Risk – Avatar The Last Airbender

# Purpose and goal

I was inspired to make my project related to Avatar – The Last Airbender because I really enjoy the show itself. Therefore, I came up with the idea to make a Risk game but Avatar themed. I wanted my game to be interactive and aesthetically pleasing. I really focused primarily on the user interface, choosing cool images and colors that coordinate with them. I initially wanted to have single player mode and multiplayer mode. But I ran out of time to program the AIs. Hence, only multiplayer game would work. For this project, I applied Object-Oriented-Programming to make codes less messy.

# User interface

As the most crucial focus of my project, the user interface in this game is very attractive. I used simple logos to represent players, and draw these images over the board. The buttons and states are very interactive, as they change colors when you hover over them. The dice of the actual Risk board game is located at the right corner of the game screen to show what each player rolled. The end screen is also responsive to the game, in that it takes the actual winner’s image and display it on the screen.

# Modules and functions

I used the pygame module and applied object oriented programming. Below is a list of classes and their functions.

### Class TextObjs 🡨 Class Button 🡨 Class State

TextObjs is used for things such as titles and reminders in the game. It has as a constructor that allows me to put in text content, its coordinate, font, and color.

Button extends TextObjs although the draw function is more advanced. It draws a rectangle underneath the text to make it look like a button. It also has an update function that updates the button’s color when hovered over.

State extends button and is very similar to Button except for that it is circularly shaped.

### Class StartMenu – Tutorial – Option – Transition – EndMenu

All these classes are very similar. They all have buttons and TextObjs. They facilitate between each game screen and makes the game play smoother.

StartMenu has a title and four buttons.

Tutorial has a title, a bunch of Textobjs as instructions, and 2 buttons to go back and forth.

Option has a title and 2 buttons to turn music on and off

EndMenu has a title, 2 buttons and 1 image of the winner.

All these classes don’t have Gameplay functions. They are just part of the interface.

### Class Player

Player is the class that controls the actual game play. It works together with the Board class to create the gameplay experience.

A player can claim state, attack, conquer, place troop, and lose state. To be more explicit, a player can access the states within the board and change the variables in the state with the previous functions I mentioned.

A player holds an image to represent itself. These images are stored in the image folder.

### Class Board

The board class is both an interface and the data structure of the game. A board holds a bunch of states, and these states holds values of troops. A board also holds 5 dice, 3 for attacking, and 2 for rolling. When a player attack, it calls the dice within the board class, and the board class roll dice and display them on the screen.

A board holds a board map image for the game. This image is stored in the top level.

### Class Die

The die class extends buttons (a square button) that controls the dice rolling feature of Risk. But a die has the roll function to randomly return a number between 1 and 6.