Jackson Shuminski

Project Syphon

Milestone 2

1. Jackson Shuminski and Project Syphon
2. I am the Architect.
3. I created the finite state machine in the Game1 class and and created “manager” classes that have updates and draws. The switch statement checks the peek method of a stack that holds an enum called “gameState”. This allows for back buttons that pop to the last state. The enums that are currently in are the finite state machine are the Game and Menu.
   1. The menu is managed by a class called MenuManager. The MenuManager class has two buttons. I made buttons modify the outer stack that is used in the finite state machine. They also change color slightly when hovered over with the mouse. One of the buttons in the MenuManager pushes a gameState.Game into the stack. It’s texture is that of a start button. The other button will close the game if clicked.
   2. The GameManager was where we put the game logic in. I added a pause to this manager that will switch off the updates of select objects so that everything freezes but buttons still work. It swaps between paused and playing when escape is pressed for the first frame. This holds a lot of the groups work and it has an update and draw.
   3. The Map class was a class I made to hold and manage the turrets and load a level for the turrets and base. The map decides where on a grid to place the turrets by reading a binary file (which was made by the external editor). The first two ints are the width and height to make the map grid. The tiles will change size depending on how big the map is. The next set of ints represents what each tile is. Using nested for loops the file is read and used to fill a 2d array with BasicTurret objects and one MainStructure object. Parts of the array are left null if there is no structure. This class also manages the structures.
   4. The BasicTurret class has a list of enemies passed in its update. It uses this to pick the closest viable target that it can attack. When a target is set it pivots to face its gunbarrel toward the target and starts doing its firing animation. It also damages the target. This was the most tedious part to code because it was very buggy. I ironed out almost of the bugs and they work smoothly.
   5. The MainStructure class was unique because it spans over 4 tiles. When loading it in it was given values to cover multiple spots on the screen.
   6. For the External Tool I modified the Level Editor from gdaps 1 to give int values depending on the color of the tile. For this I just used dictionaries with colors as keys and the associated int as a value.
4. The most buggy part was trying to get the turrets to draw properly. Because they have both a sprite sheet and had a rotation, the draw method would have weird bugs. The tiles would display shifted off to the side and moved around the screen in weird swinging motions as it turned. Because I was using the same code that worked for the player’s rotation, I knew that it worked. I worked on this bug for a long while to avail before I figured out that the origin is based on the source rectangle. After that I discovered another bug. The turrets were slightly off spot and rotated so that they faced in almost the right direction but not really. After a long while of trying and failing to fix this bug, I found the solution. It was being drawn half a width of a tile off in the x and y direction. Before I added rotation they were perfectly in place. Once I fixed this they began to rotate properly.
5. The architecture remained mostly unchanged. Some interfaces were made into abstract classes.