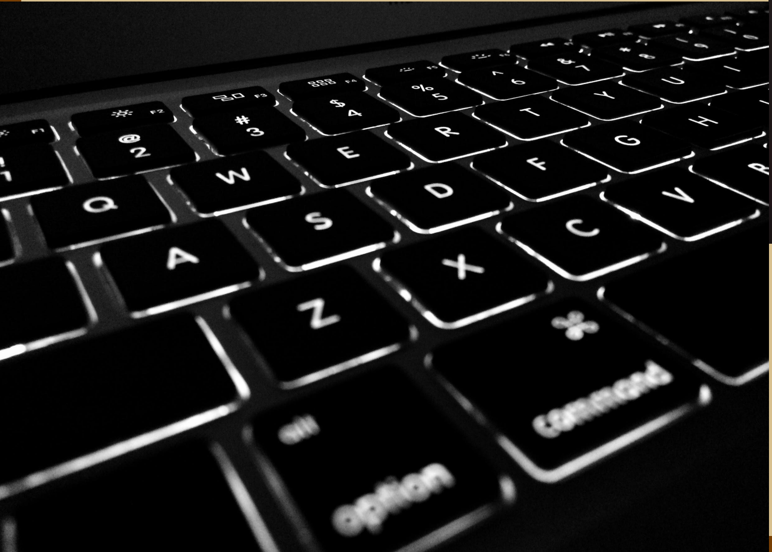
T1A3



Terminal_App

Educational...

I like games, but they don’t always have to be a waste of time...

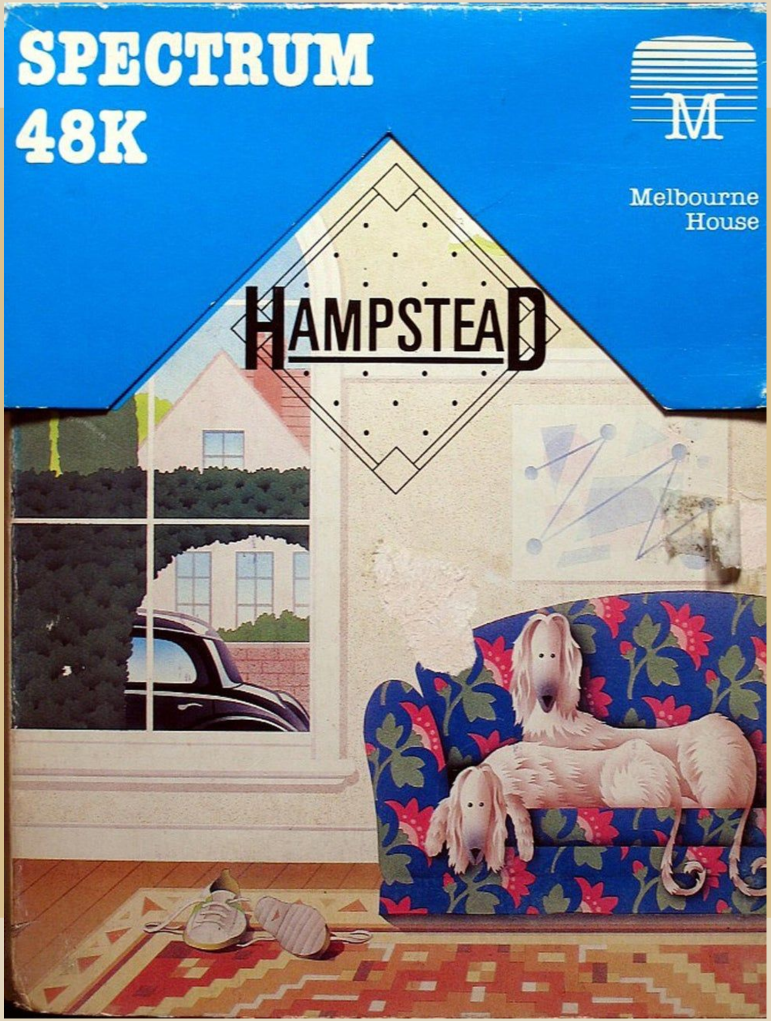
...they can also be...

...educational.

(I know....wot?!?)

Text based

Terminal is natively a text based display, and without learning a whole games library like Gosu, we have little options...



RPSG

...or the Role Playing Spelling Game.

At its heart it’s a memory game.

But it’s also a role playing game...

How the &*@! does that work?

Well, you get to pick a character:

- Barbarian
- Wizard
- Thief

And then you defeat the enemy by "**cprrectly speling**' as many words as you can. (see what I did there?)

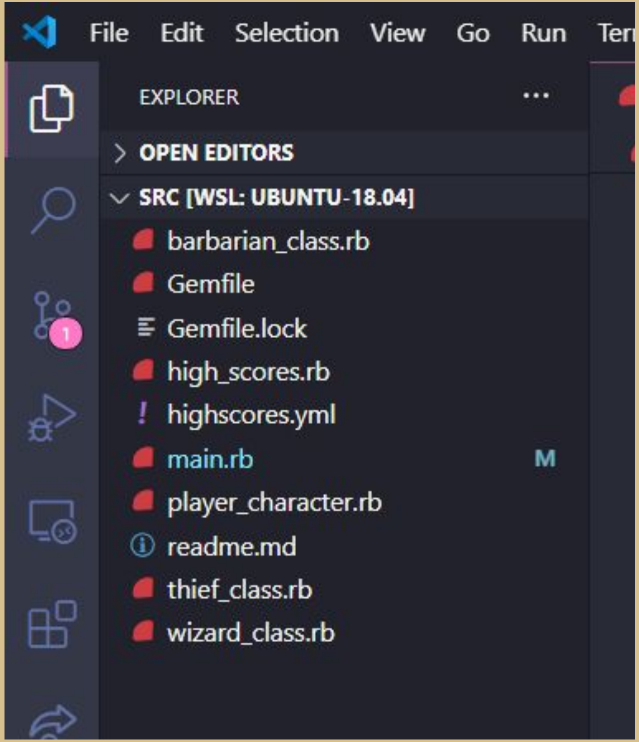
(plus special abilities!)

A closer look...

What's involved?

- Main Ruby file
- Parent character class
  - Child character classes
- Leaderboard class
- And a YAML file

This structure explores classes and inheritance, writing and reading to file and an effort to simplify methods as logically as possible



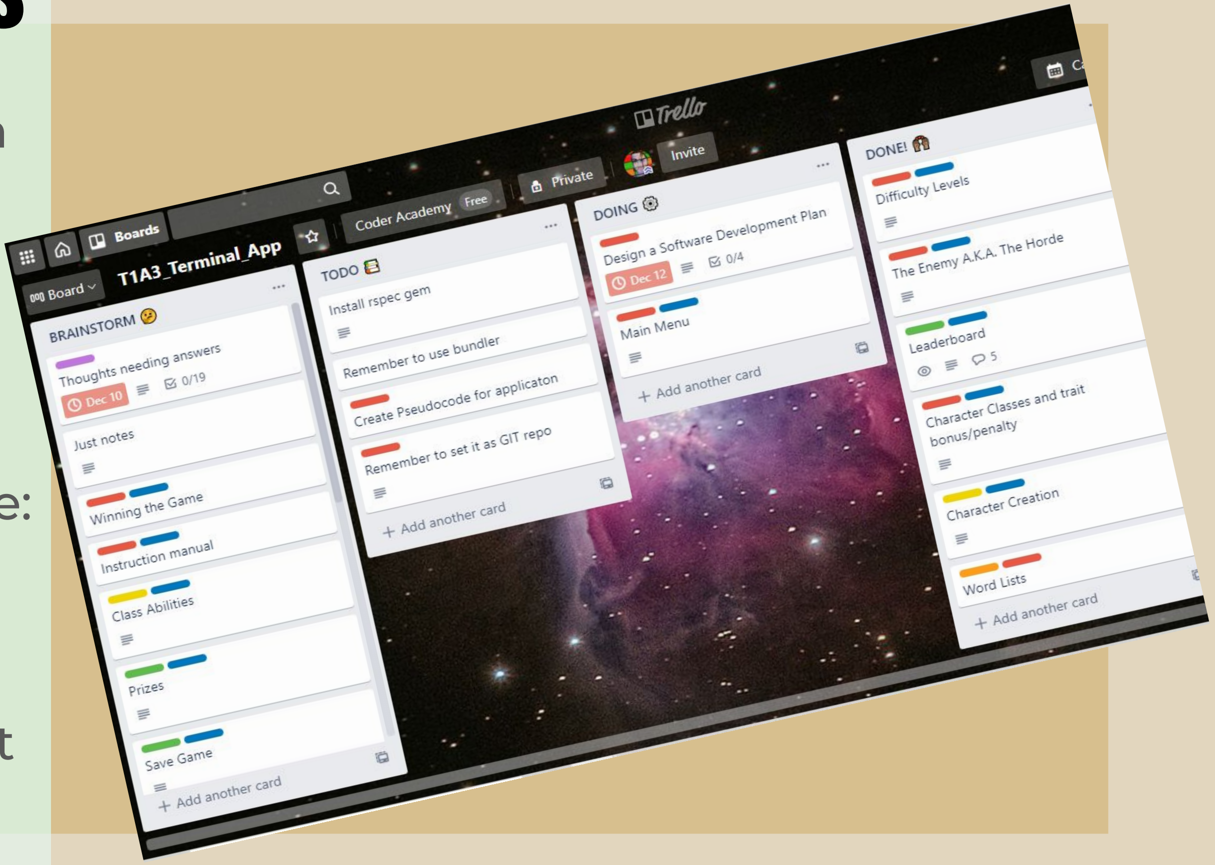
```
barbarian_class.rb
Gemfile
Gemfile.lock
high_scores.rb
highscores.yml
main.rb
player_character.rb
readme.md
thief_class.rb
wizard_class.rb
```

Program Features

- Character creation
- Difficulty levels
- Special Abilities
- Leaderboard
- Replayability

Stretch goals for future:

- Save Game
- Save character
- Special events/loot



BRAINSTORM
- Thoughts needing answers (Dec 10, 0/19)
- Just notes
- Winning the Game
- Instruction manual
- Class Abilities
- Prizes
- Save Game
- + Add another card

TODO
- Install rspec gem
- Remember to use bundler
- Create Pseudocode for applicaton
- Remember to set it as GIT repo
- + Add another card

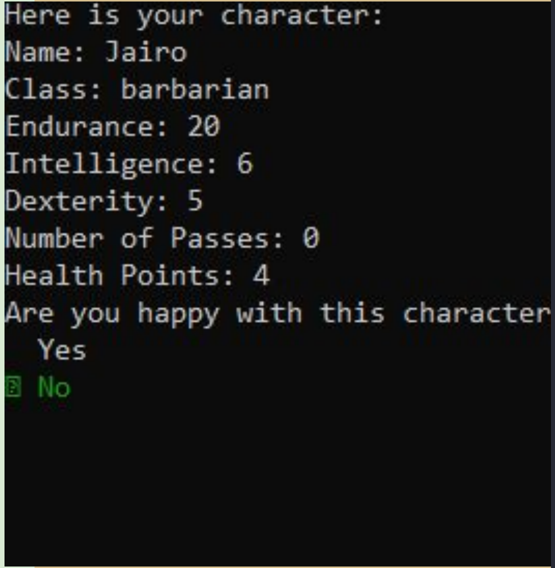
DOING
- Design a Software Development Plan (Dec 12, 0/4)
- Main Menu
- + Add another card

DONE!
- Difficulty Levels
- The Enemy A.K.A. The Horde
- Leaderboard (5)
- Character Classes and trait bonus/penalty
- Character Creation
- Word Lists
- + Add another card

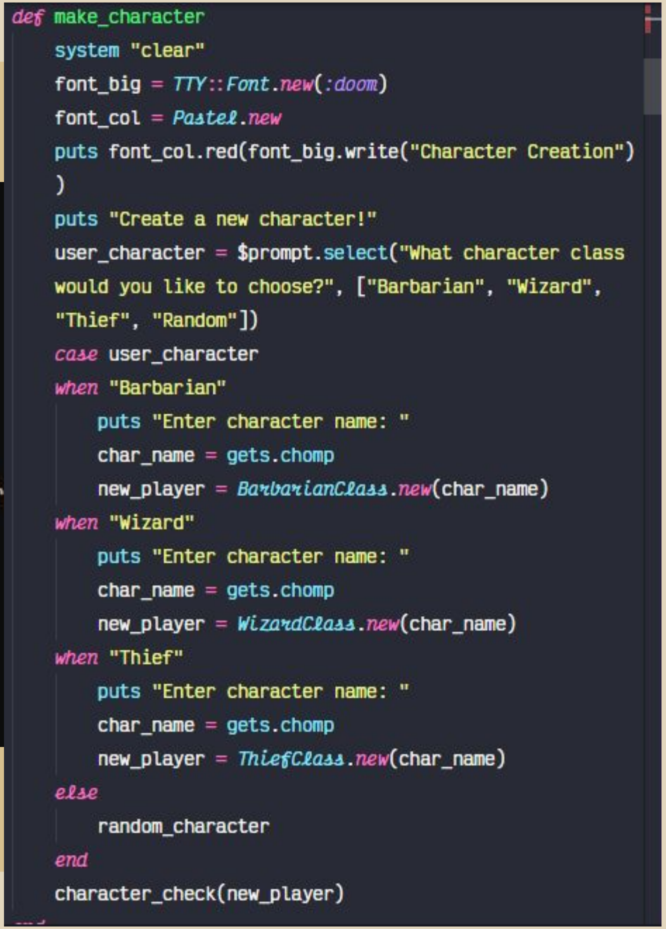
Character Creation

Player can pick a class, or go for a random character.

They are presented with the character sheet and have the option to reroll their character if unhappy.



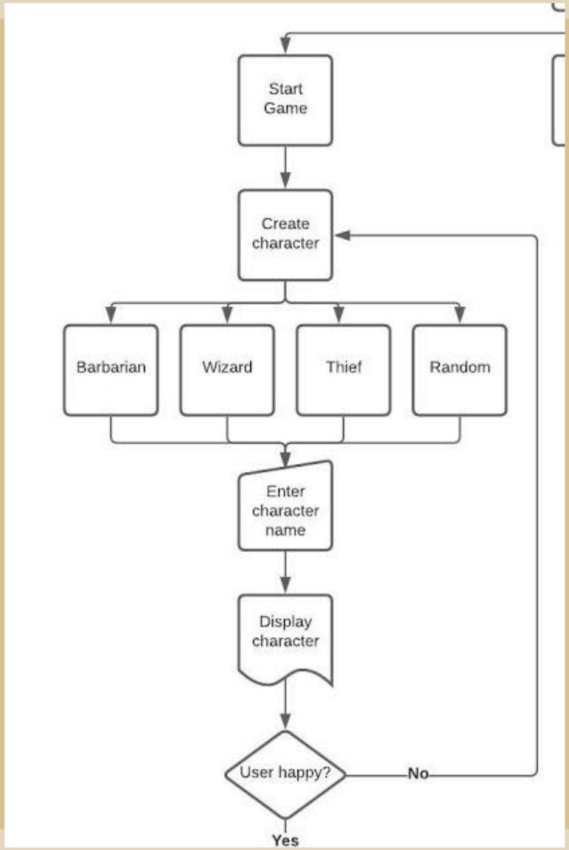
```
Here is your character:
Name: Jairo
Class: barbarian
Endurance: 20
Intelligence: 6
Dexterity: 5
Number of Passes: 0
Health Points: 4
Are you happy with this character
  Yes
‣ No
```



```ruby
def make_character
  system "clear"
  font_big = TTY::Font.new(:doom)
  font_col = Pastel.new
  puts font_col.red(font_big.write("Character Creation")
  )
  puts "Create a new character!"
  user_character = $prompt.select("What character class
  would you like to choose?", ["Barbarian", "Wizard",
  "Thief", "Random"])
  case user_character
  when "Barbarian"
    puts "Enter character name: "
    char_name = gets.chomp
    new_player = BarbarianClass.new(char_name)
  when "Wizard"
    puts "Enter character name: "
    char_name = gets.chomp
    new_player = WizardClass.new(char_name)
  when "Thief"
    puts "Enter character name: "
    char_name = gets.chomp
    new_player = ThiefClass.new(char_name)
  else
    random_character
  end
  character_check(new_player)
```

Logic in creation

The beauty in creating a character, is that you don't have to stick with it. You can give into the randomness of reroll until you have something you'd like to play with...



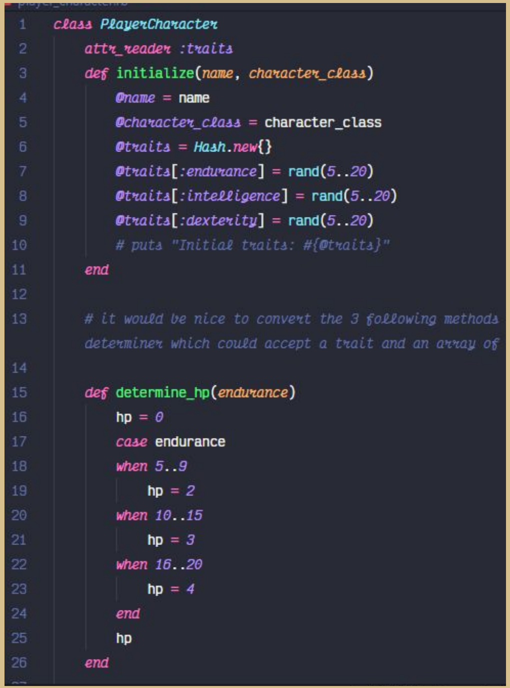
Character objects

All characters have similar traits but each “class” has something personal to them.

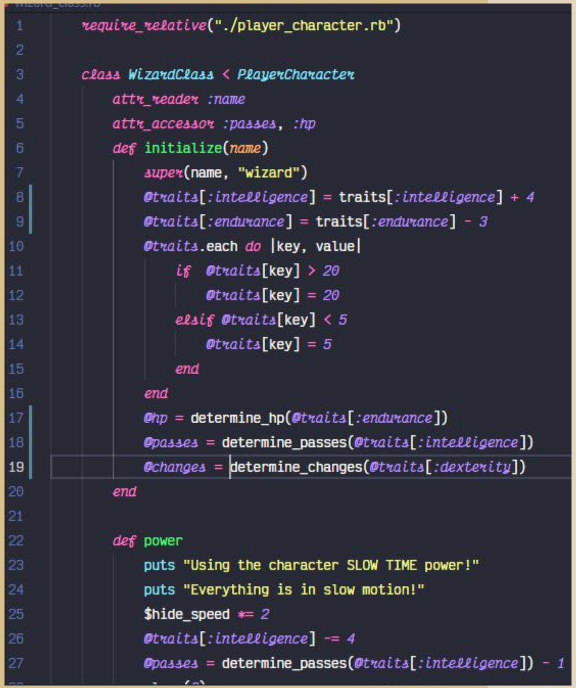
Barbarians have a higher endurance, for example, and therefore have more health points.

Each class also has its own special ability:

- Barbarian - BIG STRIKE
- Wizard - SLOW TIME
- THIEF - BE SNEAKY



```ruby
class PlayerCharacter
    attr_reader :traits
    def initialize(name, character_class)
        @name = name
        @character_class = character_class
        @traits = Hash.new{}
        @traits[:endurance] = rand(5..20)
        @traits[:intelligence] = rand(5..20)
        @traits[:dexterity] = rand(5..20)
        # puts "Initial traits: #{@traits}"
    end

    # it would be nice to convert the 3 following methods
    determiner which could accept a trait and an array of

    def determine_hp(endurance)
        hp = 0
        case endurance
        when 5..9
            hp = 2
        when 10..15
            hp = 3
        when 16..20
            hp = 4
        end
        hp
    end
```

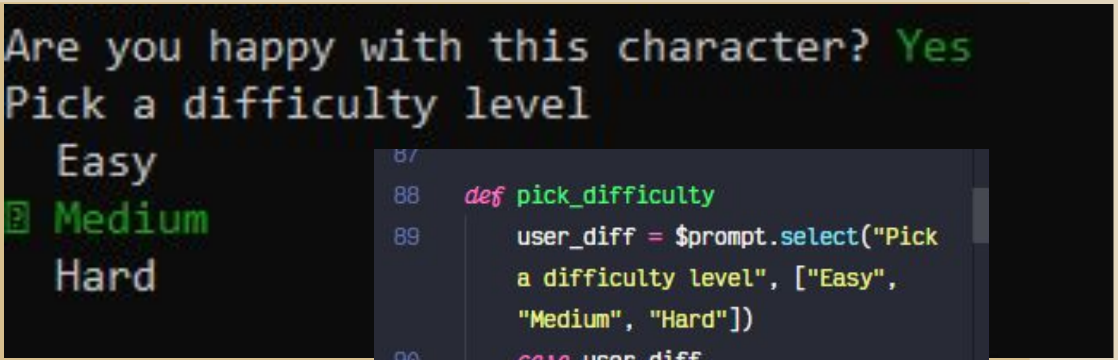


```ruby
    require_relative("./player_character.rb")

    class WizardClass < PlayerCharacter
        attr_reader :name
        attr_accessor :passes, :hp
        def initialize(name)
            super(name, "wizard")
            @traits[:intelligence] = traits[:intelligence] + 4
            @traits[:endurance] = traits[:endurance] - 3
            @traits.each do |key, value|
                if  @traits[key] > 20
                    @traits[key] = 20
                elsif @traits[key] < 5
                    @traits[key] = 5
                end
            end
            @hp = determine_hp(@traits[:endurance])
            @passes = determine_passes(@traits[:intelligence])
            @changes = determine_changes(@traits[:dexterity])
        end

        def power
            puts "Using the character SLOW TIME power!"
            puts "Everything is in slow motion!"
            $hide_speed *= 2
            @traits[:intelligence] -= 4
            @passes = determine_passes(@traits[:intelligence]) - 1
```

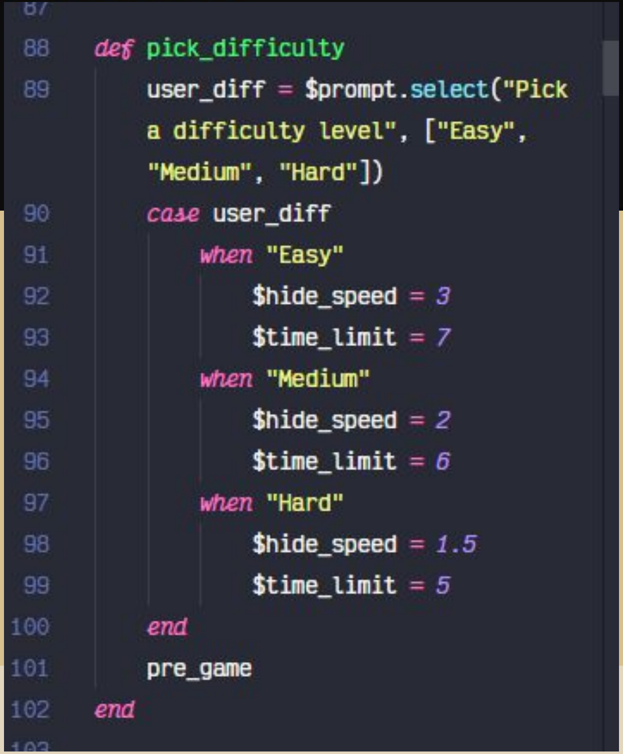
Choosing difficulty

It's good to have options, and not everyone likes too hard a challenge.

The difficulty levels here represent how much time a player has to look at a word ($hide_speed) and how long they have to enter it ($time_limit)



```
Are you happy with this character? Yes
Pick a difficulty level
  Easy
▸ Medium
  Hard
```

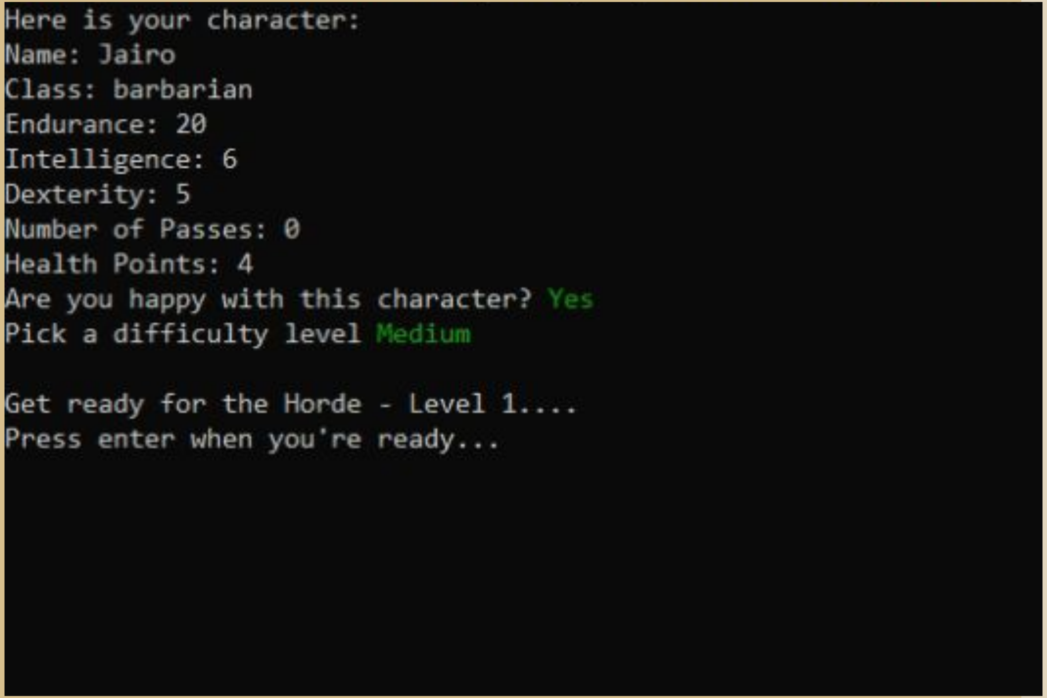


```ruby
def pick_difficulty
    user_diff = $prompt.select("Pick a difficulty level", ["Easy", "Medium", "Hard"])
    case user_diff
        when "Easy"
            $hide_speed = 3
            $time_limit = 7
        when "Medium"
            $hide_speed = 2
            $time_limit = 6
        when "Hard"
            $hide_speed = 1.5
            $time_limit = 5
    end
    pre_game
end
```

The flow of the game

There are three waves of enemies in the game, each represented by an increasing number of words.

The player is presented a word for a certain time (which decreases with each level) and then have to input that word in a given time (which also decreases)

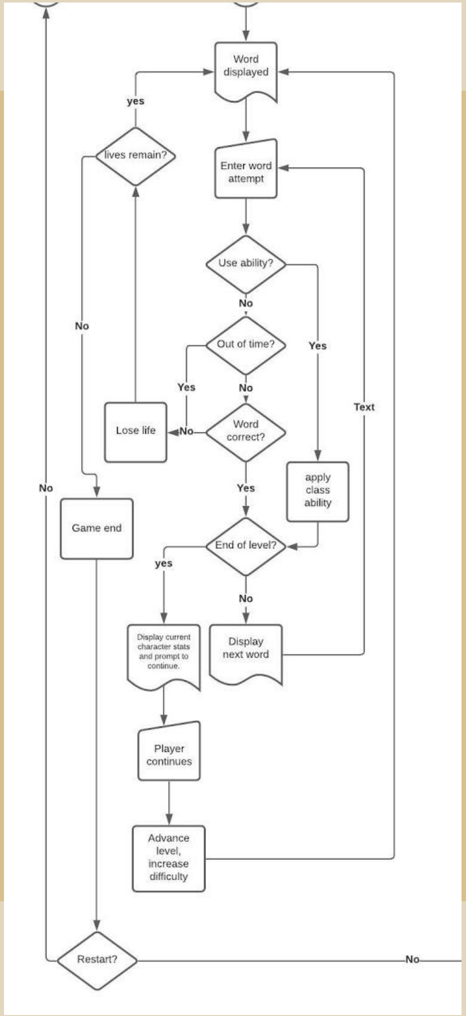


```
Here is your character:
Name: Jairo
Class: barbarian
Endurance: 20
Intelligence: 6
Dexterity: 5
Number of Passes: 0
Health Points: 4
Are you happy with this character? Yes
Pick a difficulty level Medium

Get ready for the Horde - Level 1....
Press enter when you're ready...
```

It's a bit loopy…!

The game loop is quite simple and uses a few conditional control structures to progress to success or failure…

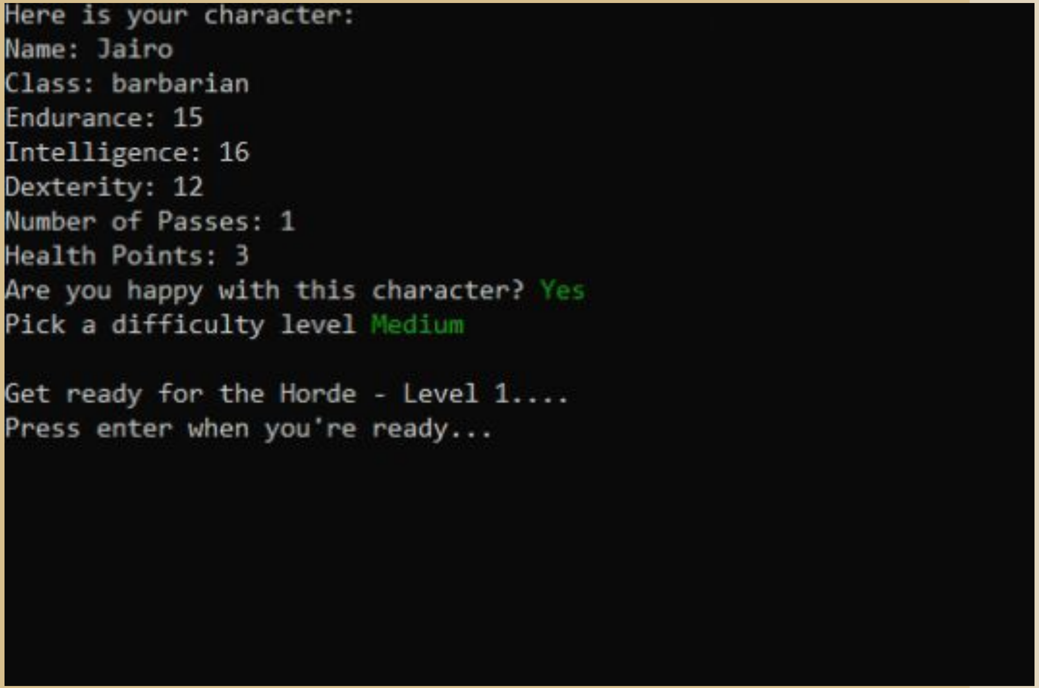
…I still haven't manage to complete the game!



Special Abilities

All characters have a "Pass" ability, which they can use a number of times dependant on their intelligence trait - this simply allows them to pass by a word without penalty.

Character abilities do other things, like the barbarian who can sacrifice health points to wipe out have a wave of enemies in one hit!

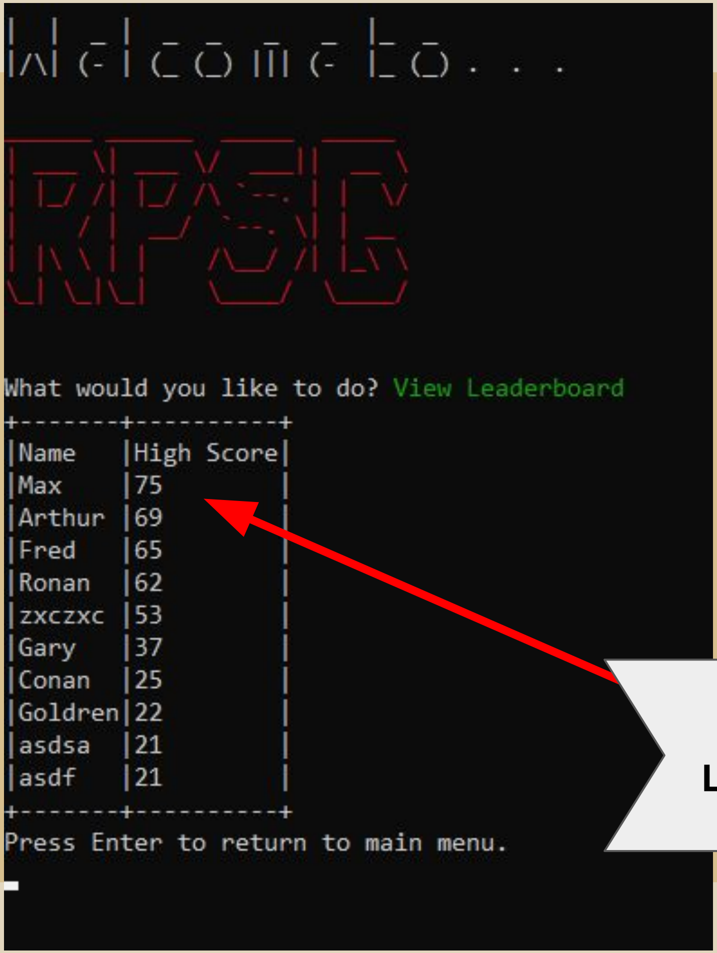


```
Here is your character:
Name: Jairo
Class: barbarian
Endurance: 15
Intelligence: 16
Dexterity: 12
Number of Passes: 1
Health Points: 3
Are you happy with this character? Yes
Pick a difficulty level Medium

Get ready for the Horde - Level 1....
Press enter when you're ready...
```

Making a record

In the “olden days”, leaderboards on arcade games were the whole reason to play.

The quicker you spell your words correctly, the higher score you’ll gain, and upon death (or winning) you will be immortalised on a scoreboard!



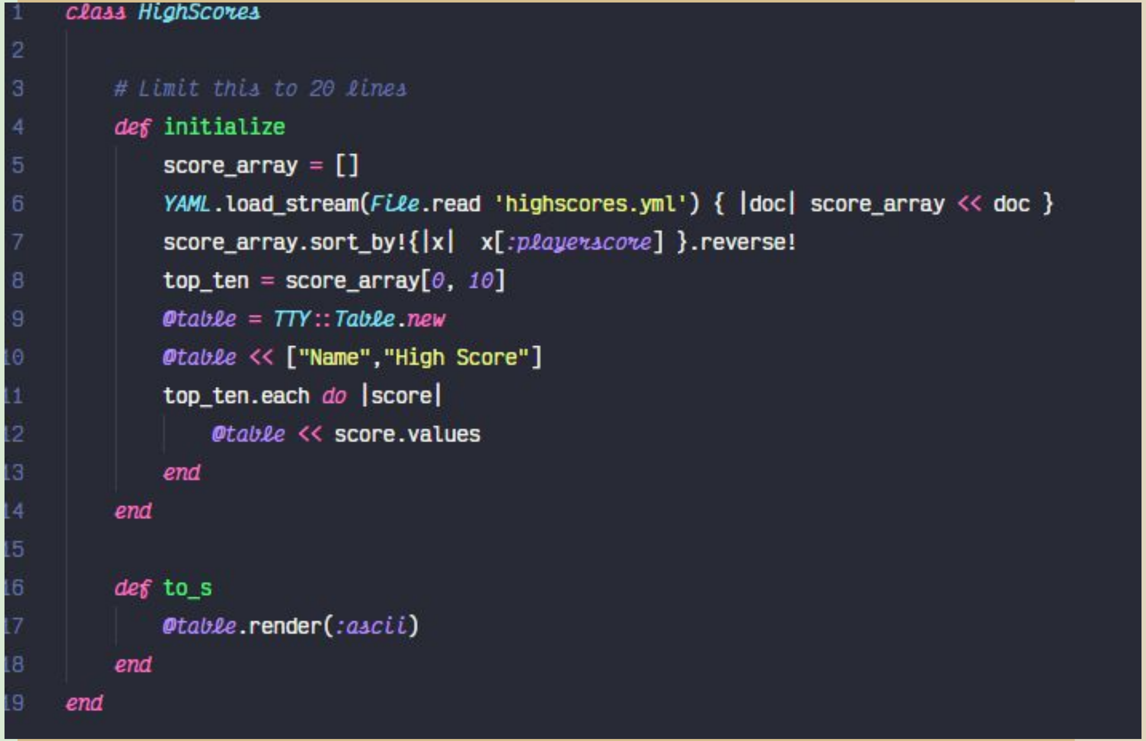
```
What would you like to do? View Leaderboard
+-------+----------+
|Name   |High Score|
|Max    |75        |
|Arthur |69        |
|Fred   |65        |
|Ronan  |62        |
|zxczxc |53        |
|Gary   |37        |
|Conan  |25        |
|Goldren|22        |
|asdsa  |21        |
|asdf   |21        |
+-------+----------+
Press Enter to return to main menu.
```

MAX IS LEADER!

Be Persistent!

Although this is only a local app, we can still have some persistent data.

High scores are saved to a YAML file and then read back in to give the top 10 best scorers in the game.



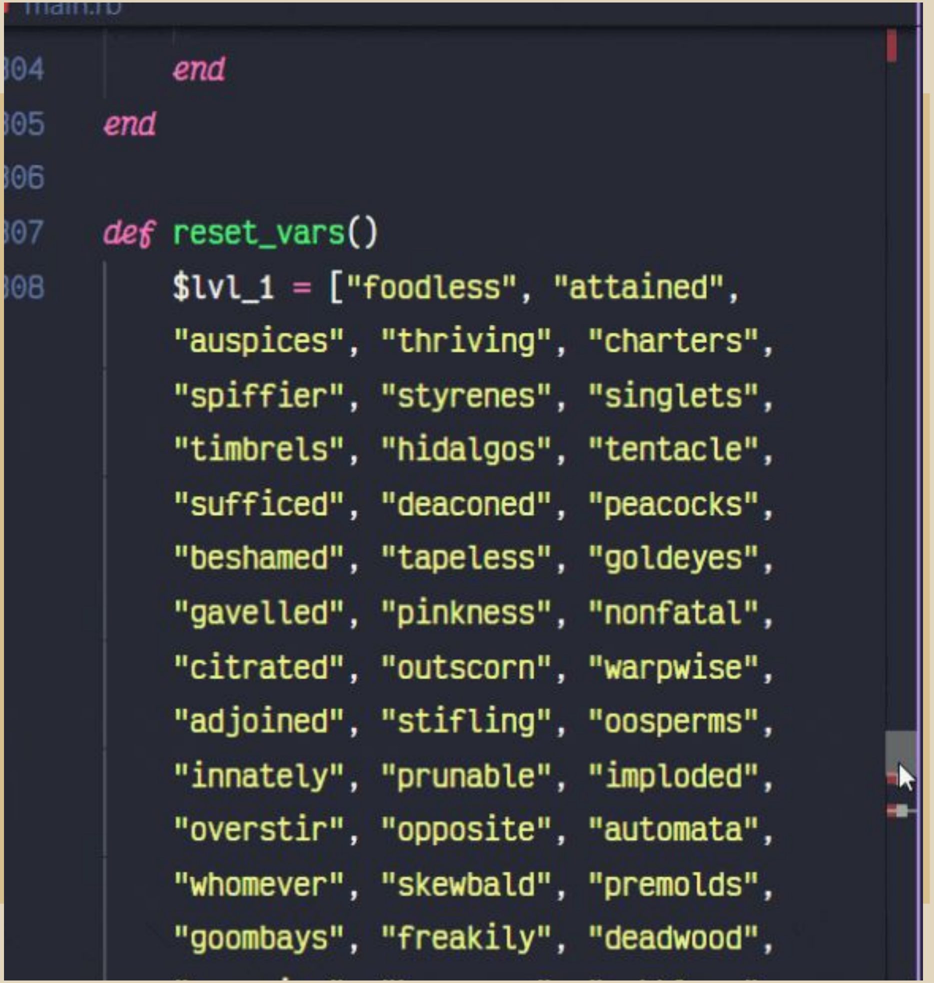
```
class HighScores

    # Limit this to 20 lines
    def initialize
        score_array = []
        YAML.load_stream(File.read 'highscores.yml') { |doc| score_array << doc }
        score_array.sort_by!{|x|  x[:playerscore] }.reverse!
        top_ten = score_array[0, 10]
        @table = TTY::Table.new
        @table << ["Name","High Score"]
        top_ten.each do |score|
            @table << score.values
        end
    end

    def to_s
        @table.render(:ascii)
    end
end
```

Just one more go...

The best games are the ones you want to play again....and again...

For a terminal app, this can be a hard task, but I have a pool of words for each level which are drawn from randomly which will give a little replayability...

Words are deleted as you play to ensure no repeats, but are all reset with wach new game.



```ruby
        end
end

def reset_vars()
    $lvl_1 = ["foodless", "attained",
    "auspices", "thriving", "charters",
    "spiffier", "styrenes", "singlets",
    "timbrels", "hidalgos", "tentacle",
    "sufficed", "deaconed", "peacocks",
    "beshamed", "tapeless", "goldeyes",
    "gavelled", "pinkness", "nonfatal",
    "citrated", "outscorn", "warpwise",
    "adjoined", "stifling", "oosperms",
    "innately", "prunable", "imploded",
    "overstir", "opposite", "automata",
    "whomever", "skewbald", "premolds",
    "goombays", "freakily", "deadwood",
```

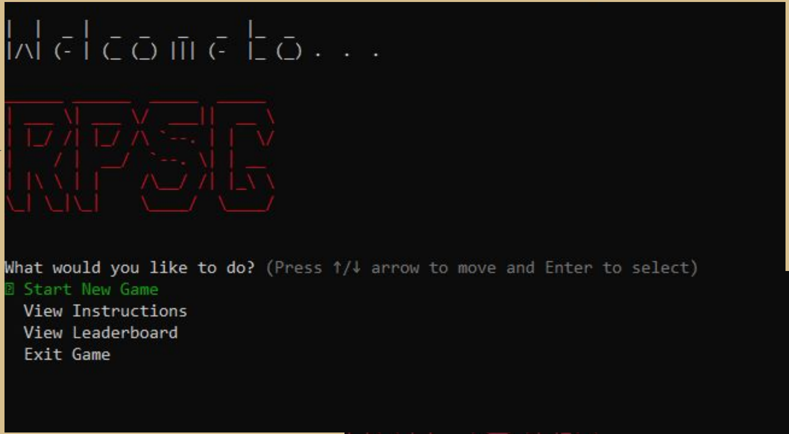
There's treasure in this...

Gems used:

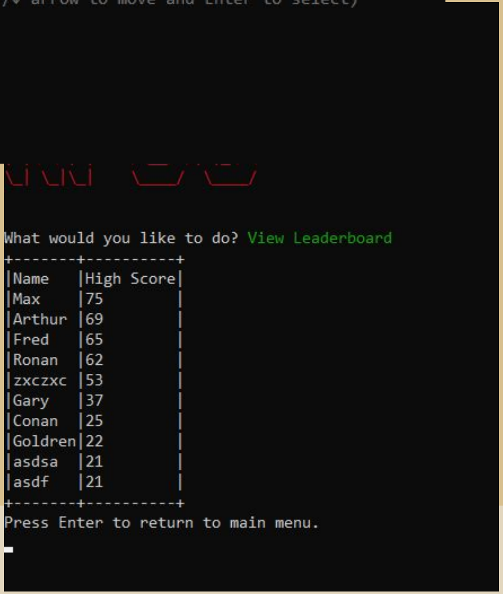
- colorize
- tty-font
- tty-prompt
- tty-table
- ...and bundler ;)

Plus some standard libraries:

- yaml
- time
- timeout



```
What would you like to do? (Press ↑/↓ arrow to move and Enter to select)
  Start New Game
  View Instructions
  View Leaderboard
  Exit Game
```



```
What would you like to do? View Leaderboard
+-------+----------+
|Name   |High Score|
|Max    |75        |
|Arthur |69        |
|Fred   |65        |
|Ronan  |62        |
|zxczxc |53        |
|Gary   |37        |
|Conan  |25        |
|Goldren|22        |
|asdsa  |21        |
|asdf   |21        |
+-------+----------+
Press Enter to return to main menu.
```

A little boy at heart

So yeah, I did want to make a game, I like playing games...I even like spelling games.

Anyone can play this, any age, any gender, even if they know nothing of RPG's.

I took this chance while I could to make a game of sorts, because it was my choice.

...but seriously

I learned a lot. And learned that I still have a lot to learn. I'll never know it all.

<u>**DEVELOPMENT PROCESS**</u>

- Wrote my plan down on paper
- Wrote some stuff on a Trello board
- Actually did refer to it at some points
- Sank into days of code
- Tried to be more prepared than in previous assignments
- Pretty happy with the result