Snake Game

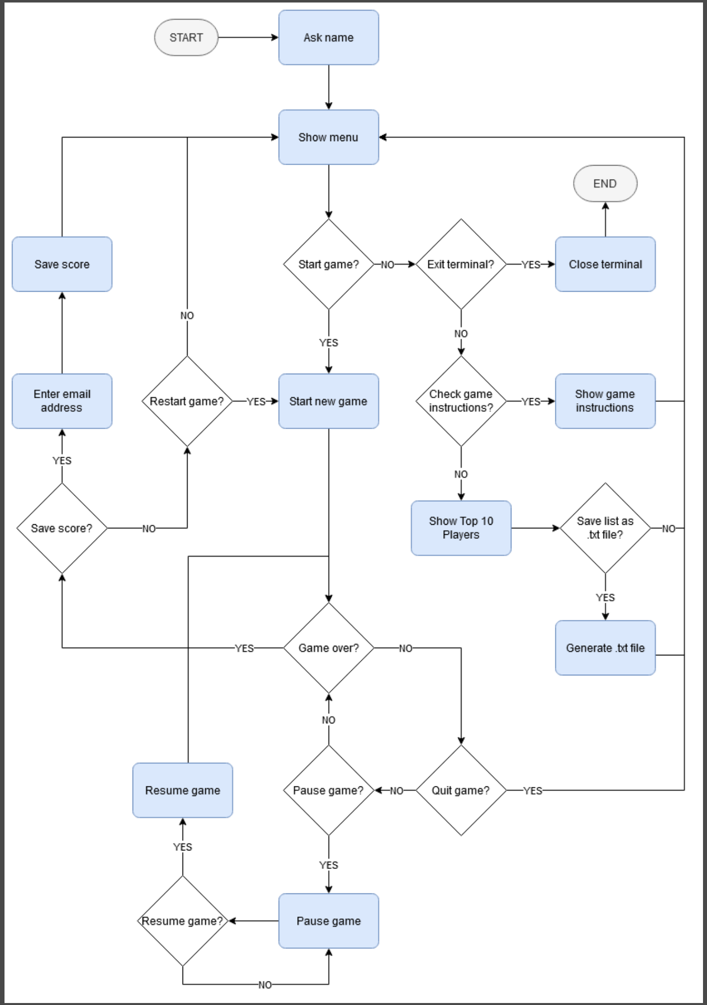
A Ruby Terminal Application

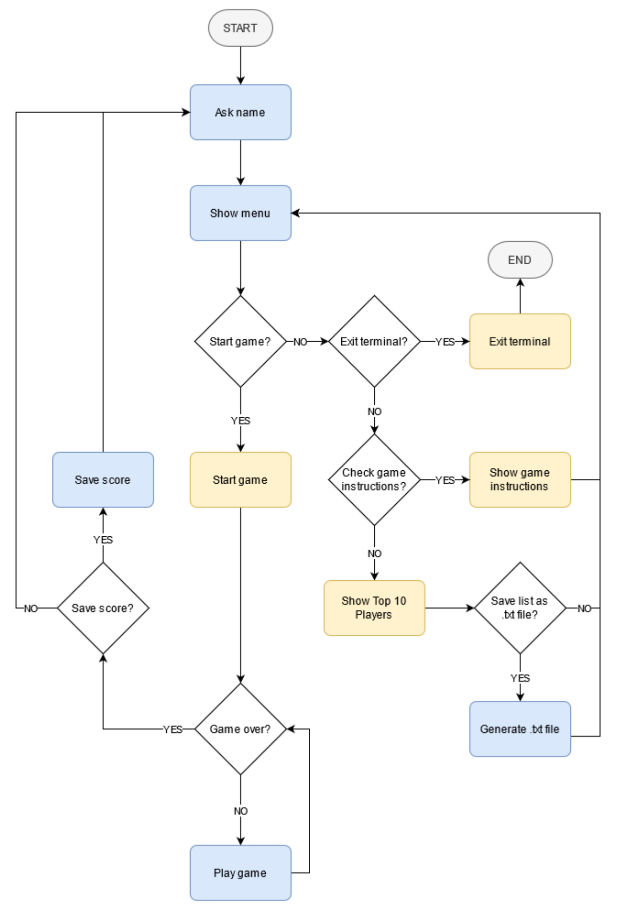
●●●

Irah Rosete

19 December 2020

Flow


START
Ask name
Show menu
END
Save score
Start game?
NO
Exit terminal?
YES
Close terminal
NO
YES
NO
Enter email address
Restart game?
YES
Start new game
Check game instructions?
YES
Show game instructions
YES
NO
Save score?
NO
Show Top 10 Players
Save list as .txt file?
NO
YES
YES
Game over?
NO
Generate .txt file
NO
Resume game
Pause game?
NO
Quit game?
YES
YES
YES
Resume game?
Pause game
NO


START
Ask name
Show menu
END
Start game?
NO
Exit terminal?
YES
Exit terminal
YES
NO
Save score
Start game
Check game instructions?
YES
Show game instructions
YES
NO
NO
Save score?
Show Top 10 Players
Save list as .txt file?
NO
YES
YES
Game over?
Generate .txt file
NO
Play game

Features

- Menu
  - Game
    - User Control
    - Scoring System
    - ~~Pause / Resume / Quit~~
  - Play Instructions
  - Top 10 Players
    - File Output
    - ~~Notification~~
  - Exit

Process

Planning

README file

Flow Chart

Implementation Board

Initial Code

Menu - easiest

Hard code:

- How to Play instructions
- Top 10 Players list

Blocker

Ruby2D does not work on WSL

Sidetrack: ASCII Art

Plan B: Create the game raw

Process

Serious Code

Board class

Snake class

Used File class to the Top 10 Players list

Prey class and Random class

Scoring system logic

Another Blocker

User control

Sidetrack: YAML to create a database for Top 10 Players

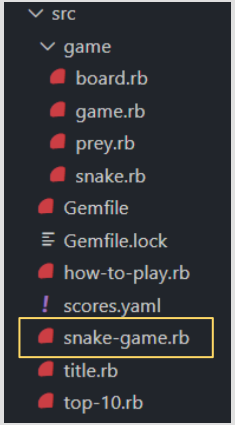
Assistance on:

- IO/console getch
- .txt content format
- Regenerate prey

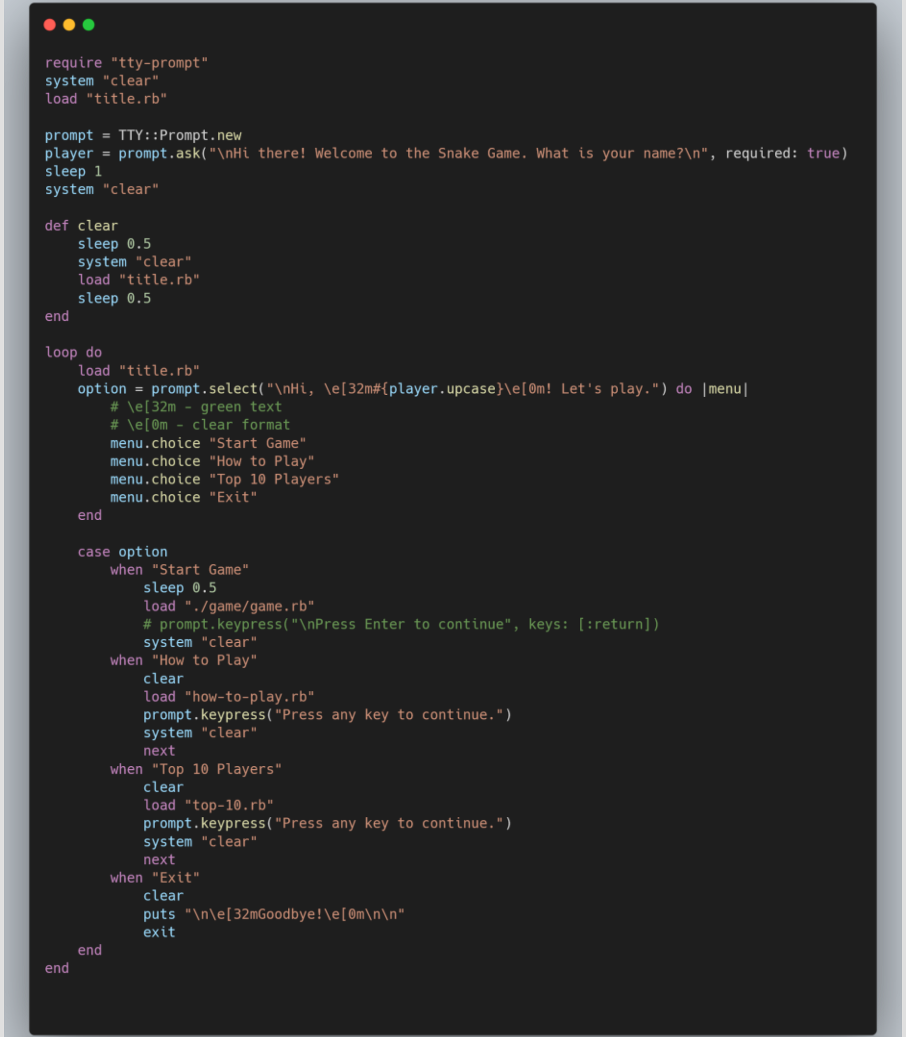
Unresolved Blockers

Make the snake move

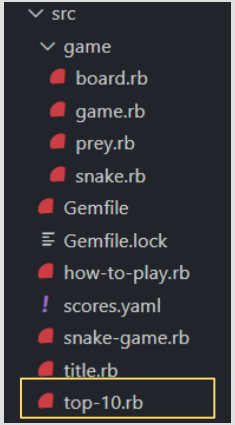
Catch new score and player name to determine new Top 10



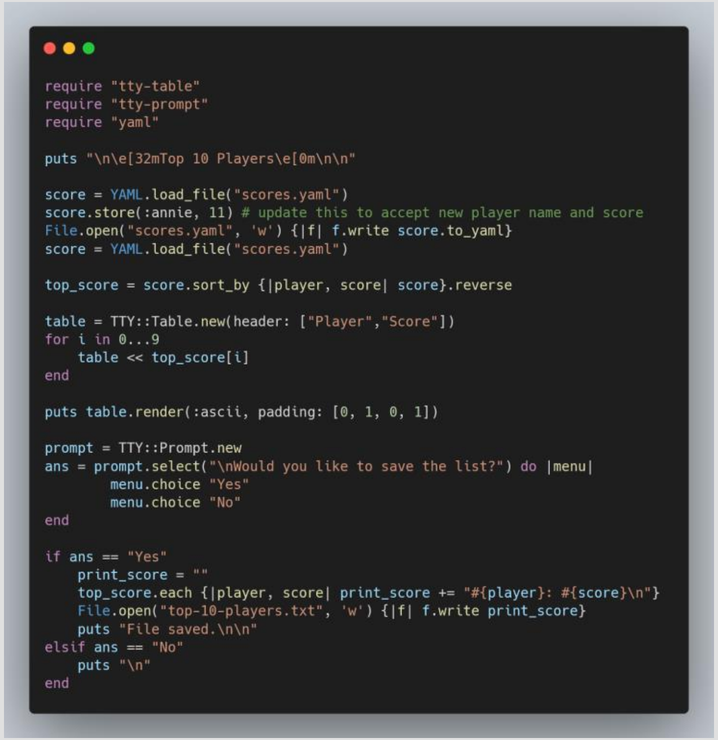
```
src
  game
    board.rb
    game.rb
    prey.rb
    snake.rb
  Gemfile
  Gemfile.lock
  how-to-play.rb
  scores.yaml
  snake-game.rb
  title.rb
  top-10.rb
```



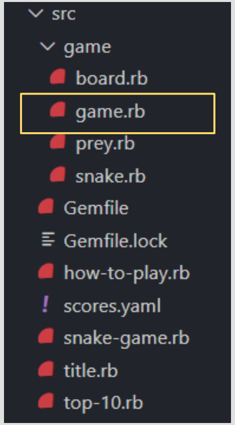
```ruby
require "tty-prompt"
system "clear"
load "title.rb"

prompt = TTY::Prompt.new
player = prompt.ask("\nHi there! Welcome to the Snake Game. What is your name?\n", required: true)
sleep 1
system "clear"

def clear
    sleep 0.5
    system "clear"
    load "title.rb"
    sleep 0.5
end

loop do
    load "title.rb"
    option = prompt.select("\nHi, \e[32m#{player.upcase}\e[0m! Let's play.") do |menu|
        # \e[32m - green text
        # \e[0m - clear format
        menu.choice "Start Game"
        menu.choice "How to Play"
        menu.choice "Top 10 Players"
        menu.choice "Exit"
    end

    case option
        when "Start Game"
            sleep 0.5
            load "./game/game.rb"
            # prompt.keypress("\nPress Enter to continue", keys: [:return])
            system "clear"
        when "How to Play"
            clear
            load "how-to-play.rb"
            prompt.keypress("Press any key to continue.")
            system "clear"
            next
        when "Top 10 Players"
            clear
            load "top-10.rb"
            prompt.keypress("Press any key to continue.")
            system "clear"
            next
        when "Exit"
            clear
            puts "\n\e[32mGoodbye!\e[0m\n\n"
            exit
    end
end
```



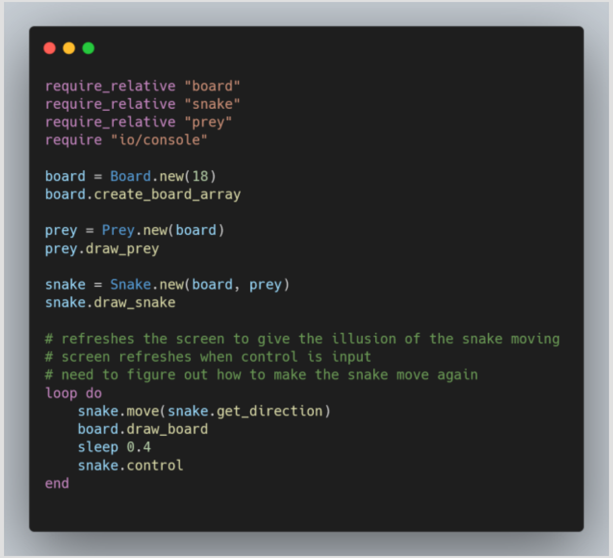
```
src
  game
    board.rb
    game.rb
    prey.rb
    snake.rb
  Gemfile
  Gemfile.lock
  how-to-play.rb
  scores.yaml
  snake-game.rb
  title.rb
  top-10.rb
```



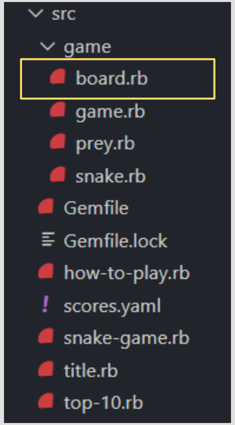
```ruby
require "tty-table"
require "tty-prompt"
require "yaml"

puts "\n\e[32mTop 10 Players\e[0m\n\n"

score = YAML.load_file("scores.yaml")
score.store(:annie, 11) # update this to accept new player name and score
File.open("scores.yaml", 'w') {|f| f.write score.to_yaml}
score = YAML.load_file("scores.yaml")

top_score = score.sort_by {|player, score| score}.reverse

table = TTY::Table.new(header: ["Player","Score"])
for i in 0...9
    table << top_score[i]
end

puts table.render(:ascii, padding: [0, 1, 0, 1])

prompt = TTY::Prompt.new
ans = prompt.select("\nWould you like to save the list?") do |menu|
        menu.choice "Yes"
        menu.choice "No"
end

if ans == "Yes"
    print_score = ""
    top_score.each {|player, score| print_score += "#{player}: #{score}\n"}
    File.open("top-10-players.txt", 'w') {|f| f.write print_score}
    puts "File saved.\n\n"
elsif ans == "No"
    puts "\n"
end
```



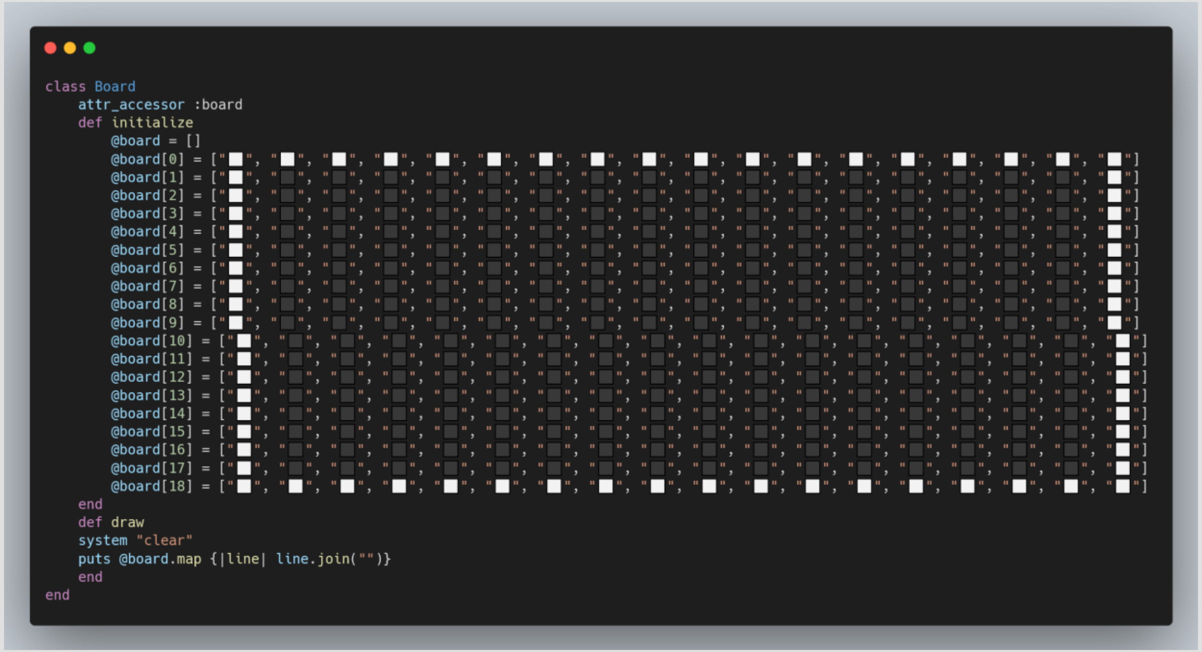
```
src
  game
    board.rb
    game.rb
    prey.rb
    snake.rb
  Gemfile
  Gemfile.lock
  how-to-play.rb
  scores.yaml
  snake-game.rb
  title.rb
  top-10.rb
```



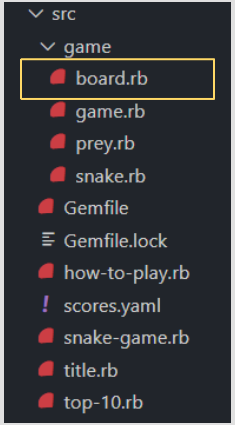
```ruby
require_relative "board"
require_relative "snake"
require_relative "prey"
require "io/console"

board = Board.new(18)
board.create_board_array

prey = Prey.new(board)
prey.draw_prey

snake = Snake.new(board, prey)
snake.draw_snake

# refreshes the screen to give the illusion of the snake moving
# screen refreshes when control is input
# need to figure out how to make the snake move again
loop do
    snake.move(snake.get_direction)
    board.draw_board
    sleep 0.4
    snake.control
end
```



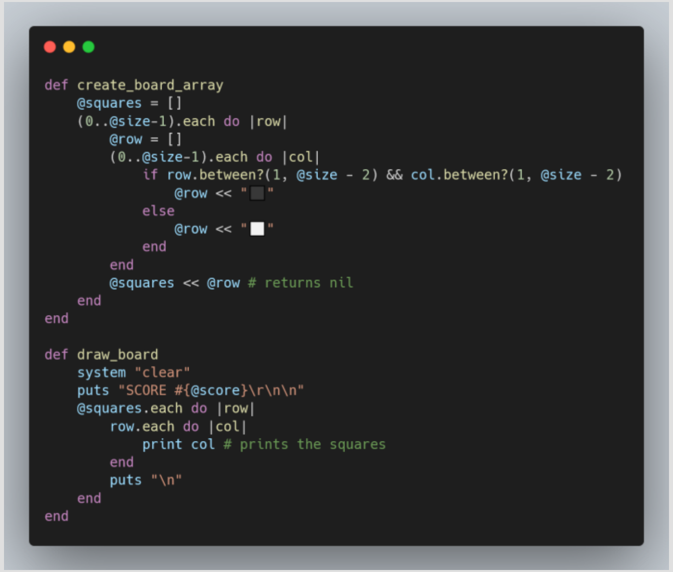
```
src
  game
    board.rb
    game.rb
    prey.rb
    snake.rb
  Gemfile
  Gemfile.lock
  how-to-play.rb
  scores.yaml
  snake-game.rb
  title.rb
  top-10.rb
```



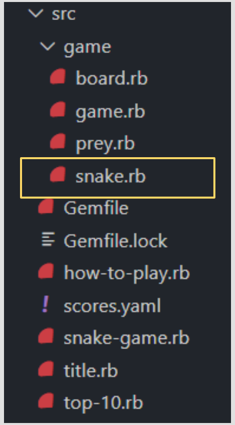
```ruby
class Board
    attr_accessor :board
    def initialize
        @board = []
        @board[0] = ["⬜", "⬜", "⬜", "⬜", "⬜", "⬜", "⬜", "⬜", "⬜", "⬜", "⬜", "⬜", "⬜", "⬜", "⬜", "⬜", "⬜", "⬜"]
        @board[1] = ["⬜", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬜"]
        @board[2] = ["⬜", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬜"]
        @board[3] = ["⬜", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬜"]
        @board[4] = ["⬜", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬜"]
        @board[5] = ["⬜", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬜"]
        @board[6] = ["⬜", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬜"]
        @board[7] = ["⬜", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬜"]
        @board[8] = ["⬜", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬜"]
        @board[9] = ["⬜", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬜"]
        @board[10] = ["⬜", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬜"]
        @board[11] = ["⬜", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬜"]
        @board[12] = ["⬜", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬜"]
        @board[13] = ["⬜", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬜"]
        @board[14] = ["⬜", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬜"]
        @board[15] = ["⬜", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬜"]
        @board[16] = ["⬜", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬜"]
        @board[17] = ["⬜", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬛", "⬜"]
        @board[18] = ["⬜", "⬜", "⬜", "⬜", "⬜", "⬜", "⬜", "⬜", "⬜", "⬜", "⬜", "⬜", "⬜", "⬜", "⬜", "⬜", "⬜", "⬜"]
    end
    def draw
    system "clear"
    puts @board.map {|line| line.join("")}
    end
end
```

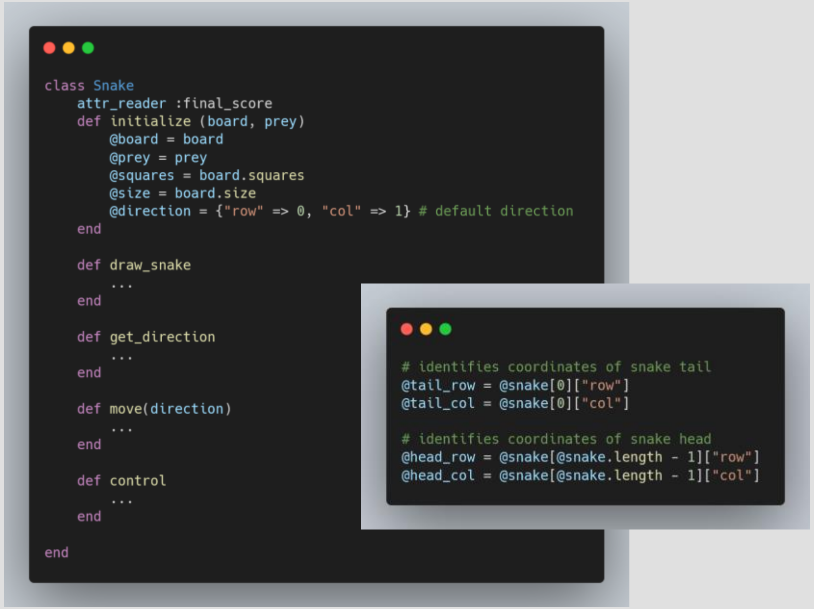


```
src
  game
    board.rb
    game.rb
    prey.rb
    snake.rb
  Gemfile
  Gemfile.lock
  how-to-play.rb
  scores.yaml
  snake-game.rb
  title.rb
  top-10.rb
```



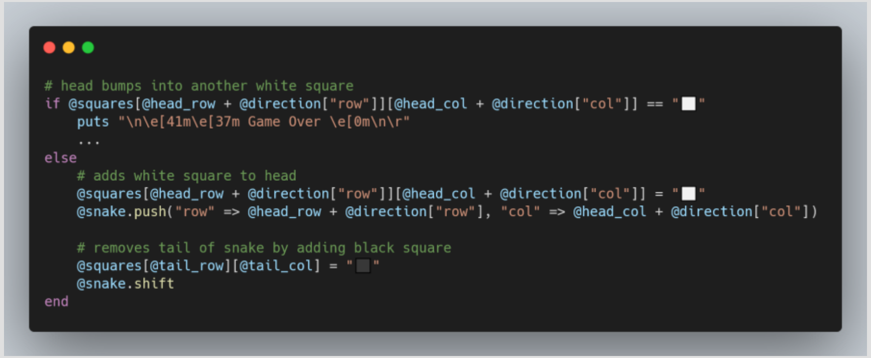
```ruby
def create_board_array
    @squares = []
    (0..@size-1).each do |row|
        @row = []
        (0..@size-1).each do |col|
            if row.between?(1, @size - 2) && col.between?(1, @size - 2)
                @row << "⬛"
            else
                @row << "⬜"
            end
        end
        @squares << @row # returns nil
    end
end

def draw_board
    system "clear"
    puts "SCORE #{@score}\r\n\n"
    @squares.each do |row|
        row.each do |col|
            print col # prints the squares
        end
        puts "\n"
    end
end
```



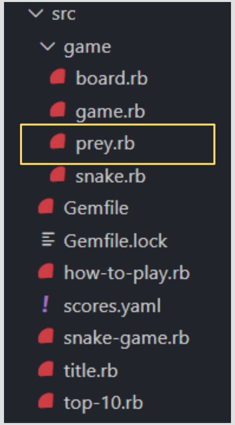


```ruby
class Snake
    attr_reader :final_score
    def initialize (board, prey)
        @board = board
        @prey = prey
        @squares = board.squares
        @size = board.size
        @direction = {"row" => 0, "col" => 1} # default direction
    end

    def draw_snake
        ...
    end

    def get_direction
        ...
    end

    def move(direction)
        ...
    end

    def control
        ...
    end

end
```

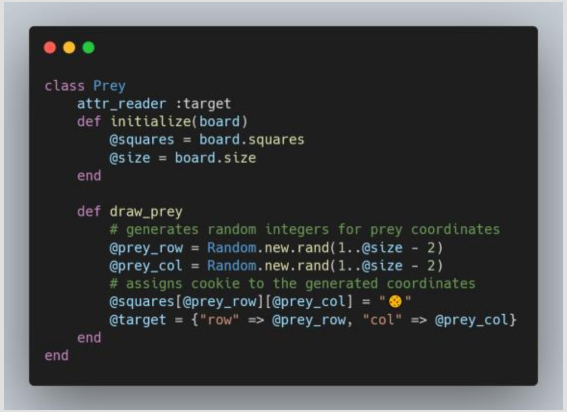
```ruby
# identifies coordinates of snake tail
@tail_row = @snake[0]["row"]
@tail_col = @snake[0]["col"]

# identifies coordinates of snake head
@head_row = @snake[@snake.length - 1]["row"]
@head_col = @snake[@snake.length - 1]["col"]
```



```ruby
# head bumps into another white square
if @squares[@head_row + @direction["row"]][@head_col + @direction["col"]] == "⬜"
    puts "\n\e[41m\e[37m Game Over \e[0m\n\r"
    ...
else
    # adds white square to head
    @squares[@head_row + @direction["row"]][@head_col + @direction["col"]] = "⬜"
    @snake.push("row" => @head_row + @direction["row"], "col" => @head_col + @direction["col"])

    # removes tail of snake by adding black square
    @squares[@tail_row][@tail_col] = "⬛"
    @snake.shift
end
```



```
src
  game
    board.rb
    game.rb
    prey.rb
    snake.rb
  Gemfile
  Gemfile.lock
  how-to-play.rb
  scores.yaml
  snake-game.rb
  title.rb
  top-10.rb
```



```ruby
class Prey
    attr_reader :target
    def initialize(board)
        @squares = board.squares
        @size = board.size
    end

    def draw_prey
        # generates random integers for prey coordinates
        @prey_row = Random.new.rand(1..@size - 2)
        @prey_col = Random.new.rand(1..@size - 2)
        # assigns cookie to the generated coordinates
        @squares[@prey_row][@prey_col] = "🍪"
        @target = {"row" => @prey_row, "col" => @prey_col}
    end
end
```

Demo

Challenges

Dependencies	● Ruby2D ● Had to force a Plan B
Re/Learning	● Grids ● Coordinates (x, y)
Scaling down	● Had to remove features to make project achievable ● Then look to add when basic functions are done
Research	● Never ending research

Learnings

RoRo Sydney	● Attending the meet up taught me about being able to use emojis as string!
Ruby Public classes	● File ● Random
Key Variable	● @square
More than just Ruby	● Planning, organising files and classes ● Visualising what needs to happen & working backwards

Thank You!