

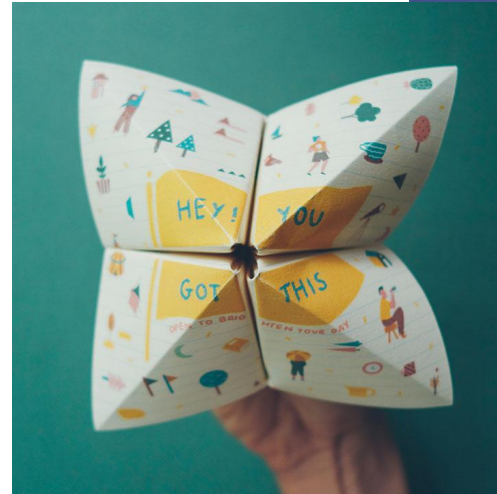
T1A3 Terminal Application

Jacqueline Cope

A dark blue diagonal gradient bar that starts from the bottom left corner and extends towards the top right corner, covering the lower half of the slide.

What is Chatterbox?

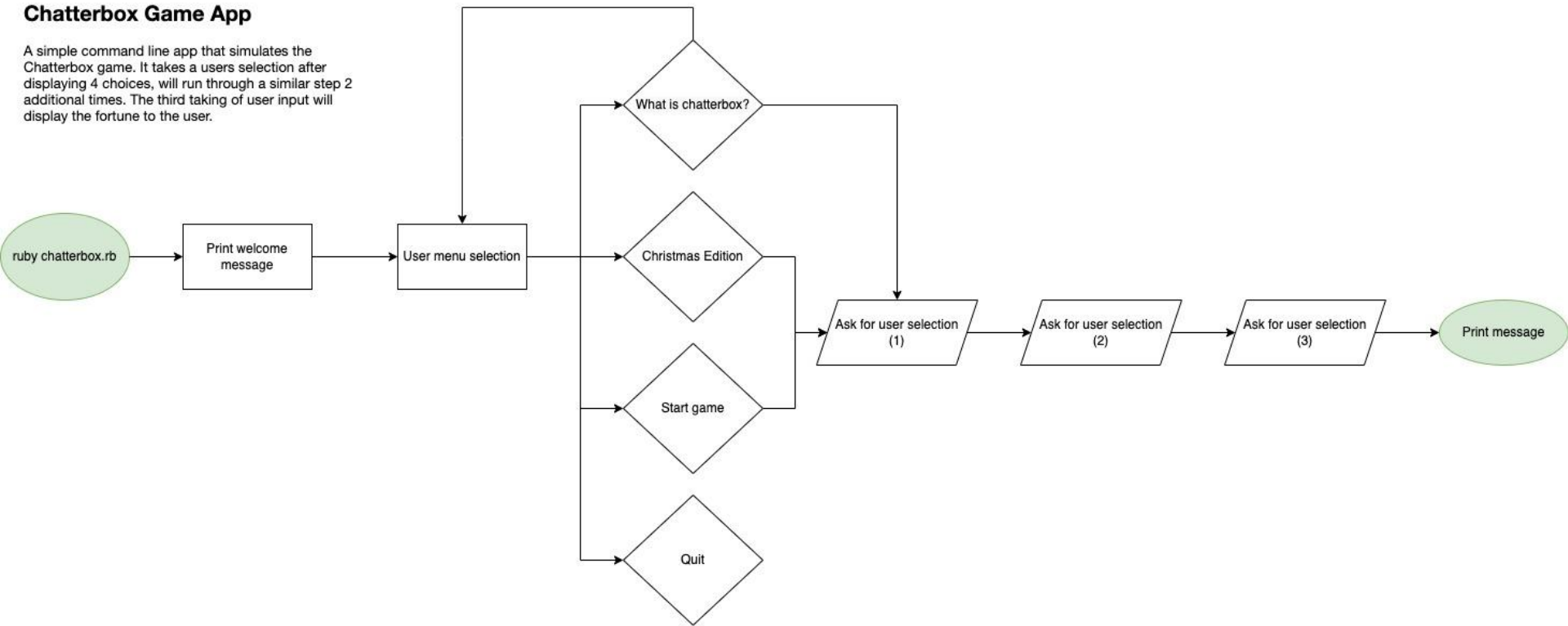
- Traditional origami fortune teller game
- Common children's game
- User has to make a number of decisions and is revealed message at the end
- Easily adapted to be played in various way - truth or dare, decision maker, jokes etc.



App Diagram

Chatterbox Game App

A simple command line app that simulates the Chatterbox game. It takes a users selection after displaying 4 choices, will run through a similar step 2 additional times. The third taking of user input will display the fortune to the user.



W711EN 03 (CH11111111111111111111)

You seek further understanding of what is Chatterbox...

What was once a childhood favourite has again been brought back to life in the form of a terminal game.

Chatterbox is traditionally an origami style game that was introduced to the western world in 1928. It was formally a fortune teller, however over has evolved and adapted to various types of games - subject to the creativity of the individual. Other names you may have heard include 'Salt Cellar', 'Cootie Catcher' or 'Fortune Teller'. The original game is played with specially folded paper (Origami), however the terminal game brings a different spin on the game. The game may have different themes, such as decision making, truth or dare or silly messages, however the steps follow a similar sequence:

- Step 1) Choose a colour from the menu
- Step 2) The colour will be spelled out
- Step 3) Choose a number from the menu
- Step 4) This number will be counted out
- Step 5) Pick another number from the menu
- Step 6) Reveal the hidden message

Select an option (Press ↑/↓ arrow to move and Enter to select)

- Start game
- Christmas Edition
- Quit

Features

- Menus - Through the use of TTY-prompt
- User Input & Output
- Use of Colours - Colorize & Lolize
- ASCII Images
- Watch this space

```
o_..".....\
{..-a"a-}
{c-._o-.)f\|
\{ ^ } \|
/.' } \|
|>:< '-._/ '.,} \_/ / ( )
>:< '-._/ '-._/ ( ' "
>:< \| \| \| \|
\| -{ } - \|
'. _ \|
/ (
jgs _.' /\ '. _.' /\ '. _.' /\ '. _.'
==
'====='
```

Ho ho ho

Why was the turkey in the pop group?

Because he was the only one with drumsticks!%

Walk-Through



- Made up of cases

- Takes user input via menu

- Confirms user choice

- Allows letters of word to print individually

#BASIC MENU SYSTEM

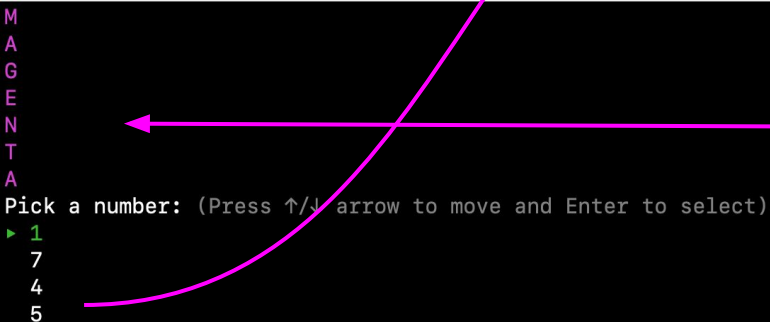
```
choices = {
    'Start game': 1,
    'Christmas Edition': 2,
    'What is Chatterbox?': 3,
    'Quit': 4
}

input = prompt.select('Select an option to start', choices)
puts input

case input
when 1
    puts 'You have selected to Play'.colorize(:orange)
    system "clear"
    colours = { 'Blue': 1, 'Magenta': 2, 'Yellow': 3, 'Emerald': 4 }
    colour_choice = prompt.select('Pick a colour:', colours)
    puts "You have selected #{colours.key(colour_choice)}"
    # Assigns key names as an uppercase string to the variable letters
    letters = colours.key(colour_choice).to_s.upcase
    system "clear"
    spell_l(letters)
    number_choice1 = prompt.select('Pick a number:', random_numbers)
    system "clear"
    puts "You have selected #{number_choice1}"
    count_nums(number_choice1)
    number_choice2 = prompt.select('Pick another number:', random_numbers)
    system "clear"
    puts fortune_ascii_art
    puts "You have selected #{number_choice2}"
    fortune(number_choice2, rainbow)
```



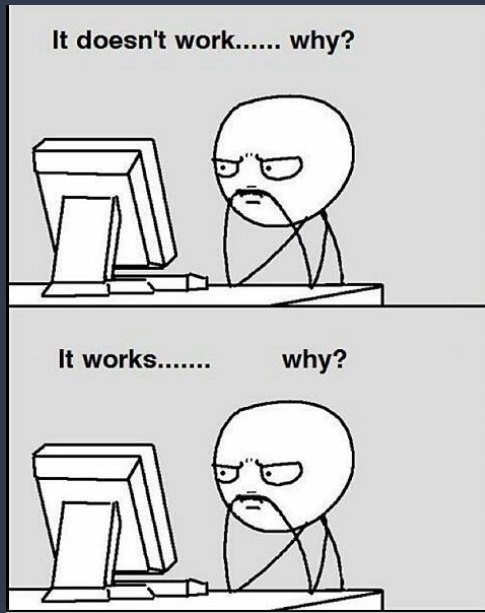
Methods



```
# prints banner in welcome page
def banner_ascii
  begin
    puts File.read('./data/banner.txt')
  rescue
    puts "Something unexpected happened to the image that should be here".colorize(:red)
  end
end

# returns an array of 4 random numbers from 1-8
def random_numbers
  numbers = [1, 2, 3, 4, 5, 6, 7, 8]
  random_nums = numbers.shuffle[0, 4]
  return random_nums
end

# loops through the colour selected, printing each letter on a separate line in its colour
def spell_l(letters)
  if letters == 'BLUE'
    letters.split('').each { |l| puts l.colorize(:cyan) }
  elsif letters == 'MAGENTA'
    letters.split('').each { |l| puts l.colorize(:magenta) }
  elsif letters == 'YELLOW'
    letters.split('').each { |l| puts l.colorize(:yellow) }
  else
    letters.split('').each { |l| puts l.colorize(:green) }
  end
end
```

Development Review

Challenges:

- Understanding the logic of how to write the code
- Using gems
- Iteration method for generating random number
- Use of hashes in menu options
- Testing
- Planning

Ethical Issues:

- Potentially problematic for those with vision impairments or colour blindness

Favourite Parts:

- Finding cool gems - eg. 'Lolize'
- Writing code that works!
- Gradually starting to understand how to handle error messages

Thank you!