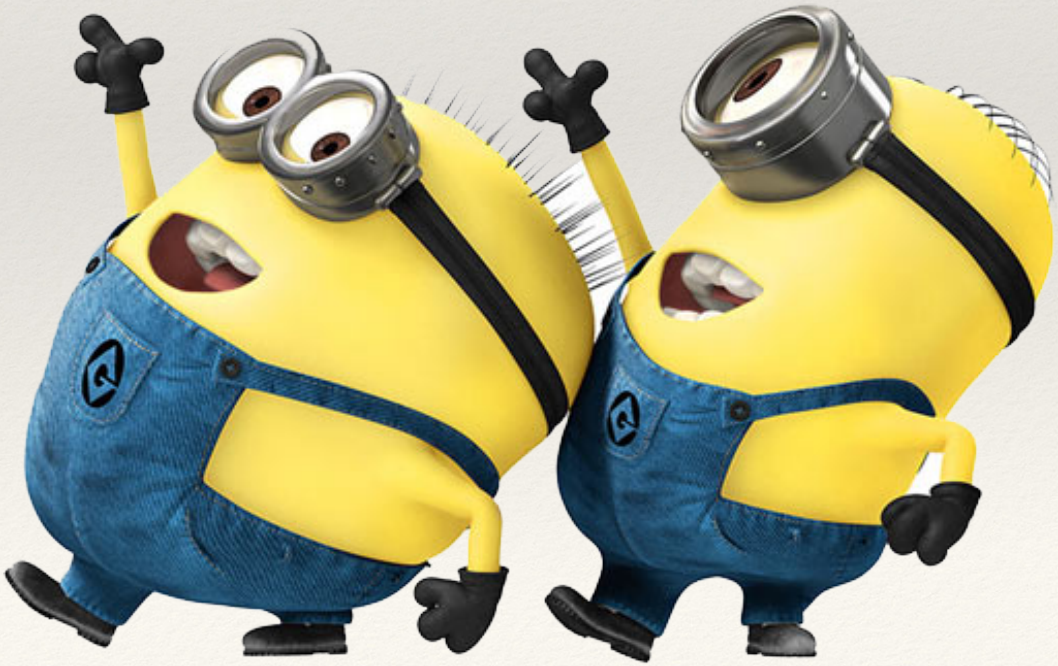
Binary Ipsum

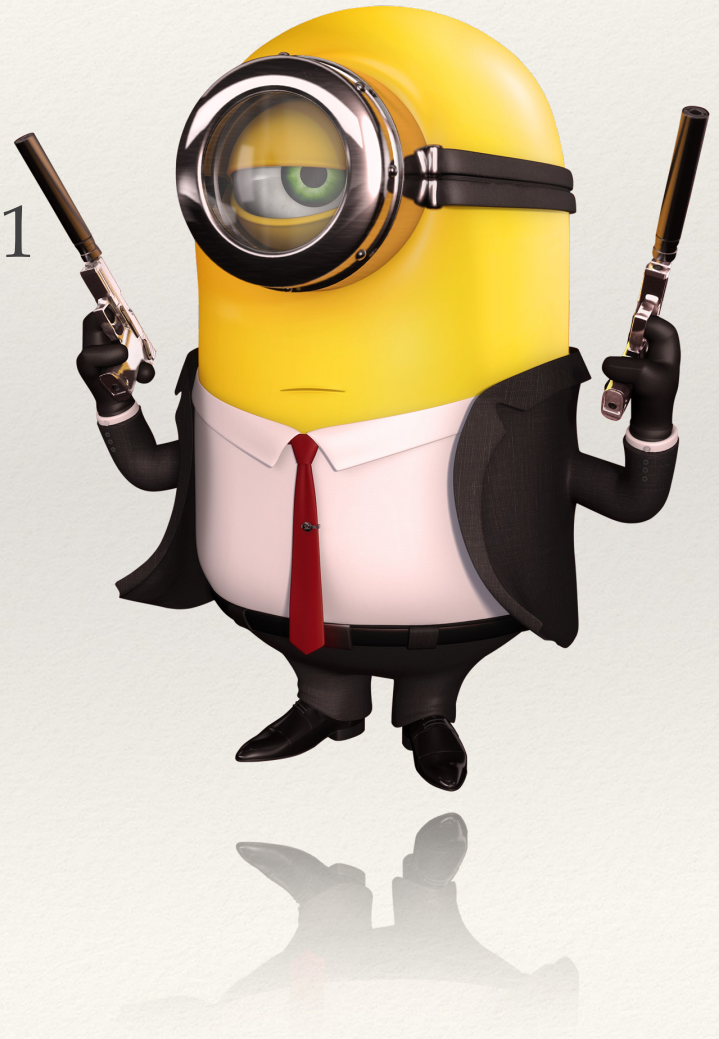
Goals

- Demonstrate a solution, via TDD, to solve a problem programmatically, using Ruby
- Learn something new
- Have some fun



Understand, you will

- 01010010 01110101 01100010 01111001
- 紅寶石
- Ruby



Example: Lorem Ipsum

- Doloribus rerum voluptatem molestias ut. Quis enim sapiente totam. Nihil qui consequuntur exercitationem iusto enim reiciendis. Optio et nihil quia voluptatibus minima.

Example: Binary Ipsum

- 01000100 01101111 01101100 01101111 01110010 01101001 01100010 01110101 01110011 00100000 01110010 01100101 01110010 01110101 01101101 00100000 01110110 01101111 01101100 01110101 01110000 01110100 01100001 01110100 01100101 01101101 00100000 01101101 01101111 01101100 01100101 01110011 01110100 01101001 01100001 01110011 00100000 01110101 01110100 00101110 00100000 01010001 01110101 01101001 01110011 00100000 01100101 01101110 01101001 01101101 00100000 01110011 01100001 01110000 01101001 01100101 01101110 01110100 01100101 00100000 01110100 01101111 01110100 01100001 01101101 00101110 00100000 01001110 01101001 01101000 01101001 01101100 00100000 01110001 01110101 01101001 00100000 01100011 01101111 01101110 01110011 01100101 01110001 01110101 01110101 01101110 01110100 01110101 01110010 00100000 01100101 01111000 01100101 01110010 01100011 01101001 01110100 01100001 01110100 01101001 01101111 01101110 01100101 01101101 00100000 01101001 01110101 01110011 01110100 01101111 00100000 01100101 01101110 01101001 01101101 00100000 01110010 01100101 01101001 01100011 01101001 01100101 01101110 01100100 01101001 01110011 00101110 00100000 01001111 01110000 01110100 01101001 01101111 00100000 01100101 01110100 00100000 01101110 01101001 01101000 01101001 01101100 00100000 01110001 01110101 01101001 01100001 00100000 01110110 01101111 01101100 01110101 01110000 01110100 01100001 01110100 01101001 01100010 01110101 01110011 00100000 01101101 01101001 01101110 01101001 01101101 01100001 00101110

Approximately what we want

- BinaryIpsum.new(string: “some string here”)
- -OR-
- BinaryIpsum.new(random: true, sentences: 7)

Backstory

Before moving forward I'd like to take a few minutes to discuss a possible approach to taking a Lorem Ipsum sentence and converting it to Binary. Most of us humans do not normally converse in binary. What exactly is binary? How does it relate to the task at hand?

In the simplest term, we visualize binary as either 0 or 1. Let's look at a word that is more familiar to us silly humans: Ruby

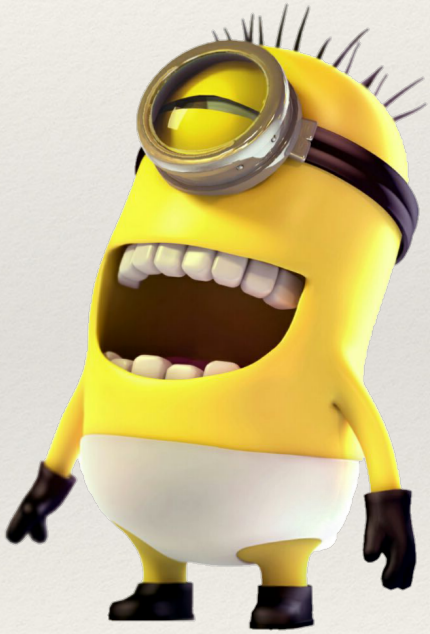
The word 'Ruby' is a collection/string of characters. Those characters being:

R, u, b and y

continued…

Behind the scenes, those letters have a decimal representation. For example:

- R = 82
- u = 117
- b = 98
- y = 121



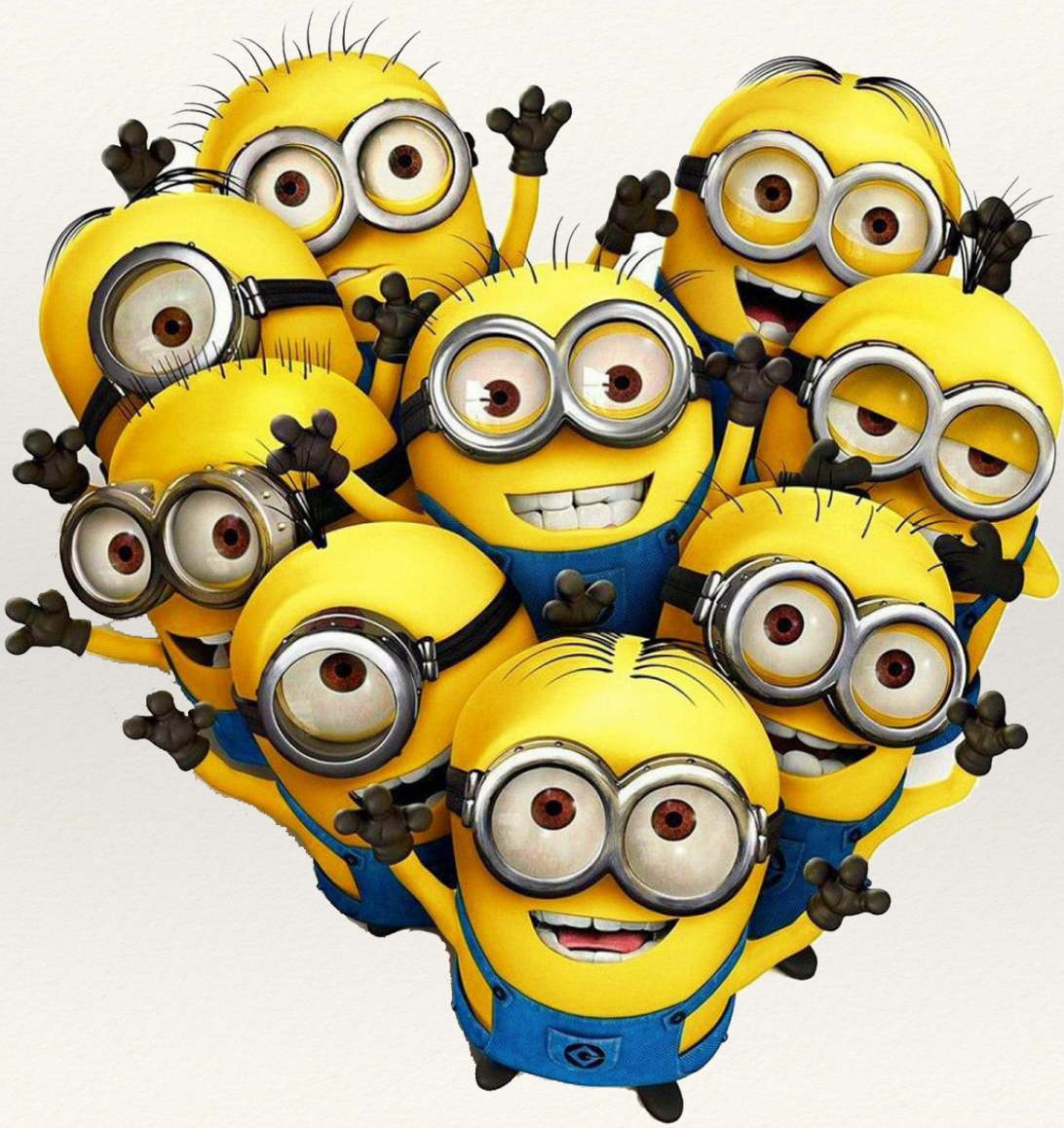
Converter

```
      128  64  32  16 |  8  4  2  1
0   =   0   0   0   0 |  0  0  0  0
255 =   1   1   1   1 |  1  1  1  1
```

“Failure is the first step towards success”

–Johnny Noclui

Time To Code



Revisit the Converter

```
      128  64  32  16  |  8  4  2  1
0   =   0   0   0   0  |  0  0  0  0
255 =   1   1   1   1  |  1  1  1  1
```

- R = 82 = 01010010
- u = 117 = 01110101
- b = 98 = 01100010
- y = 121 = 01111001

Play AsciiCasts

Our Solution

```
class BinaryIpsum
  require 'faker'

  attr_reader :lorem_string

  def initialize(string: 'Ruby', random: false, sentences: 5)
    return @lorem_string = Faker::Lorem.sentences(sentences).join(' ') if random
    @lorem_string = string
  end

  def to_char_codes
    lorem_string.chars.map { |ltr| ltr.ord }
  end

  def to_binary
    binary_sentence = self.to_char_codes.map { |int| int.to_s(2) }
    binary_sentence = binary_sentence.map { |binary_word| zero_pad(binary_word) }
    binary_sentence.join(' ')
  end

  def zero_pad(binary_value)
    binary_value.rjust(8,'0')
  end

end
```

Challenge

- Convert the binary for a ‘space’ character to a colon(:) or something similar.
- Make a command line tool based off of our solution, with text based help (--h)

Resources

- Binary Ipsum Repo - http://github.com/hogihung/binary_ipsum
- Vagrant VM for Binary Ipsum - http://github.com/hogihung/vagrant_binary_ipsum
- Keynote presentation - http://johnhogarty.com/binary_ipsum/BinaryIpsum.zip

In Closing...

- Twiter/Github: hogihung
- Blog: http://ognt.io -OR- http://oldguynewtrick.com
- Videos created using: http://asciinema.org

The End