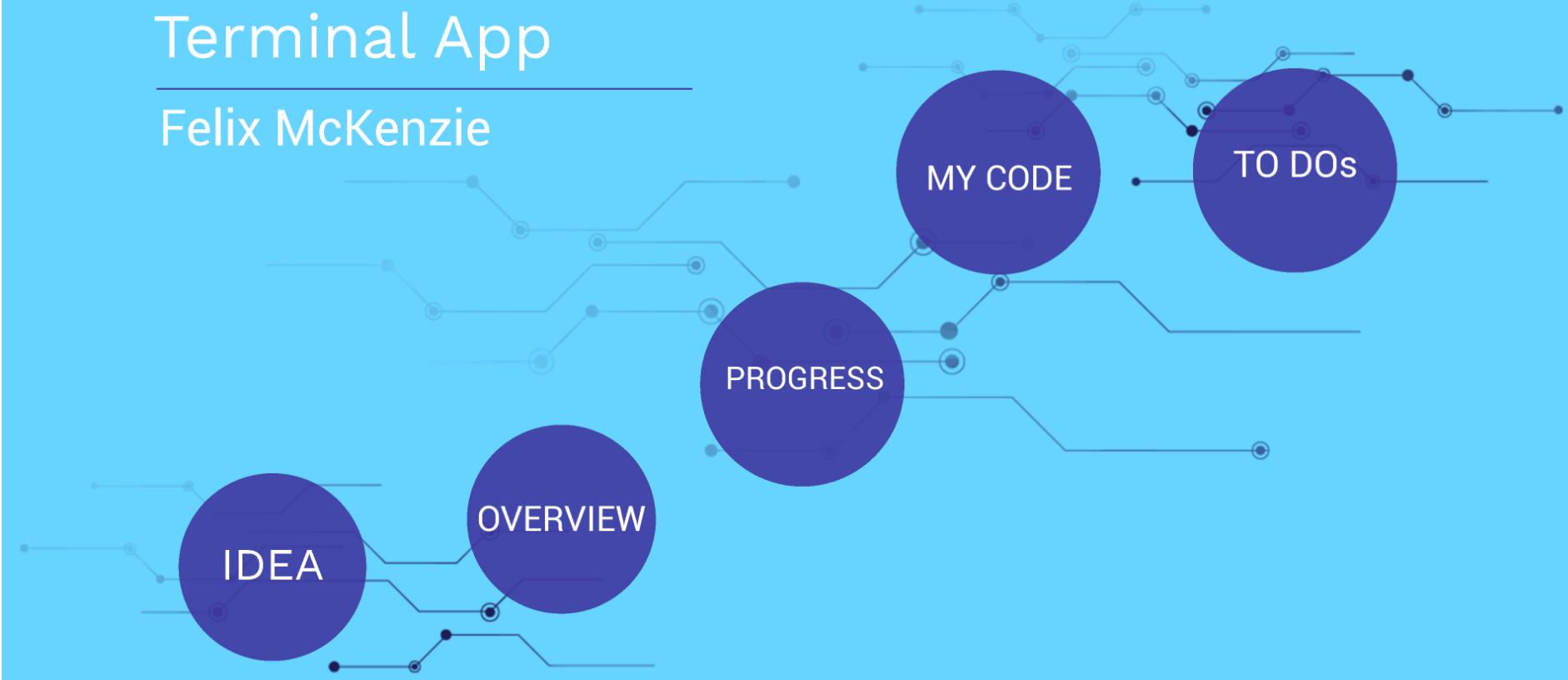


# Terminal App

Felix McKenzie



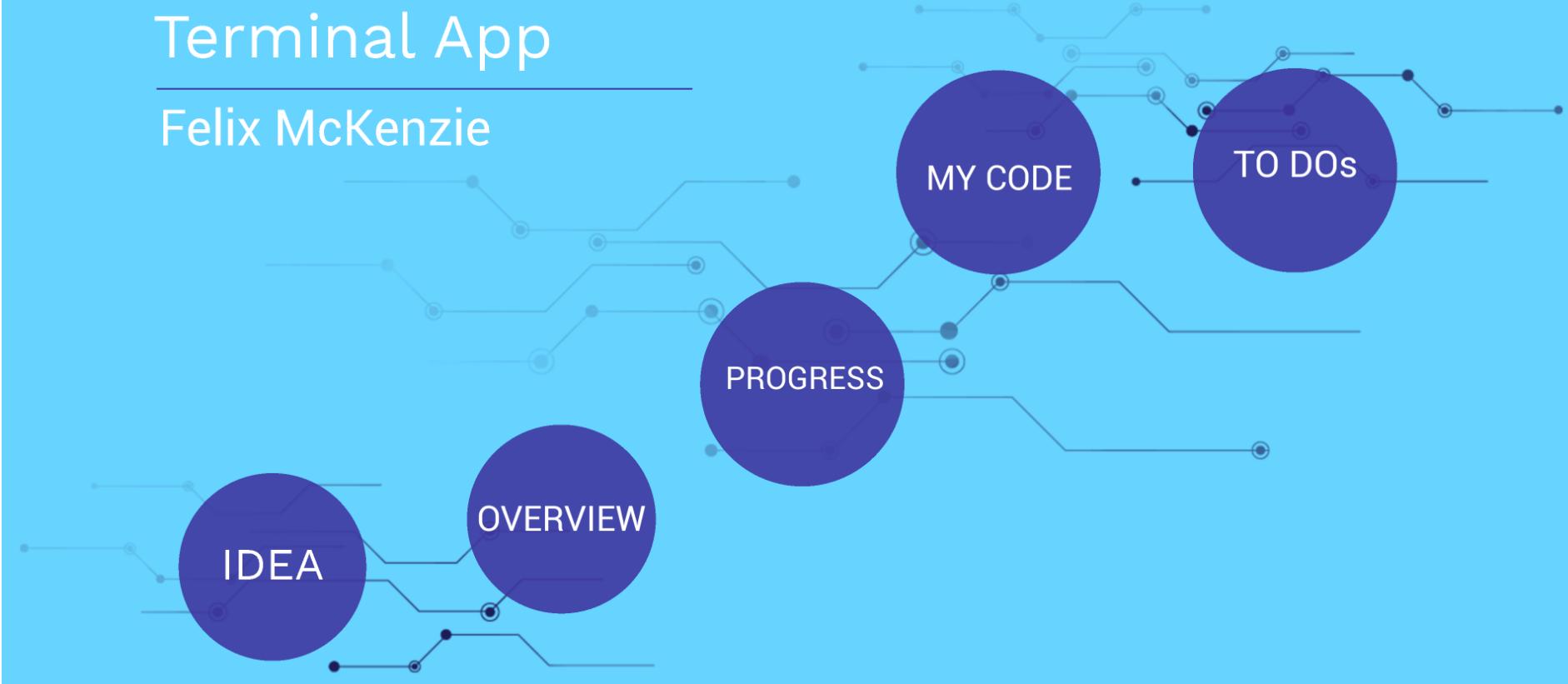
## THE IDEA

Quiz app for the terminal



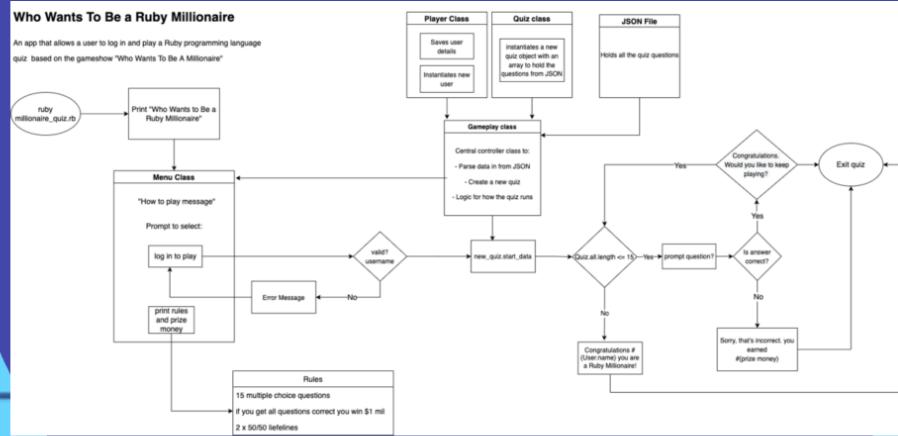
# Terminal App

Felix McKenzie



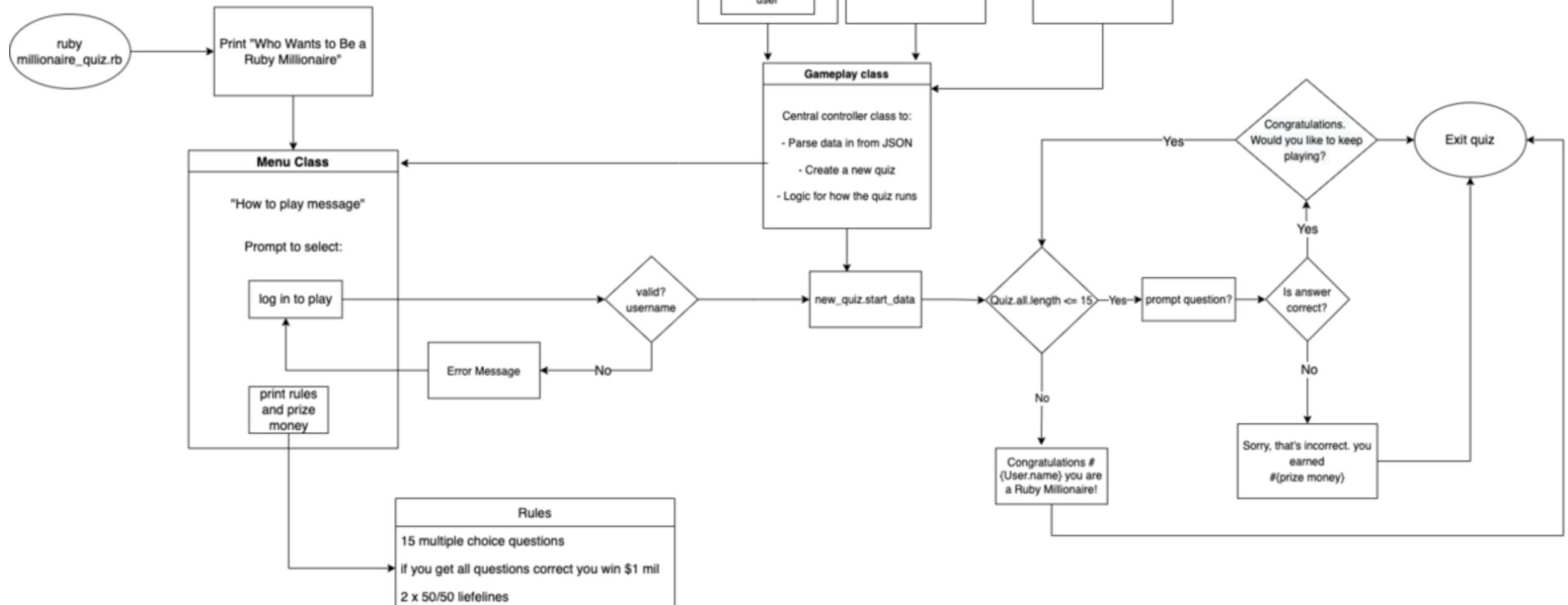
# FLOW CHART

## overview



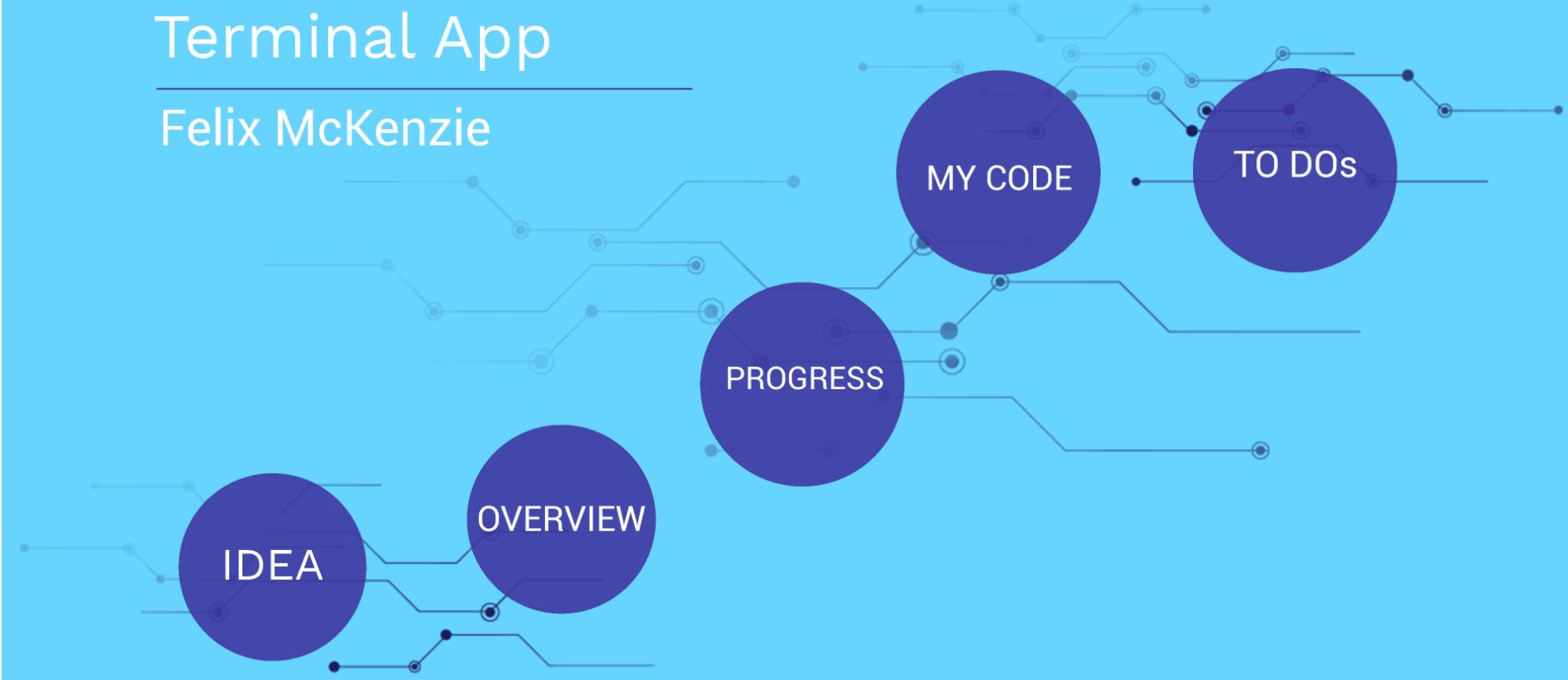
## Who Wants To Be a Ruby Millionaire

An app that allows a user to log in and play a Ruby programming language quiz based on the gameshow "Who Wants To Be A Millionaire"

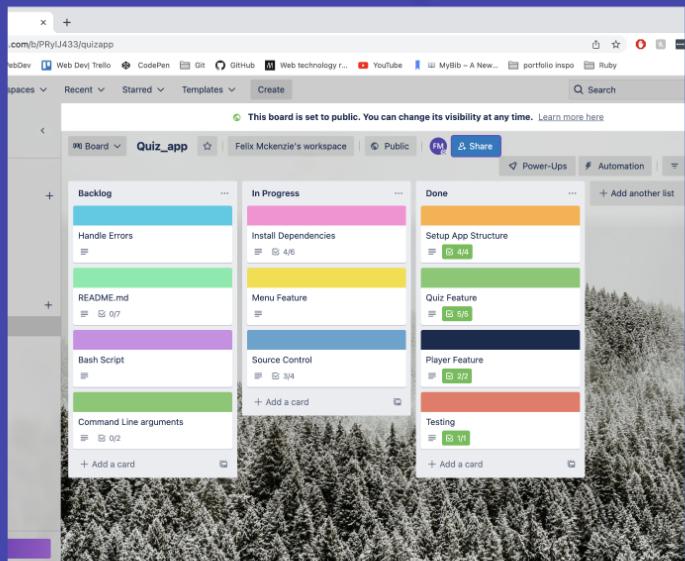


# Terminal App

Felix McKenzie



# SO FAR....

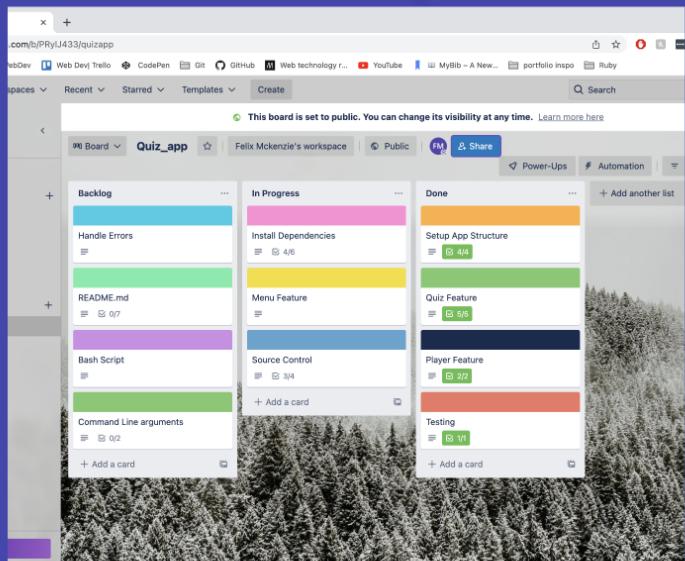


## CHALLENGES

Passing methods and variables between classes. "Gluing it all together"

Debugging code where it uses multiple classes. I need to remind myself to slow down and read the error messages properly

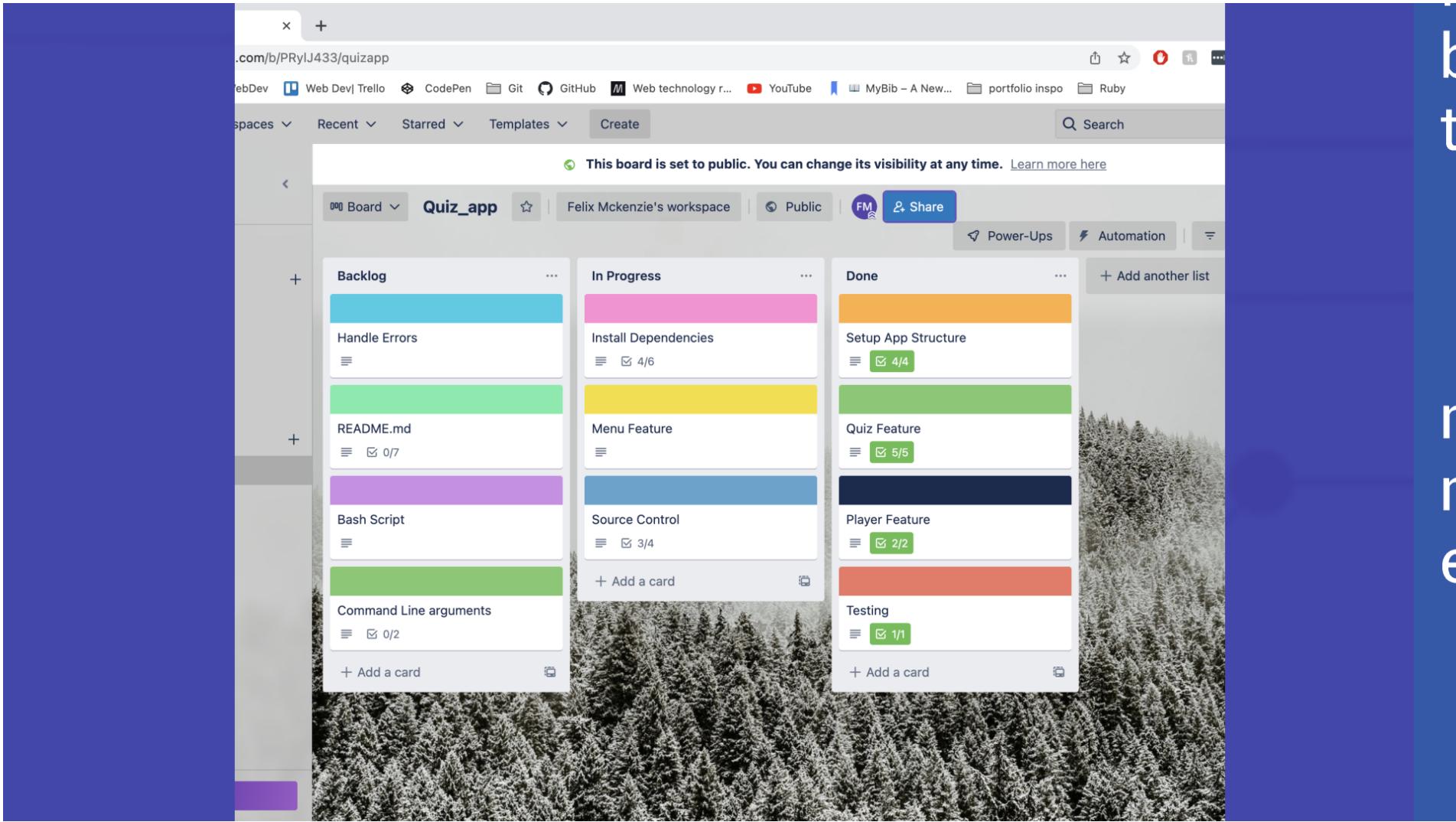
# SO FAR....



## CHALLENGES

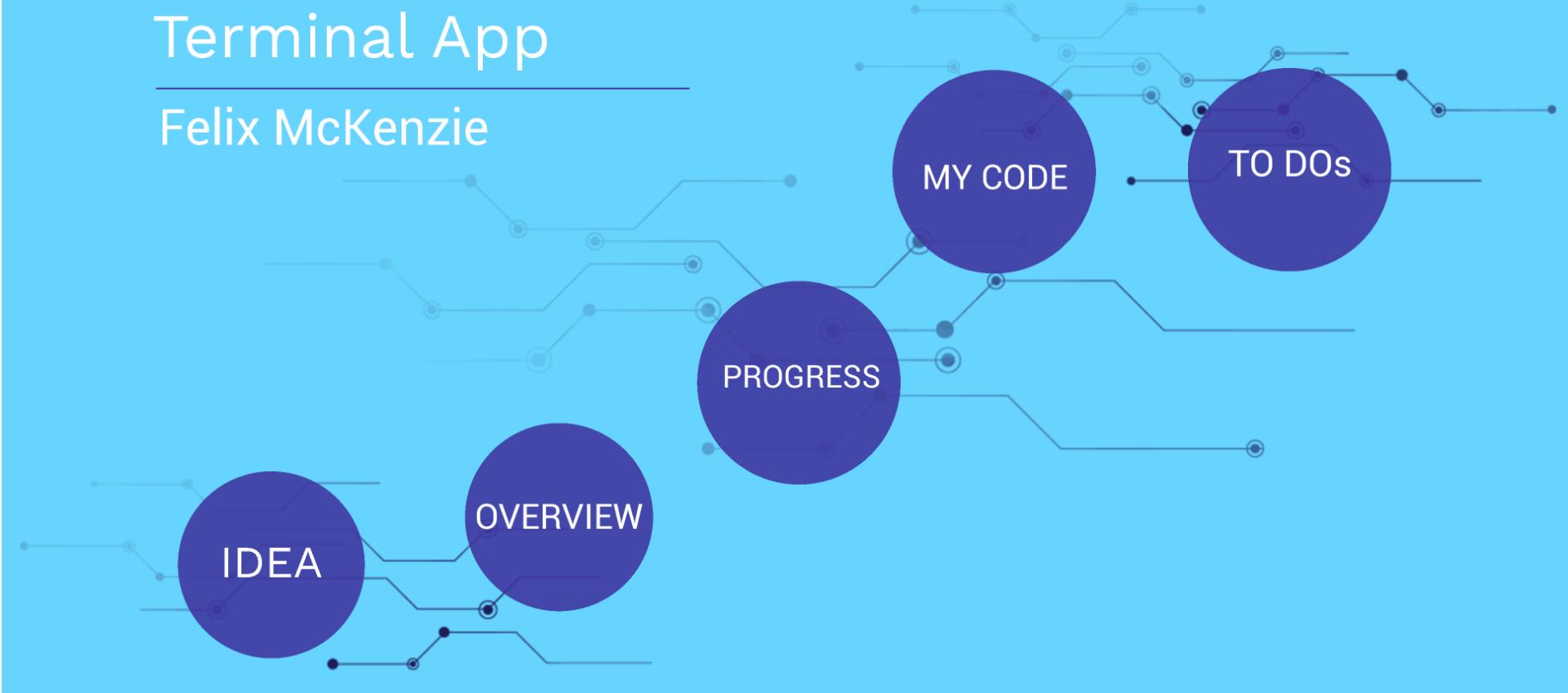
Passing methods and variables between classes. "Gluing it all together"

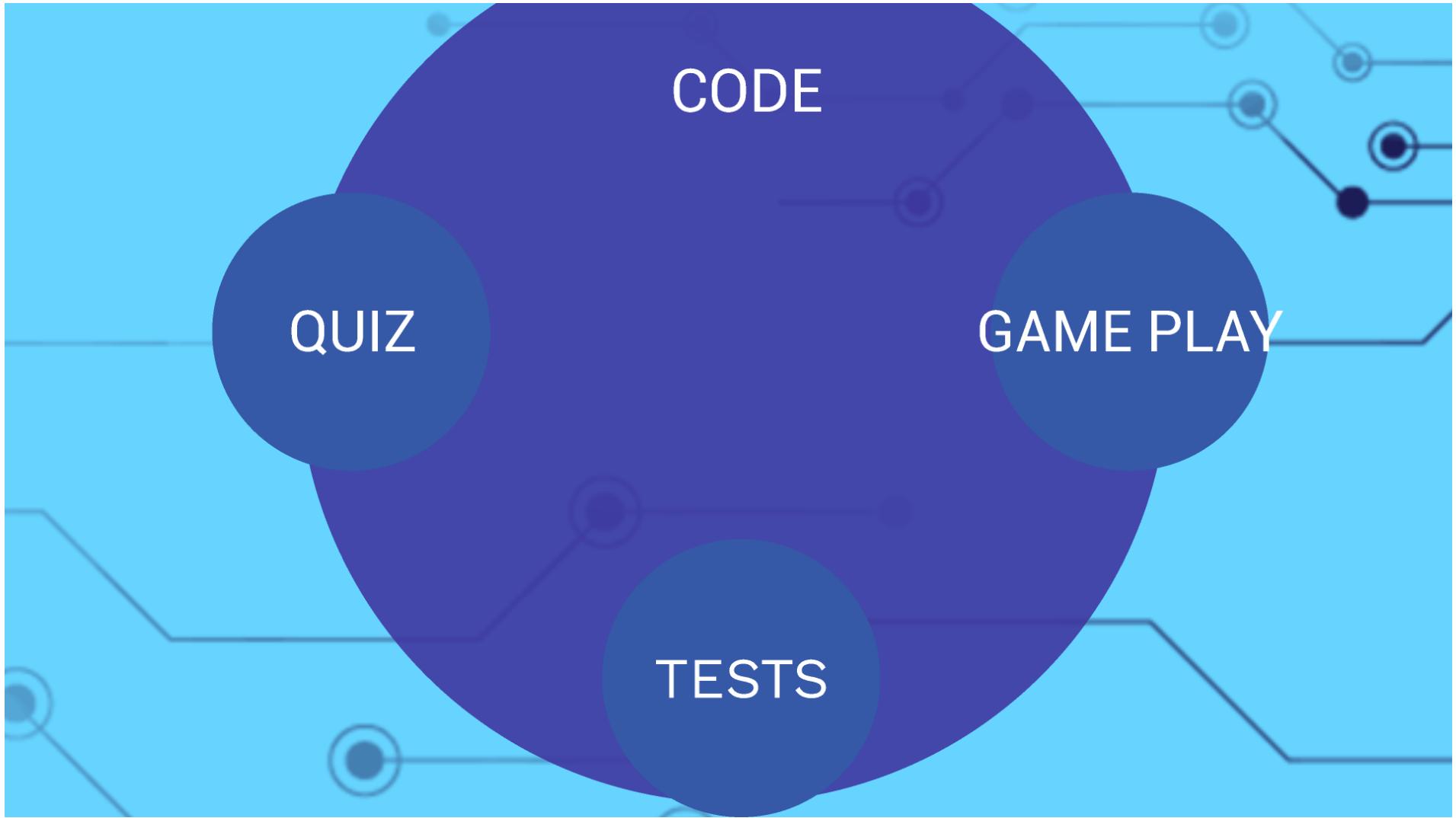
Debugging code where it uses multiple classes. I need to remind myself to slow down and read the error messages properly



# Terminal App

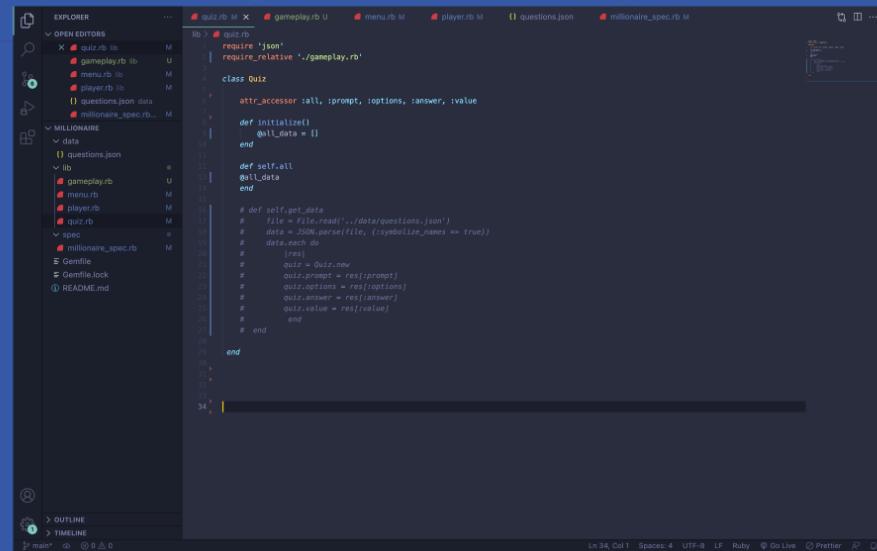
Felix McKenzie





# QUIZ CLASS

New instance created when data is parsed from JSON file, has an array instance variable that holds the questions



```
quiz.rb M X gameplay.rb U menu.rb M player.rb M questions.json millionaire_spec.rb M

require 'json'
require_relative './gameplay.rb'

class Quiz
  attr_accessor :all, :prompt, :options, :answer, :value
  def initialize()
    @all_data = []
  end

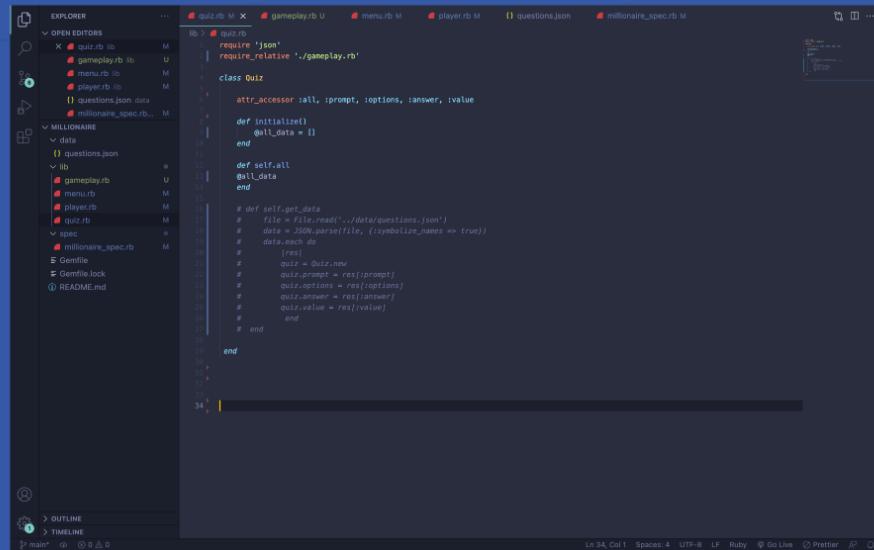
  def self.all
    @all_data
  end

  # def self.get_data
  #   file = File.read('../data/questions.json')
  #   data = JSON.parse(file, (symbolize_names => true))
  #   data.each do
  #     |res|
  #     quiz = Quiz.new
  #     quiz.prompt = res[:prompt]
  #     quiz.options = res[:options]
  #     quiz.answer = res[:answer]
  #     quiz.value = res[:value]
  #   end
  # end

end
```

# QUIZ CLASS

New instance created when data is parsed from JSON file, has an array instance variable that holds the questions



```
quiz.rb M × gameplay.rb U menu.rb M player.rb M questions.json millionaire_spec.rb M

require 'json'
require_relative './gameplay.rb'

class Quiz
  attr_accessor :all, :prompt, :options, :answer, :value
  def initialize()
    @all_data = []
  end

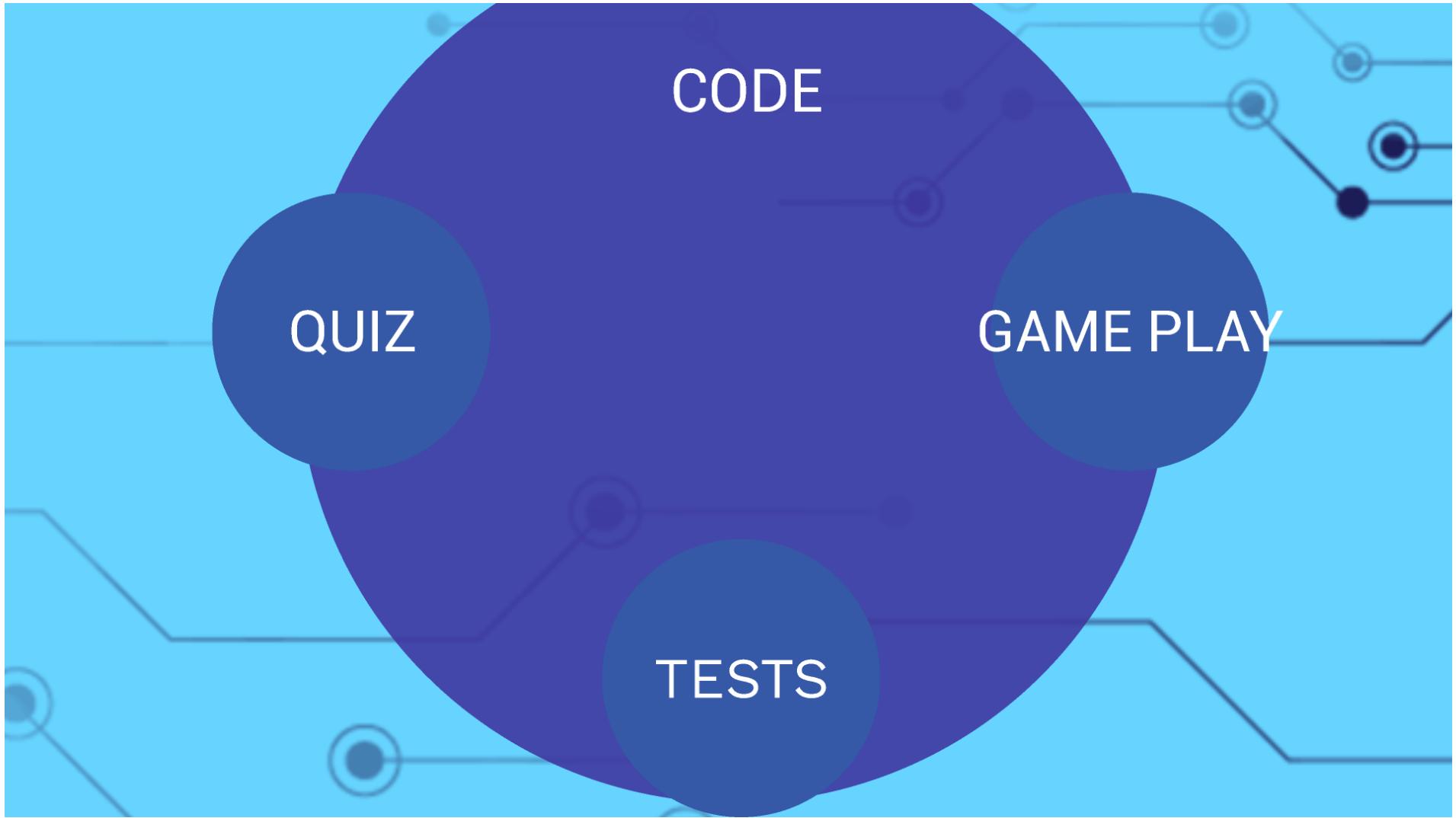
  def self.all
    @all_data
  end

  # def self.get_data
  #   file = File.read('../data/questions.json')
  #   data = JSON.parse(file, (symbolize_names => true))
  #   data.each do
  #     |res|
  #     quiz = Quiz.new
  #     quiz.prompt = res[:prompt]
  #     quiz.options = res[:options]
  #     quiz.answer = res[:answer]
  #     quiz.value = res[:value]
  #   end
  # end

end
```

The screenshot shows the VS Code interface with the following details:

- Explorer View:** Shows the project structure. The root folder contains files like quiz.rb, gameplay.rb, menu.rb, player.rb, questions.json, and millionaire\_spec.rb. Inside the MILLIONAIRE folder, there are subfolders lib, data, and spec, along with Gemfile and Gemfile.lock.
- Editor View:** The quiz.rb file is open. The code defines a Quiz class with attr\_accessor for :all, :prompt, :options, :answer, and :value. It includes an initialize method that initializes @all\_data as an empty array. It also includes a self.all method that returns @all\_data. A get\_data method is present but commented out, showing how it would read from a JSON file and parse it into an array of hashes, each containing prompt, options, answer, and value.
- Bottom Status Bar:** Shows the current file is quiz.rb, line 34, column 1. It also displays settings for Spaces: 4, UTF-8, LF, Ruby, Go Live, Prettier, and other icons.



# GAMEPLAY CLASS

Contains core methods to parse data, execute a quiz, allow the user to interact and determine outcomes.

The image shows two side-by-side code editors in a development environment, likely Visual Studio Code, illustrating the evolution of a Ruby class named `Gameplay`.

**Left Editor (Line 0 to Line 45):**

```
class Gameplay
  def initialize(player)
    @player = player
  end

  def self.get_data
    File.read("../Data/questions.json")
  end

  data = JSON.parse(self.get_data, {symbolize_names: true})
  data.map do |res|
    [res[:question], res[:options], res[:value]]
  end
end

def start_quiz(player)
  data = Gameplay.get_data
  prompt = TTY::Prompt.new
  data.each do |question|
    question[:prompt] = prompt.select(question.prompt, question.options)
    question[:answer] = question[:options].sample
    question[:value] = question[:value]
  end
end

def check_answer(input, answer, value)
  if input == answer
    puts "#{$input} is correct, you've earned #{$value}"
  else
    puts "#{$input} is incorrect!"
    puts "Thanks for playing"
    exit
  end
end

def announce_winner(username, question, quiz)
  if question == quiz.last
    puts "Congratulations #{$username}, You are a Ruby Millionaire"
  end
end
```

**Right Editor (Line 11 to Line 54):**

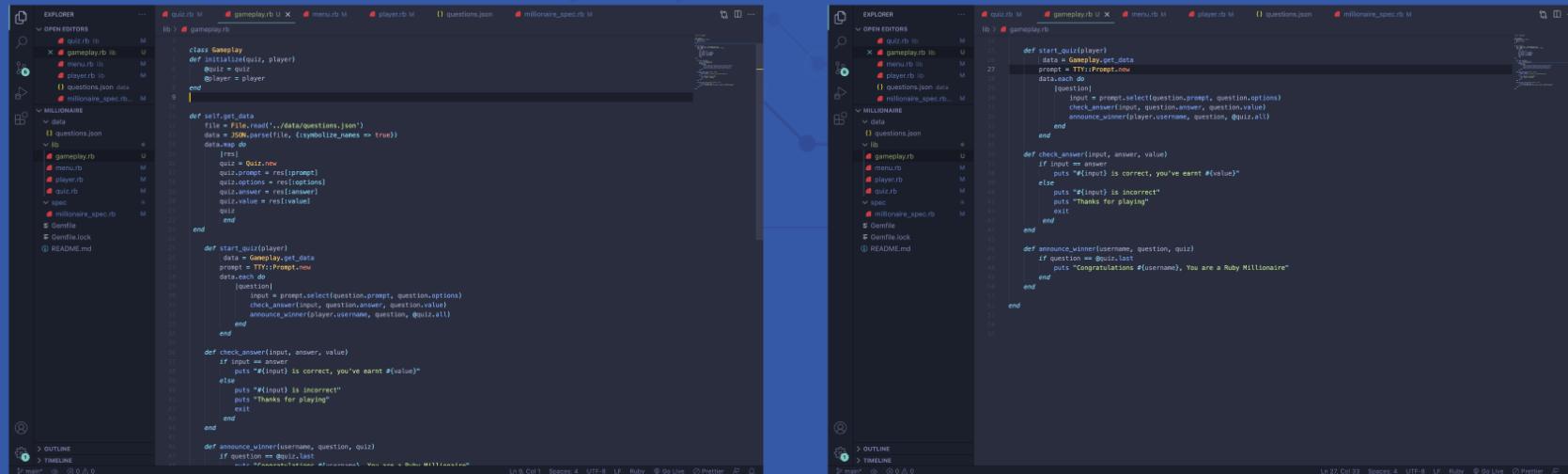
```
def start_quiz(player)
  data = Gameplay.get_data
  prompt = TTY::Prompt.new
  data.each do |question|
    question[:prompt] = prompt.select(question.prompt, question.options)
    question[:answer] = question[:options].sample
    question[:value] = question[:value]
  end
end

def check_answer(input, answer, value)
  if input == answer
    puts "#{$input} is correct, you've earned #{$value}"
  else
    puts "#{$input} is incorrect!"
    puts "Thanks for playing"
    exit
  end
end

def announce_winner(username, question, quiz)
  if question == quiz.last
    puts "Congratulations #{$username}, You are a Ruby Millionaire"
  end
end
```

# GAMEPLAY CLASS

Contains core methods to parse data, execute a quiz, allow the user to interact and determine outcomes.



```
class Gameplay
  def initialize(player)
    @quiz = quiz
    @player = player
  end

  def self.get_data
    File.read("../Data/questions.json")
  end

  data = JSON.parse(self.get_data, {symbolize_names: true})
  data.map do |res|
    [res[:question], res[:options], res[:value]]
  end
end

def start_quiz(player)
  data = Gameplay.get_data
  prompt = TTY::Prompt.new
  data.each do |question|
    question[:prompt] = prompt.select(question.prompt, question.options)
    check_answer(prompt.read, question.answer, question.value)
    announce_winner(player.username, question, @quiz.all)
  end
end

def check_answer(input, answer, value)
  if input == answer
    puts "#{input} is correct, you've earned #{value}"
  else
    puts "#{input} is incorrect!"
    puts "Thanks for playing"
    exit
  end
end

def announce_winner(username, question, quiz)
  if quiz.all == quiz.all
    puts "Congratulations #{username}, You are a Ruby Millionaire"
  end
end
```

The screenshot shows a code editor interface with a dark theme. On the left is the Explorer sidebar, which lists project files and folders. The main area displays a Ruby script named `gameplay.rb`. The script defines a `Gameplay` class with methods for initializing with a quiz and player, getting data from a JSON file, starting the quiz, checking answers, and announcing winners. It also includes a `check_answer` helper method and a `announce_winner` method that prints a congratulatory message.

```
quiz.rb M gameplay.rb U menu.rb M player.rb M questions.json millionaire_spec.rb M

EXPLORER
OPEN EDITORS
lib > gameplay.rb
3
4  class Gameplay
5    def initialize(quiz, player)
6      @quiz = quiz
7      @player = player
8    end
9
10   def self.get_data
11     file = File.read('../data/questions.json')
12     data = JSON.parse(file, {symbolize_names: true})
13     data.map do |res|
14       quiz = Quiz.new
15       quiz.prompt = res[:prompt]
16       quiz.options = res[:options]
17       quiz.answer = res[:answer]
18       quiz.value = res[:value]
19       quiz
20     end
21   end
22
23   def start_quiz(player)
24     data = Gameplay.get_data
25     prompt = TTY::Prompt.new
26     data.each do |question|
27       input = prompt.select(question.prompt, question.options)
28       check_answer(input, question.answer, question.value)
29     end
30   end
31
32   def check_answer(input, answer, value)
33     if input == answer
34       puts "#{input} is correct, you've earnt #{value}"
35     else
36       puts "#{input} is incorrect"
37       puts "Thanks for playing"
38       exit
39     end
40   end
41
42   def announce_winner(username, question, quiz)
43     if question == @quiz.last
44       puts "Congratulations, #username! You are a Ruby Millionaire!"
45     end
46   end
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
99
```

```
quiz.rb M gameplay.rb U menu.rb M player.rb M questions.json millionaire_spec.rb M

lib > gameplay.rb

def start_quiz(player)
  | data = Gameplay.get_data
  prompt = TTY::Prompt.new
  data.each do
    |question|
      input = prompt.select(question.prompt, question.options)
      check_answer(input, question.answer, question.value)
      announce_winner(player.username, question, @quiz.all)
    end
  end

def check_answer(input, answer, value)
  if input == answer
    puts "#{input} is correct, you've earnt #{value}"
  else
    puts "#{input} is incorrect"
    puts "Thanks for playing"
    exit
  end
end

def announce_winner(username, question, quiz)
  if question == @quiz.last
    puts "Congratulations #{username}, You are a Ruby Millionaire"
  end
end
```

EXPLORER

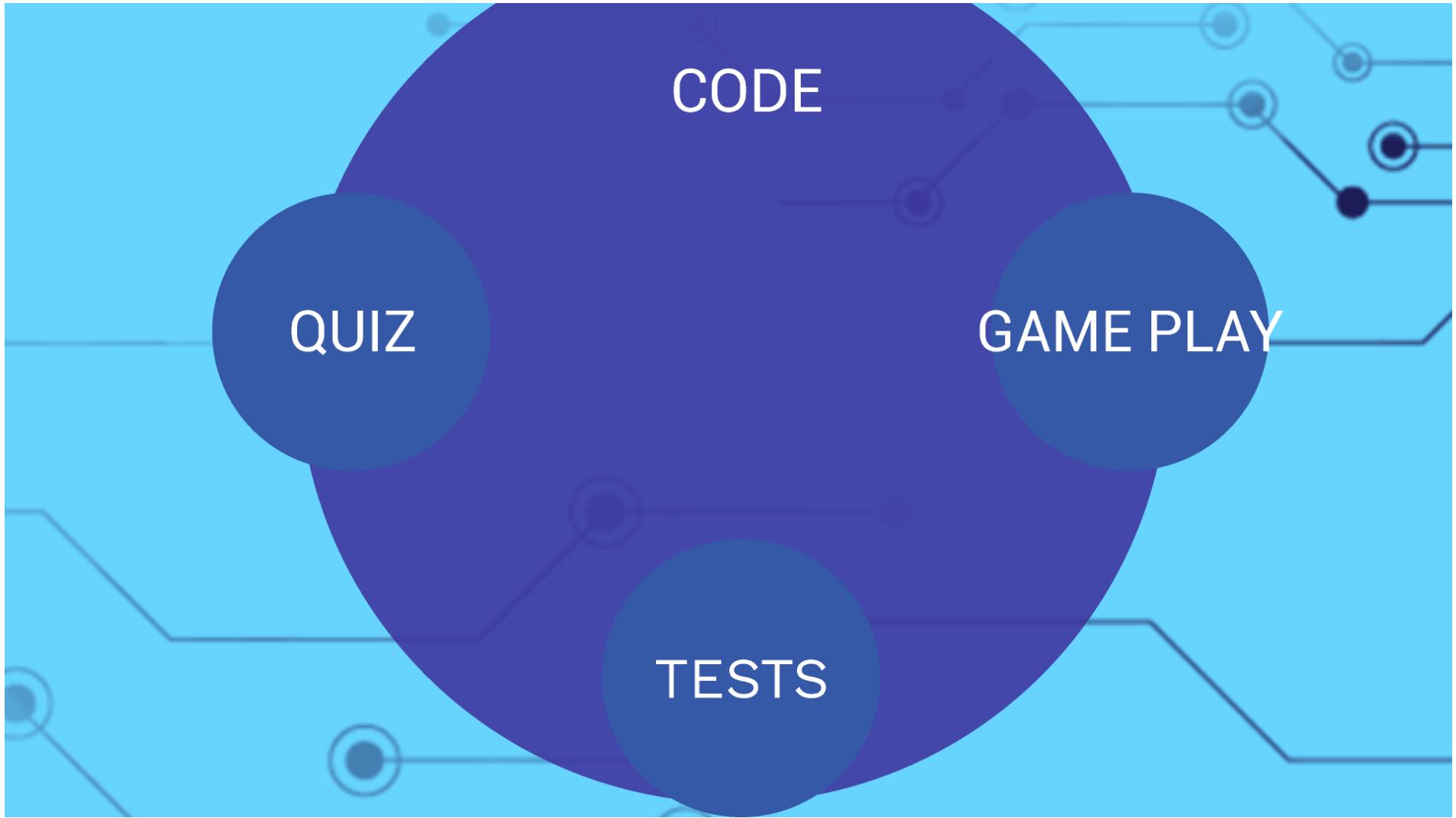
- OPEN EDITORS
  - quiz.rb lib M
  - gameplay.rb lib U
  - menu.rb lib M
  - player.rb lib M
  - questions.json data
  - millionaire\_spec.rb... M
- MILLIONAIRE
  - data
    - questions.json
  - lib
    - gameplay.rb U
    - menu.rb M
    - player.rb M
    - quiz.rb M
  - spec
    - millionaire\_spec.rb M
- Gemfile
- Gemfile.lock
- README.md

OUTLINE

TIMELINE

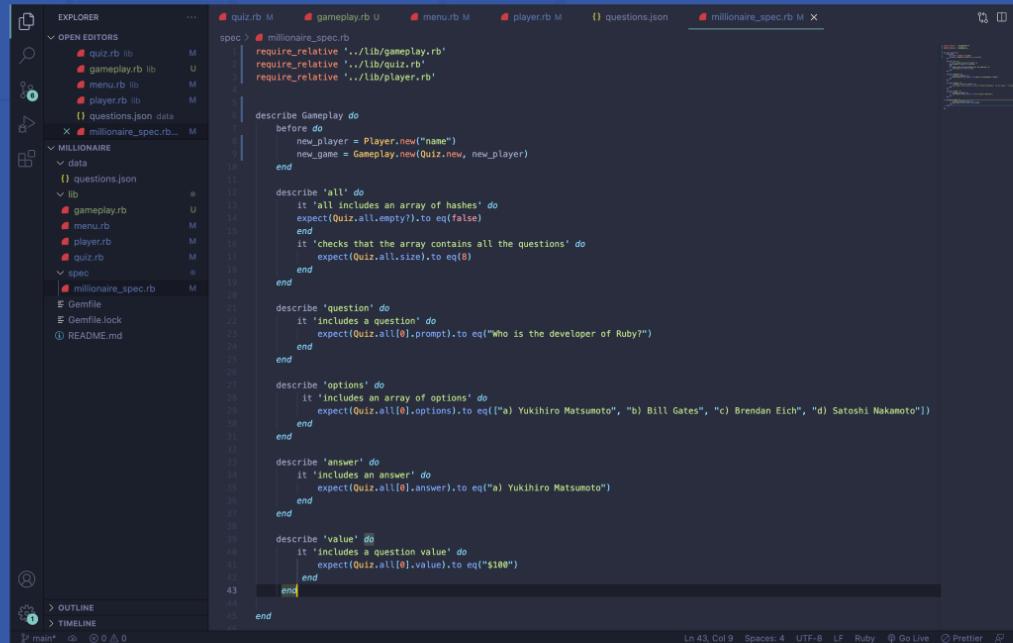
main\* ↻ 0 0 △ 0

Ln 27, Col 33 Spaces: 4 UTF-8 LF Ruby Go Live Prettier



# RSPEC

Tests were passing now they're not, to be continued.....



The screenshot shows a code editor interface with several tabs open. The tabs include: quiz.rb M, gameplay.rb U, menu.rb M, player.rb M, questions.json, and millionaire\_spec.rb M. The main editor area displays RSpec test code for a 'MILLIONAIRE' application. The code includes various describe blocks for 'Gameplay', 'question', 'options', 'answer', and 'value'. It uses expect statements to check array sizes and specific values like 'Yukihiro Matsumoto' and '\$100'. The code editor has a dark theme with syntax highlighting. On the left, there's an Explorer sidebar showing project files like quiz.rb, gameplay.rb, menu.rb, player.rb, questions.json, and Gemfile. Below the editor is an Outline panel.

```
require_relative '../lib/gameplay.rb'
require_relative '../lib/quiz.rb'
require_relative '../lib/player.rb'

describe Gameplay do
  before do
    new_player = Player.new("name")
    new_game = Gameplay.new(Quiz.new, new_player)
  end

  describe 'all' do
    it 'all includes an array of hashes' do
      expect(Quiz.all.empty?).to eq(false)
    end
    it 'checks that the array contains all the questions' do
      expect(Quiz.all.size).to eq(8)
    end
  end

  describe 'question' do
    it 'includes a question' do
      expect(Quiz.all[0].prompt).to eq("Who is the developer of Ruby?")
    end
  end

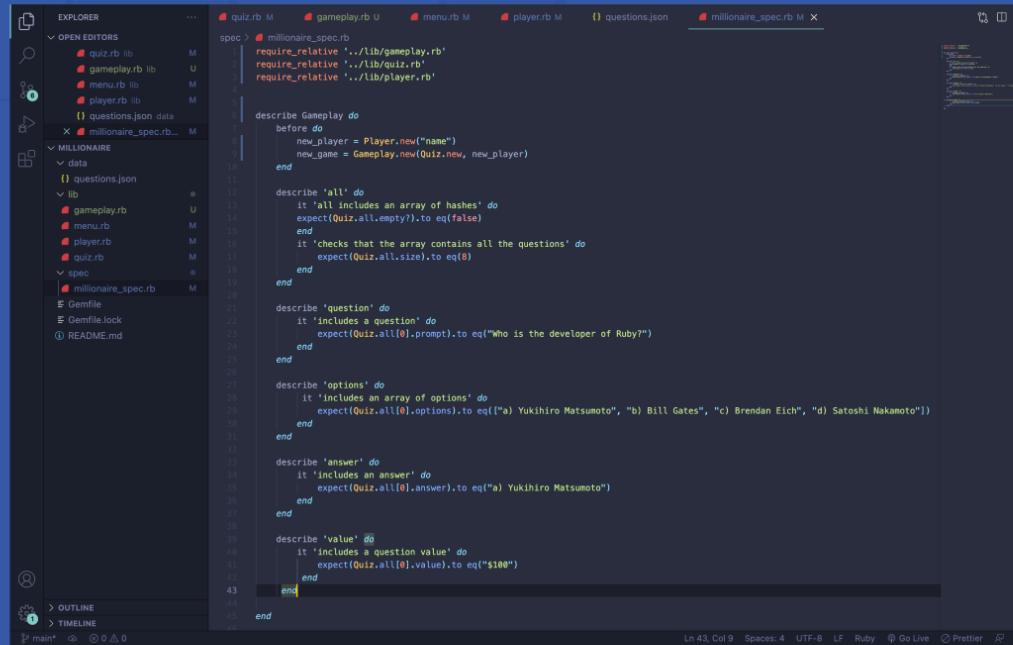
  describe 'options' do
    it 'includes an array of options' do
      expect(Quiz.all[0].options).to eq(["a) Yukihiro Matsumoto", "b) Bill Gates", "c) Brendan Eich", "d) Satoshi Nakamoto"])
    end
  end

  describe 'answer' do
    it 'includes an answer' do
      expect(Quiz.all[0].answer).to eq("a) Yukihiro Matsumoto")
    end
  end

  describe 'value' do
    it 'includes a question value' do
      expect(Quiz.all[0].value).to eq("$100")
    end
  end
end
```

# RSPEC

Tests were passing now they're not, to be continued.....



The screenshot shows a code editor interface with several files open in tabs at the top: quiz.rb, gameplay.rb, menu.rb, player.rb, questions.json, and millionaire\_spec.rb. The millionaire\_spec.rb file is the active editor, displaying RSpec test code. The code defines various describe blocks for Quiz, Gameplay, and Answer classes, with expect statements and assertions. The left sidebar shows a file tree for a project named 'MILLIONAIRE' containing files like quiz.rb, gameplay.rb, menu.rb, player.rb, questions.json, lib, spec, Gemfile, and README.md. The bottom status bar indicates the code is in Line 43, Column 9, with 4 spaces, in UTF-8 encoding, and is a Ruby file.

```
require_relative '../lib/gameplay.rb'
require_relative '../lib/quiz.rb'
require_relative '../lib/player.rb'

describe Gameplay do
  before do
    new_player = Player.new("name")
    new_game = Gameplay.new(Quiz.new, new_player)
  end

  describe 'all' do
    it 'all includes an array of hashes' do
      expect(Quiz.all.empty?).to eq(false)
    end

    it 'checks that the array contains all the questions' do
      expect(Quiz.all.size).to eq(8)
    end
  end

  describe 'question' do
    it 'includes a question' do
      expect(Quiz.all[0].prompt).to eq("Who is the developer of Ruby?")
    end
  end

  describe 'options' do
    it 'includes an array of options' do
      expect(Quiz.all[0].options).to eq(["a) Yukihiro Matsumoto", "b) Bill Gates", "c) Brendan Eich", "d) Satoshi Nakamoto"])
    end
  end

  describe 'answer' do
    it 'includes an answer' do
      expect(Quiz.all[0].answer).to eq("a) Yukihiro Matsumoto")
    end
  end

  describe 'value' do
    it 'includes a question value' do
      expect(Quiz.all[0].value).to eq("$100")
    end
  end
end
```

The screenshot shows a code editor interface with a dark theme. The left sidebar contains a tree view of files and folders:

- OPEN EDITORS: quiz.rb (M), gameplay.rb (U), menu.rb (M), player.rb (M), questions.json (D), millionaire\_spec.rb (M).
- MILLIONAIRE folder:
  - data folder: questions.json.
  - lib folder: gameplay.rb (U), menu.rb (M), player.rb (M), quiz.rb (M).
  - spec folder: millionaire\_spec.rb (M).
- Other files: Gemfile, Gemfile.lock, README.md.

The main editor area displays the content of millionaire\_spec.rb:

```
require_relative '../lib/gameplay.rb'
require_relative '../lib/quiz.rb'
require_relative '../lib/player.rb'

describe Gameplay do
  before do
    new_player = Player.new("name")
    new_game = Gameplay.new(Quiz.new, new_player)
  end

  describe 'all' do
    it 'all includes an array of hashes' do
      expect(Quiz.all.empty?).to eq(false)
    end
    it 'checks that the array contains all the questions' do
      expect(Quiz.all.size).to eq(8)
    end
  end

  describe 'question' do
    it 'includes a question' do
      expect(Quiz.all[0].prompt).to eq("Who is the developer of Ruby?")
    end
  end

  describe 'options' do
    it 'includes an array of options' do
      expect(Quiz.all[0].options).to eq(["a) Yukihiro Matsumoto", "b) Bill Gates", "c) Brendan Eich", "d) Satoshi Nakamoto"])
    end
  end

  describe 'answer' do
    it 'includes an answer' do
      expect(Quiz.all[0].answer).to eq("a) Yukihiro Matsumoto")
    end
  end

  describe 'value' do
    it 'includes a question value' do
      expect(Quiz.all[0].value).to eq("$100")
    end
  end
end
```

At the bottom of the editor, status bar items include: Ln 43, Col 9, Spaces: 4, UTF-8, LF, Ruby, Go Live, Prettier, and a few icons.