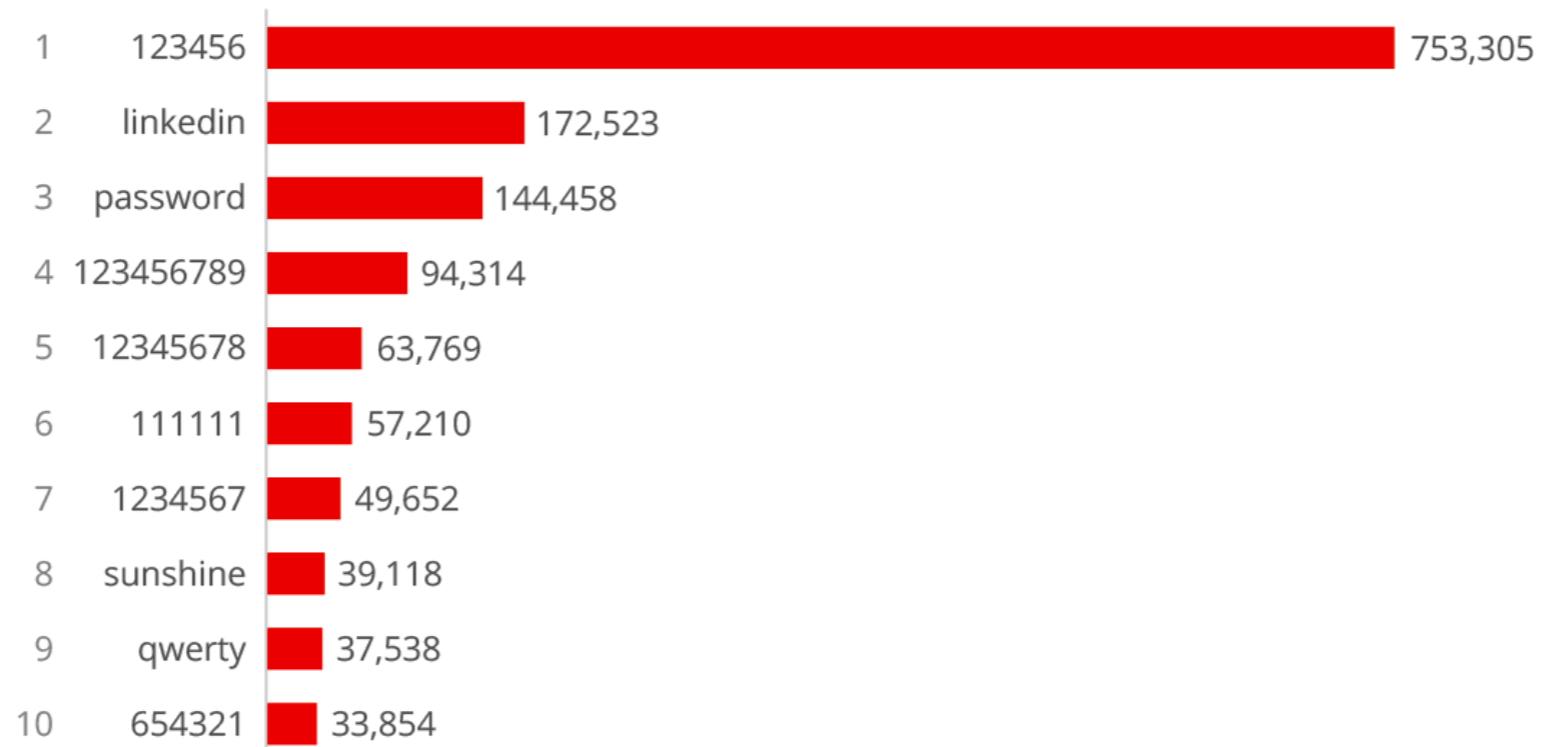
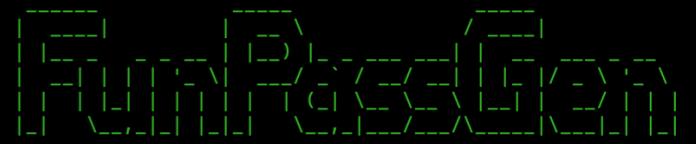


Social Media: People Still Use Pathetic Passwords

Frequency of use of LinkedIn's 10 weakest passwords

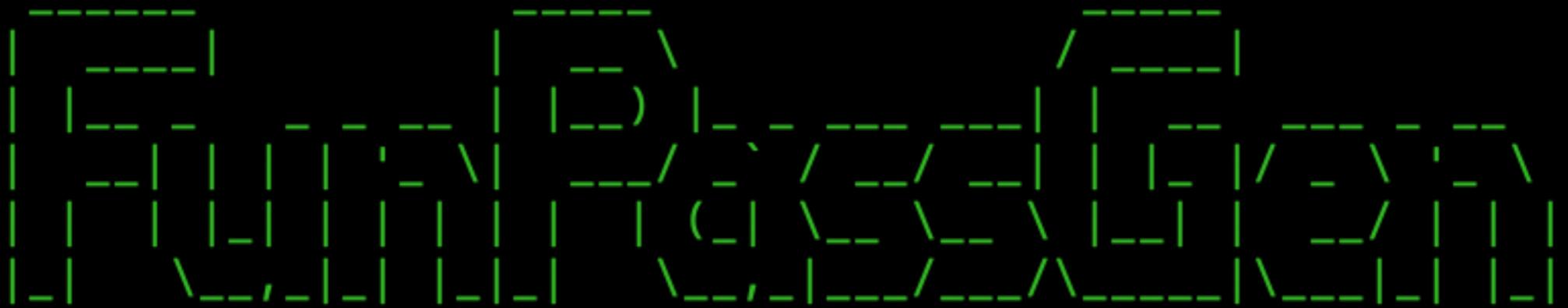


Our Process



- Brainstorming 
- App design
- Agile workflow





A Fun Password Generator

Alex Diwa and Jade Rosario
<https://github.com/alxdwa/fun-pass-gen>

The image shows a digital workspace with two boards: 'Tasks' and 'Doing'.

Tasks Board:

- Add the welcome page
- Add first question - String, get user input and store variable
- Add conditional statements to validate first question
- Add second question - String get user input and store variable
- Add conditional statements to validate second question
- Add third question - Number, get user input and store variable
- Add conditional statements to validate third question
- Write a method to combine strings and numbers to produce a password
- Output message to user that has the password, some information as well as a "memorable story" to help remember

+ Add another card

Doing Board:

- + Add a card

Done Board:

- + Add a card

The background of the workspace features a photograph of a forest with autumn-colored trees.

Code Structure



Two loops (index.rb)

- Main Flow — 2 story paths
- Encryption

Methods (methods.rb)

- ask_question
- validate_letters
- validate_numbers
- generate_password
- colorize_answers
- encrypt

```
[Alexs-MacBook-Pro-2:fun-pass-gen alexdiwa$ irb
[2.6.0 :001 >
[2.6.0 :002 >
[2.6.0 :003 >
[2.6.0 :004 >
[2.6.0 :005 > ALPHABET = "abcdefghijklmnopqrstuvwxyz"
=> "abcdefghijklmnopqrstuvwxyz"
[2.6.0 :006 > NUMBERS = "0123456789"
=> "0123456789"
[2.6.0 :007 > def validate_letters(str)
[2.6.0 :008?>   arr = str.gsub(" ", "").split("")
[2.6.0 :009?>   result = arr.all? { |letter| ALPHABET.include?(letter.downcase) }
[2.6.0 :010?>   result = false if arr.empty?
[2.6.0 :011?>   puts "Letters only, please! :)".colorize(:red) if result == false
[2.6.0 :012?>   result
[2.6.0 :013?>   end
=> :validate_letters
[2.6.0 :014 > string1 = "alex"
=> "alex"
[2.6.0 :015 > string2 = "ALExx"
=> "ALExx"
[2.6.0 :016 > string3 = " alex"
=> " alex"
[2.6.0 :017 > string4 = "#% alex"
=> "#% alex"
[2.6.0 :018 > string5 = "242"
=> "242"
[2.6.0 :019 > string6 = "_ Alex
[2.6.0 :020?> "
=> "_ Alex\n"
[2.6.0 :021 > validate_letters(string1)
=> true
[2.6.0 :022 > validate_letters(string2)
=> true
[2.6.0 :023 > validate_letters(string3)
=> true
[2.6.0 :024 > validate_letters(string4)
```

Challenges



- Cleaning up repetitive code (refactoring)
- Making up a memorable phrase
- Git and GitHub
- Time constraints

WET



```
puts "What's your first name?"
first_name = gets.chomp

until validate_letters(first_name) and first_name != "" and first_name != " "
  puts "What's your first name?"
  first_name = gets.chomp
end

puts "Nice one! What about your last name?"
last_name = gets.chomp

until validate_letters(last_name) and last_name != "" and last_name != " "
  puts "What's your last name?"
  last_name = gets.chomp
end

puts "Super! What's your lucky number?"
lucky_num = gets.chomp

until validate_numbers(lucky_num) and lucky_num != "" and lucky_num != " "
  puts "What's your lucky number?"
  lucky_num = gets.chomp
end
```

DRY



```
def ask_question(question, method)
  puts question
  var_name = gets.chomp

  until send(method, var_name)
    var_name = gets.chomp
  end

  $answers << var_name
end

ask_question("What's your first name?", "validate_letters")
ask_question("What's your last name?", "validate_letters")
ask_question("What's your lucky number?", "validate_numbers")
ask_question("What's your favourite drink? The more imaginative, the better!", "validate_letters")
ask_question("What's your favourite city?", "validate_letters")
ask_question("What's your favourite place?", "validate_letters")
p $answers
```



- **Text-to-speech**
- **More interactive and prettier UI**
- **Generating more complex and safer passwords**
- **Improving encryption/decryption methods**