Karl Kemister-Sheppard T1A3
Terminal Application

Overview

What is **Dragon's Lair**

Adventure text based game that allows the user to create a character from a selection of choices.
Once the user creates their character and character name they are then faced with a series of challenges and tests that will either end in victory or be their last journey.

Game difficulty: **Medium => Hard**

- **7x** death traps based on wrong path chosen
- **2x** paths to win which is broken into two parts as below: (**1**x true path and **1**x path that leads to bonus (To be continued) message)

Features

Variables and Global Variables: The application uses a wide range of local variables and global variables throughout the program in order to DRY up code and reuse particular functions.

Loops: While loops where used in conjunction with the tty-prompt to ensure the user could create their character.

Control Structures: Where used to create multiple guessing games to unlock the door to the lair

Conditional Statements: Used extensively to control the flow and path directions for the users adventure (==, &&, !=).

User Input: The app requires the user to input text (gets.chomp) in order to control the path/choice direction to progress within the game.

ARGV: Added to the command line argument to the index.rb to run the game_title & User name if manually added to the bash.sh file.

Bonus Feature: AfPlay = Command Line MP3 Player -which allows you to run audio files from the Terminal.
Special instructions: To kill the sound at any stage (Control+C) or allow it to run until the audio has run its timer.
afplay

Ruby Gems

1. gem "colorize", "~> 0.8.1" [Source]
   Colorize was used extensively throughout the program to help identify when key objects where being used. For example when the user types in get.chomp command anytime the variable is called it is printed to screen in .magenta

2. gem "tty-prompt", "~> 0.23.1" [Source]
   TTY-prompt has been used to create the character selection process for the program, future development of the app will utilize the tty-prompt more extensively to help minimize error handling.

3. gem "tty-progressbar", "~> 0.18.2" [source]
   TTY-progressbar is used specifically for progress actions points, such as downloading your adventure, or when the user inserts their keys from the guessing game to unlock the door to the lair.

4. gem 'emoji_regex' [Source]
   Emoji Regex has been used throughout the game to add some visuals for the user to see. I wanted to add some color and shapes To the program to make it more fun and interactive when played.

Logical Operators

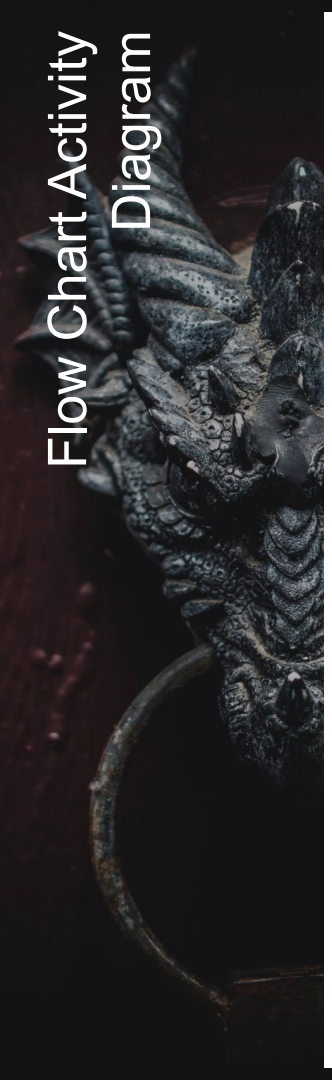
Logical Operators:

The application re-uses the main logic operators **== != &&** constantly throughout the program to check if the user has selected the correct potion in order to win their survival.

To enhance the complexity of the program, additional operators could be added to create new conditions if the user chooses a specific character, potion or even a damage score count if this was added to the application at a **later date**.

```
if second_chance == "yes"
        puts "You bend the knee and drop the #{$tool}"
        $process_id = spawn "afplay -v 0.2 item_drop.mp3"
        puts "and ask for forgiveness from the dragon"
        puts "The dragon accepts your offer and allows you to live"
        puts "\n""\n""\n"
        sleep(3)
        system "clear"
        puts $thank_you
        Process.kill("SIGKILL", $process_id)
        $process_id = spawn "afplay -v 0.2 fail_effect.mp3"
        system("killall afplay")
        return
    elsif second_chance == "no"
        puts "The dragon swoops in and eats you #{$death_emoji} (Restart Game)"
        system "clear"
        puts $try_again
        Process.kill("SIGKILL", $process_id)
        $process_id = spawn "afplay -v 0.2 fail_effect.mp3"
        system("killall afplay")
        return

    elsif user_choice_attack== "no"
        puts "You drop the #{$tool} on the floor and tell the dragon you are there to offer your protection"
        puts "The dragon sits back and agrees to the offer, and allows you safe passage in and out of the lair"
        sleep(3)
        system "clear"
        puts $thank_you
        Process.kill("SIGKILL", $process_id)
        $process_id = spawn "afplay -v 0.2 fail_effect.mp3"
        system("killall afplay")
        return
end
```

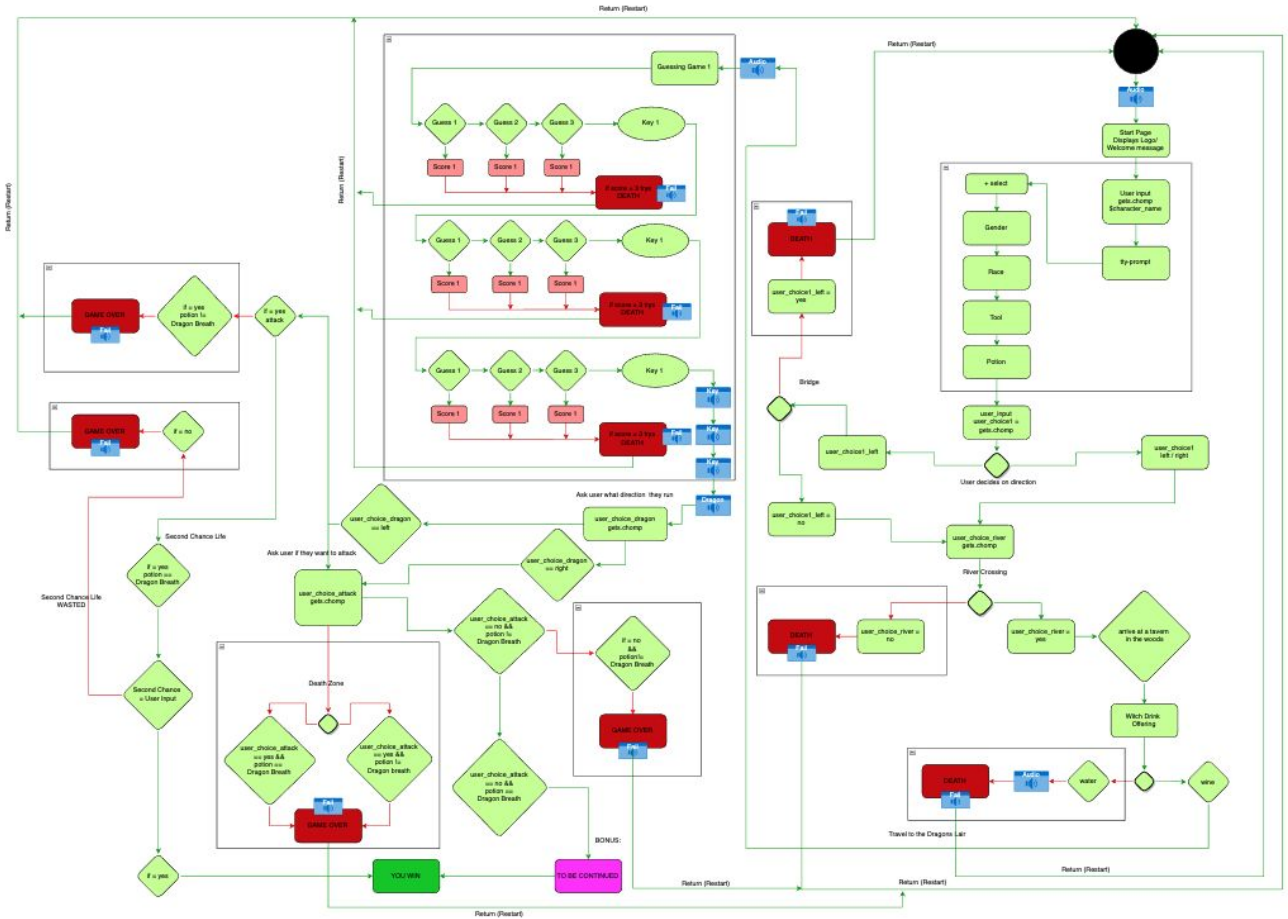


Flow Chart Activity Diagram

Dragons Lair

Adventure Survival Game

Karl Kemister-Sheppard 18.04.2022



Control Structures

Feature 1

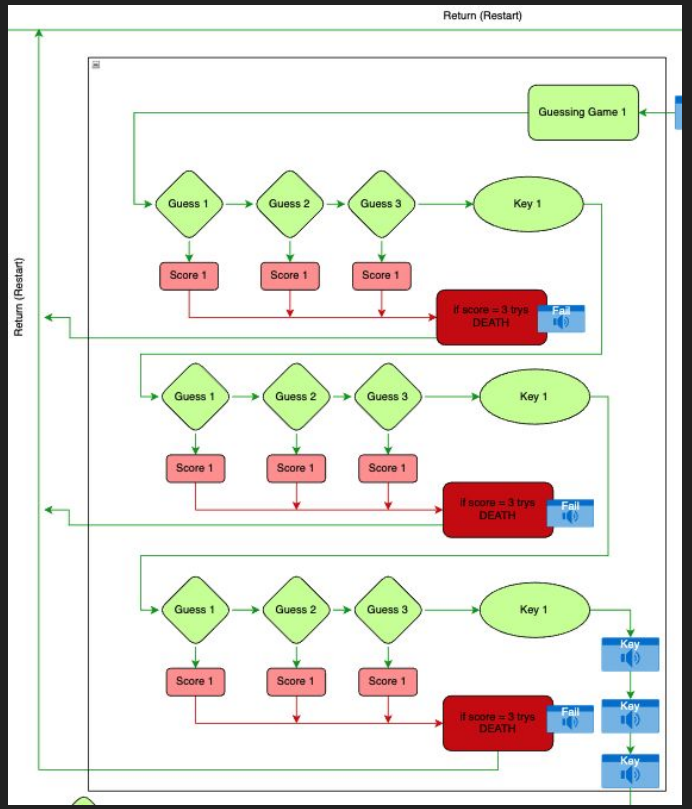
Variables: Variables are used to store guess count from user.

While Loops: Will be used within the program to iterate through the score count until the user either fails 3 times or answers the riddle correctly.

$Global Variables: Used to store text object images in global.rb to reduce code congestion when user fails.

Conditional Structure: Used to check if the users guess is != the correct guess and also check if the user is != out of guesses.

Error Handling: COMING SOON



Tty-prompt gem

Feature 2

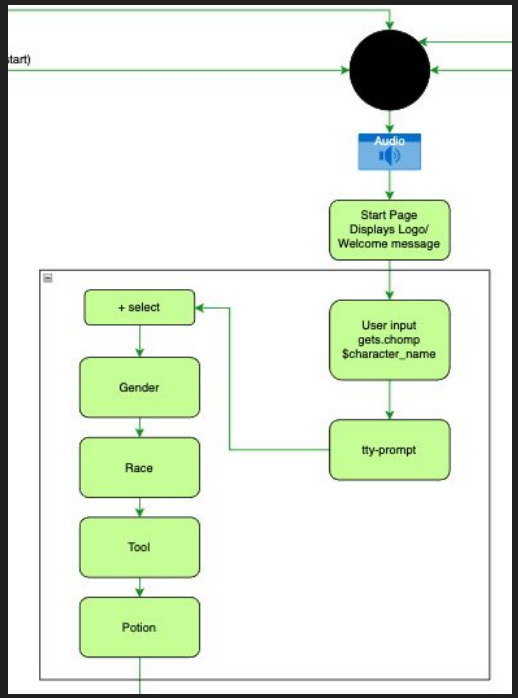
User Inputs Guess: gets.chomp was used to save the $character_name

Variables: Variables are saved inside def Methods to store $prompt.selected object for the array list.

$Global Variables: Used to save the users unique name so it could be used with string interpolation throughout the code.
In addition $global **Emoji codes** used in conjunction with ruby gem emoji regex to add additional features.

While Loops: Used to cycle through the tty-prompt arrays so the user could select from the options provided until the user selected Exit

Error Handling: While loop added to check if the input was true, meaning that while .empty? User will be notified to add input



Variables and Variable Scope.

Feature 3

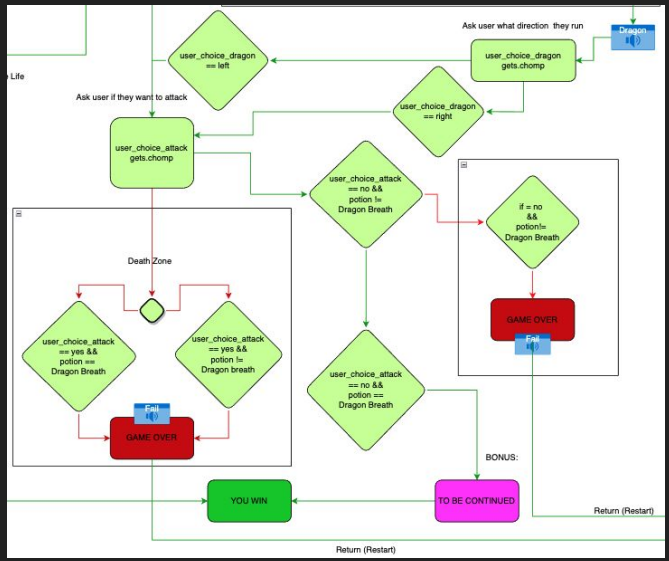
User Inputs Guess: gets.chomp

Variables: Variables are used to store guess count from user.

$Global Variables: Used to save the users $tool and $potion value so it could be used with string interpolation during adventure journey..

Conditional Statements: Used extensively to control the flow and path directions for the users adventure specifically (==, &&, !=). These were used to check the potion selection and user gets.chomp variable to check the conditions where true or false to achieve the desired path.

Error Handling: COMING SOON



Feature BONUS

Bonus Feature: AfPlay

= Command Line MP3 Player

$Global Variables: Used to save the volume control setting, afplay system call function and file.mp3/wav.

Return: Function is required when using this as once the command line is called to use main audio line, you have to initiate the kill commands if you wish to end process.

Command+C = Terminates Audio in Command Line.

Error Handling: COMING SOON

Example:

$global var saved, (-v is volume control range is (0.1=>255)

```
$process_id = spawn "afplay -v 0.2 It_Is_Coming_-_David_Fesliyan.mp3"
```

Kill Code:

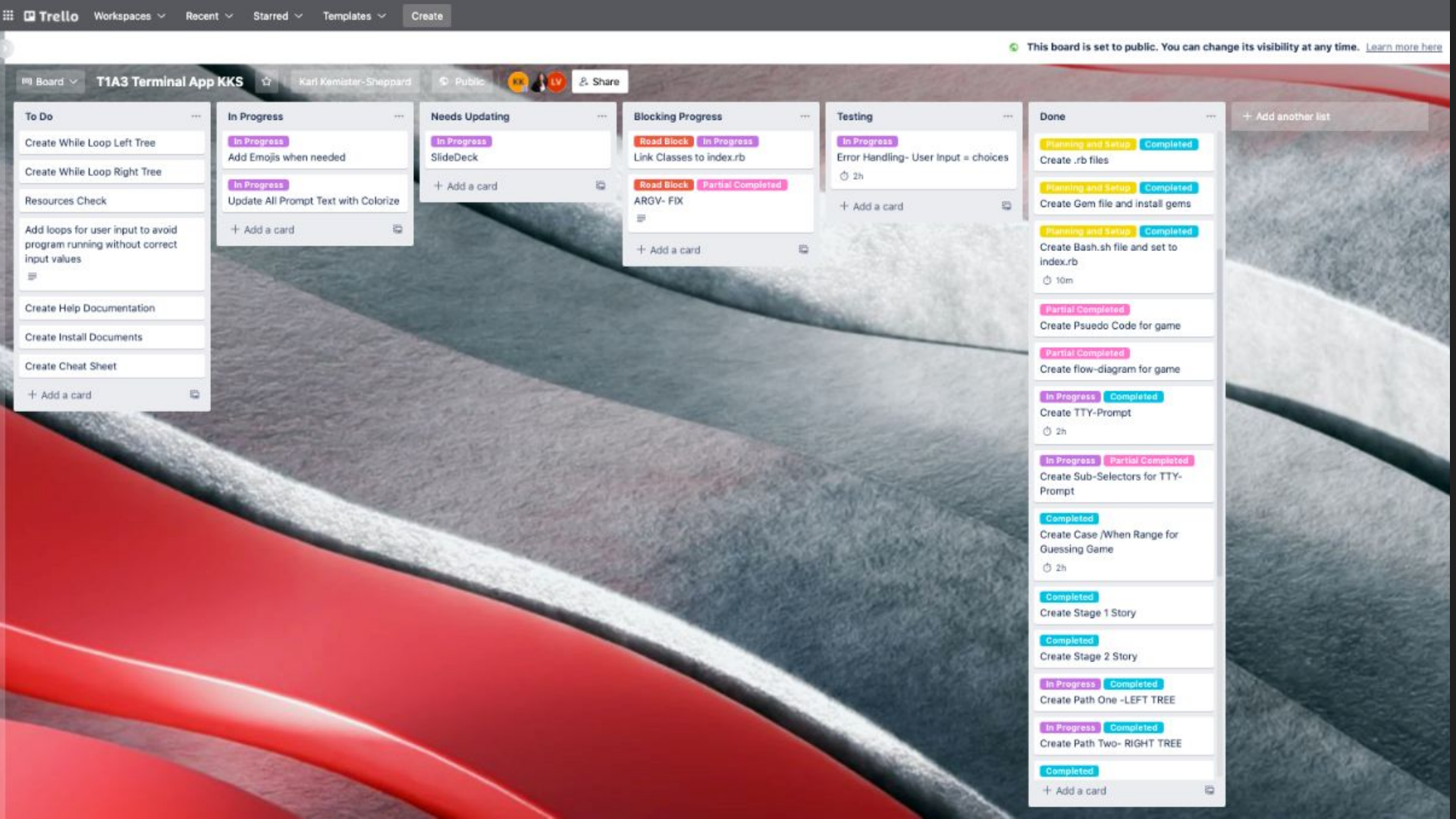
This is specific for the particular audio file.

```
Process.kill("SIGKILL", $process_id)
```

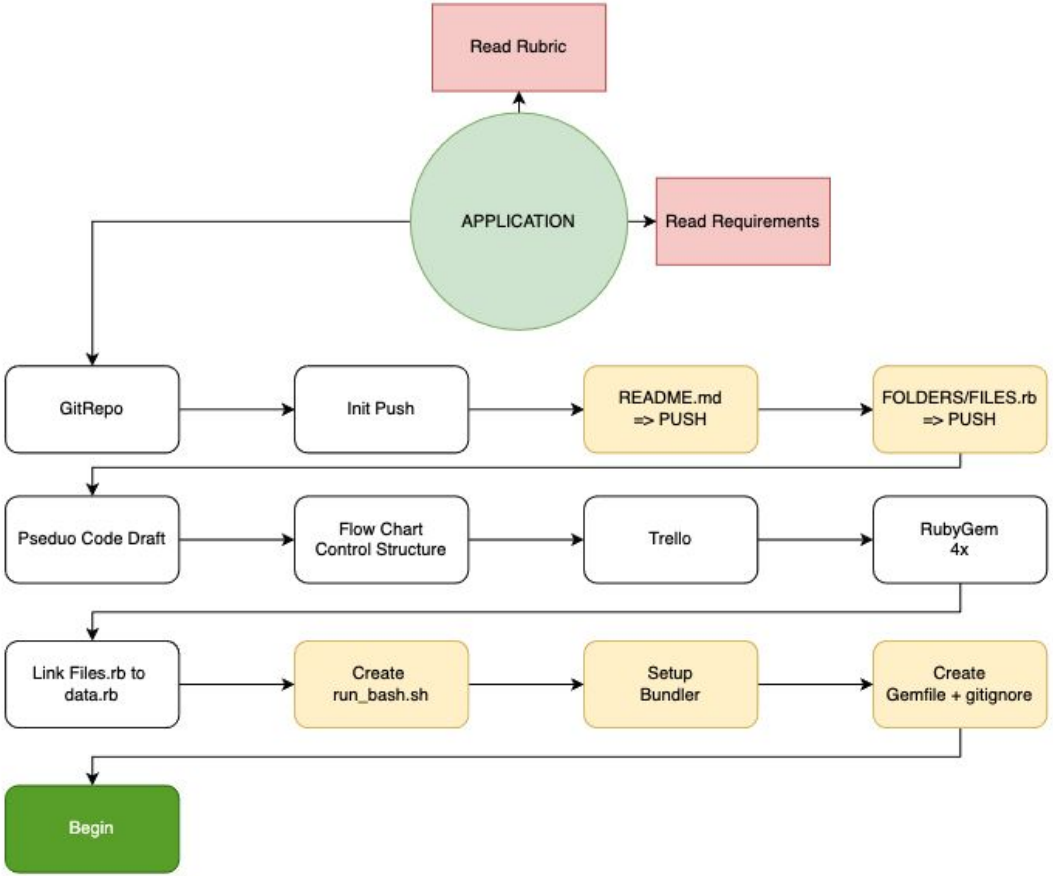
Kill Command:

Entire system when program is returned.

```
system("killall afplay")
```


Trello
Workspaces
Recent
Starred
Templates
Create
This board is set to public. You can change its visibility at any time. Learn more here
Board
T1A3 Terminal App KKS
Kari Kemister-Sheppard
Public
Share
To Do
Create While Loop Left Tree
Create While Loop Right Tree
Resources Check
Add loops for user input to avoid program running without correct input values
Create Help Documentation
Create Install Documents
Create Cheat Sheet
Add a card
In Progress
In Progress
Add Emojis when needed
In Progress
Update All Prompt Text with Colorize
Add a card
Needs Updating
In Progress
SlideDeck
Add a card
Blocking Progress
Road Block
In Progress
Link Classes to index.rb
Road Block
Partial Completed
ARGV- FIX
Add a card
Testing
In Progress
Error Handling- User Input = choices
2h
Add a card
Done
Planning and Setup
Completed
Create .rb files
Planning and Setup
Completed
Create Gem file and install gems
Planning and Setup
Completed
Create Bash.sh file and set to index.rb
10m
Partial Completed
Create Psuedo Code for game
Partial Completed
Create flow-diagram for game
In Progress
Completed
Create TTY-Prompt
2h
In Progress
Partial Completed
Create Sub-Selectors for TTY-Prompt
Completed
Create Case /When Range for Guessing Game
2h
Completed
Create Stage 1 Story
Completed
Create Stage 2 Story
In Progress
Completed
Create Path One -LEFT TREE
In Progress
Completed
Create Path Two- RIGHT TREE
Completed
Add a card
Add another list

DEVELOPMENT PROCESS PLAN



Error Handling

Error handling for the TTY-Prompt is basically dealt with by the Development team that created and maintain the Gem. It forces the user to select from a list of items presented on the display and in this case it provides no option to exit or bypass the prompt.select.

```
def potion_selection
   require "tty-prompt"
   $prompt = TTY::Prompt.new
   choice = %w(Wolf-Blood Vital-Essence Oort-Brew Soul-Dealer
Dragon-Breath)
   $potion = $prompt.select("Select your characters potion
#{$character_name}", choice)
end
```

Error handling: This example provided as used extensively throughout the program to handle the exit! Feature for the end points. When the user its a **TRY AGAIN** prompt or finds the **EXIT** the exit! Will execute first giving the program time to clean up- then it will run SystemExit to terminate the program.

```
begin
     exit!
rescue SystemExit
     p 123
end
return
```

Dry Code Practices

Code styling for the Terminal Application is designed to be simplistic, clean and easy to read.

General Overview

Wrap majority of code in separate def methods and store them in method.rb file

Add #HEADING and trailing #END around all methods in index.rb, methods.rb files for easy navigation.

Create local and $global variables to reduce the amount of code written and in addition i've grouped code blocks into def methods that can be reused across the entire application and read via the data.rb file that was created to talk to the other files.

Challenges

Classes and instance Variable Scope: was a major challenge- still working on a solution to try and use the classes that were created to allow the user to see what benefits (arguments) that where assigned to the character. At this stage the classes are not being used.

Conditional Statements: Creating the second chance was one of the more difficult statements as it required multiple links, called on variables and tty-prompt to allow the structure/ functions to work without errors.

Adding Audio: This required extensive research online as its not a function that is used much in the Terminal environment. After experimenting with various methods, samples online i was able to put together the required code to create a immersive ambience experience with the user.

Writing something from Scratch: This was the biggest challenge to date, with limited experience, prior understanding of Ruby or any other Object Oriented Languages, coming up with a working application that worked without errors was the hardest challenge.

Ethical Issues

HSP's: More commonly known as highly sensitive people, could find the application overstimulating, and may be triggered buy the sounds effects and repeating loops that become traumatising if excessive repeat play is executed in one sitting.

Humane: Some users may feel like this application is designed to inflict harm on animals, mythical creatures and not use/participate in the application for those reasons.

Misdirection: Tricking the user into thinking the application was designed to slaughter the mystical dragon.

Favourite’s

Riddles: Adding the riddles inside the application was a great way to slow down the user. It is one of the main features that was designed to increase the game difficulty and test the user mentally.

Audio: This would have to be one of my greatest achievements, the research and testing phase was well worth the end result when i heard the dragon for the first time.

Flow-Chart: Considering how complex the flow-chart is, i enjoyed the logical and mental challenge for creating such a application. Being able to build a app that includes so many paths, death traps and loops was a great way to test my current ability.

THANK YOU FOR PLAYING DRAGON'S LAIR