Terminal App: Battleships

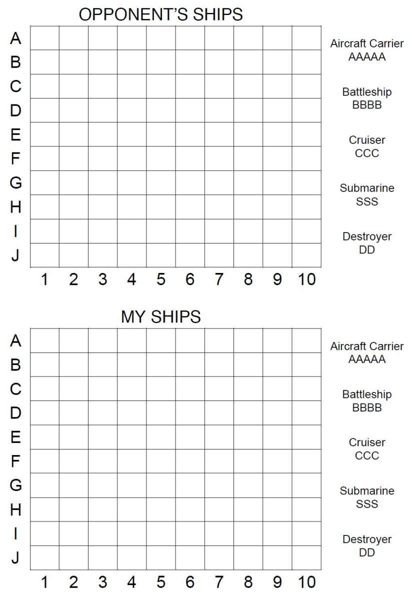
By John Thompson

Battleships is a?

A game that is inspired by the original paper game from the 1930’s.

In battleships, you take turns guessing where the computer player has placed their ships on a grid and vice-versa until someone destroys all the ships on the Grid.

Original game example



How the game is played

When the game first starts you will see your grid this is when you place your ships.
Each round you enter the grid coordinates marked on the side of the grid, first the row letter and then the column number. You will then see the computer’s grid while you are attacking and your grid while you are being attacked. A successful hit will be marked with an X and a miss is marked with a 0

The game is over once all the enemies ships have been destroyed. The score is calculated from your total successful hits.

How is it Structured?

Battleships.rb

This is the entry point for the game. It handles things like the menus and command-line arguments.

Game_elements.rb class

Game elements have attributes so that I can check their states from the game engine module, they have attributes I would need displayed on the grid or to test their states.

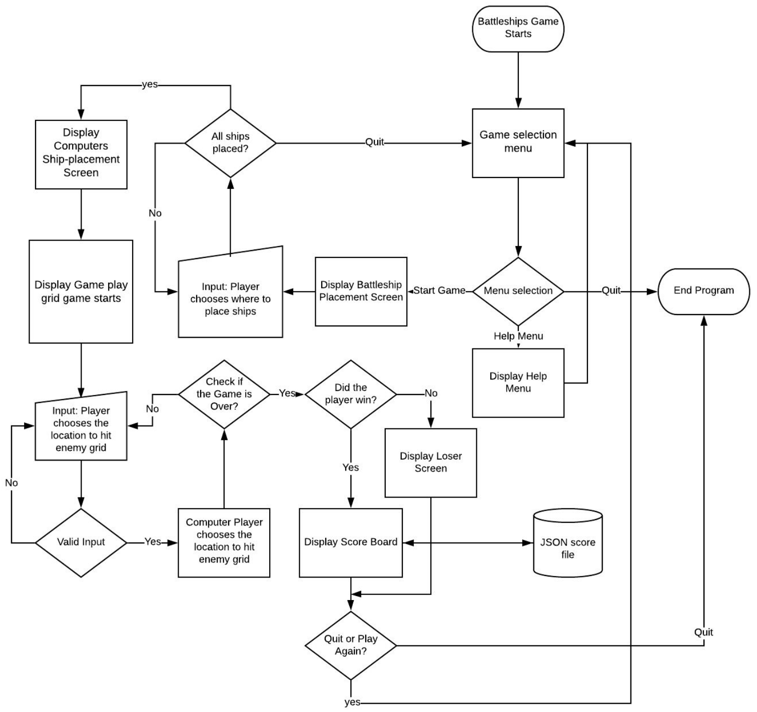
Game_engine.rb module

This module controls the creation of both player, computer grids, ships objects and gameplay logic, it basically runs the game.

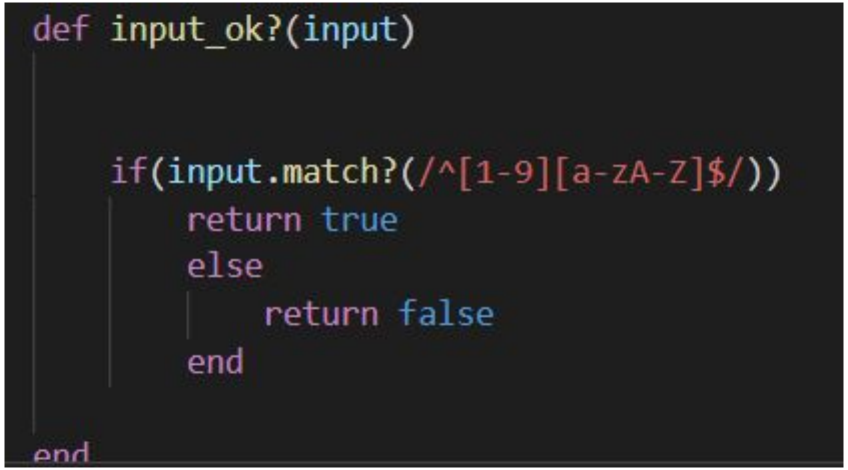
Grid.rb class

The grid is a multidimensional array and populates itself with game element objects which are place holders for grid points and labels for rows and columns of the grid, it also redraw its contents to the screen so you can see each players grid.

Code Flow Chart

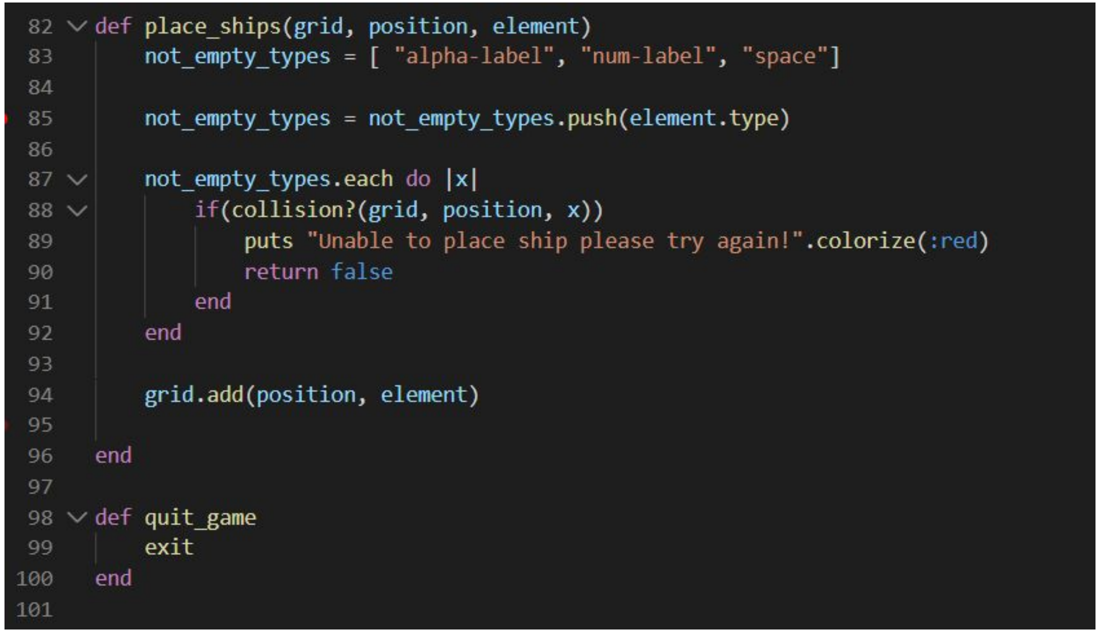

Battleships Game Starts
Game selection menu
Menu selection
Start Game
Quit
Help Menu
End Program
Display Help Menu
Display Battleship Placement Screen
Input: Player chooses where to place ships
All ships placed?
Quit
No
yes
Display Computers Ship-placement Screen
Display Game play grid game starts
Input: Player chooses the location to hit enemy grid
Valid Input
No
Yes
Computer Player chooses the location to hit enemy grid
Check if the Game is Over?
No
Yes
Did the player win?
No
Yes
Display Loser Screen
Display Score Board
JSON score file
Quit or Play Again?
Quit
yes

Code Snippets



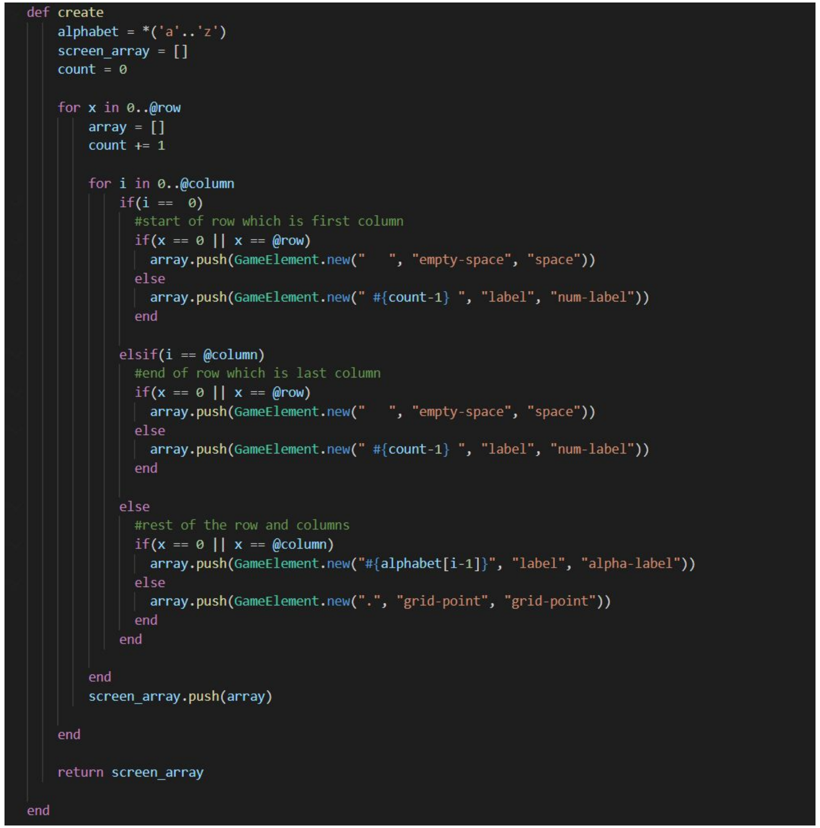
```
def input_ok?(input)


    if(input.match?(/^[1-9][a-zA-Z]$/))
        return true
        else
            return false
        end

end
```

Code Snippets



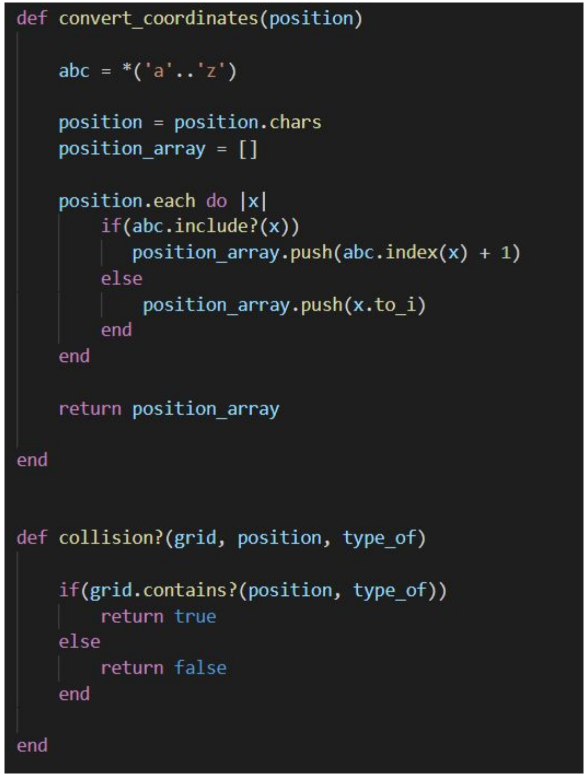
```ruby
def place_ships(grid, position, element)
    not_empty_types = [ "alpha-label", "num-label", "space"]

    not_empty_types = not_empty_types.push(element.type)

    not_empty_types.each do |x|
        if(collision?(grid, position, x))
            puts "Unable to place ship please try again!".colorize(:red)
            return false
        end
    end

    grid.add(position, element)

end

def quit_game
    exit
end

```

Code Snippets



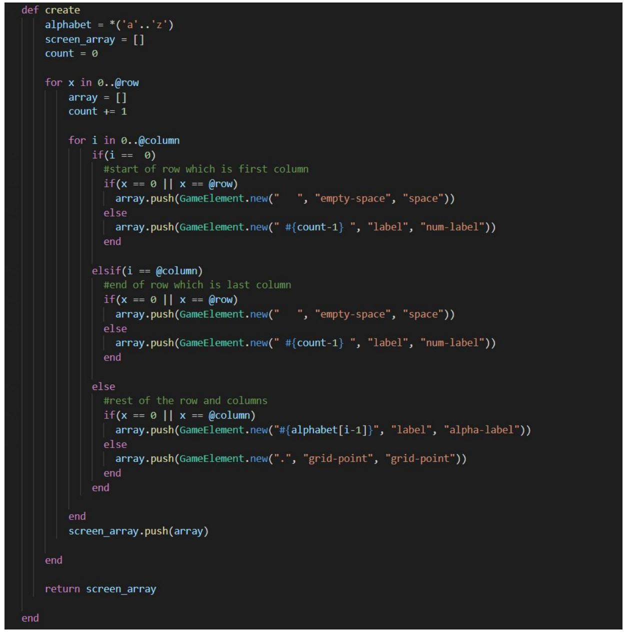
```ruby
def create
    alphabet = *('a'..'z')
    screen_array = []
    count = 0

    for x in 0..@row
        array = []
        count += 1

        for i in 0..@column
            if(i ==  0)
              #start of row which is first column
              if(x == 0 || x == @row)
                array.push(GameElement.new("   ", "empty-space", "space"))
              else
                array.push(GameElement.new(" #{count-1} ", "label", "num-label"))
              end

            elsif(i == @column)
              #end of row which is last column
              if(x == 0 || x == @row)
                array.push(GameElement.new("   ", "empty-space", "space"))
              else
                array.push(GameElement.new(" #{count-1} ", "label", "num-label"))
              end

            else
              #rest of the row and columns
              if(x == 0 || x == @column)
                array.push(GameElement.new("#{alphabet[i-1]}", "label", "alpha-label"))
              else
                array.push(GameElement.new(".", "grid-point", "grid-point"))
              end
            end

        end
        screen_array.push(array)

    end

    return screen_array

end
```

Code Snippets



```ruby
def convert_coordinates(position)

    abc = *('a'..'z')

    position = position.chars
    position_array = []

    position.each do |x|
        if(abc.include?(x))
            position_array.push(abc.index(x) + 1)
        else
            position_array.push(x.to_i)
        end
    end

    return position_array

end


def collision?(grid, position, type_of)

    if(grid.contains?(position, type_of))
        return true
    else
        return false
    end

end
```

Code Snippets - favourite Code



```ruby
def create
    alphabet = *('a'..'z')
    screen_array = []
    count = 0

    for x in 0..@row
        array = []
        count += 1

        for i in 0..@column
            if(i ==  0)
              #start of row which is first column
              if(x == 0 || x == @row)
                array.push(GameElement.new("   ", "empty-space", "space"))
              else
                array.push(GameElement.new(" #{count-1} ", "label", "num-label"))
              end

            elsif(i == @column)
              #end of row which is last column
              if(x == 0 || x == @row)
                array.push(GameElement.new("   ", "empty-space", "space"))
              else
                array.push(GameElement.new(" #{count-1} ", "label", "num-label"))
              end

            else
              #rest of the row and columns
              if(x == 0 || x == @column)
                array.push(GameElement.new("#{alphabet[i-1]}", "label", "alpha-label"))
              else
                array.push(GameElement.new(".", "grid-point", "grid-point"))
              end
            end

        end
        screen_array.push(array)

    end

    return screen_array

end
```

Time to Play!