Group Members: Jacob Brandt, Austin Kilduff, Nicholas Flanders

Group Name: Portal++

CCIS GitHub Location: https://github.ccs.neu.edu/nflans/Portal-Plus-Plus

Project Description:

This project is a 2D game with a similar concept to the well-known game Portal. The player is trapped in a room and must find a way to escape. The character is equipped with a portal gun that can allow them to instantly move between two portals of different colors which can be deployed on the walls of the maze. The game will be written in C++ using OpenGL for 2D graphics, OpenInput for handling user input, and (most likely) OpenAL for in-game audio.

Describe the major features of your project:

- Character can move left, right, and jump
- Character can move through portals and teleport between them instantaneously
- Character can launch portal from it's location
- There are multiple levels that are progressively more difficult
- Score system based on how efficiently the player completes a level

Describe the advanced feature(s) of your project, and the library/SDK/API you plan to use:

We plan to use OpenGL to render the 2-dimensional graphics for this game. The control scheme for the game involves both mouse input (for launching portals) and keyboard input (for moving the character), so we plan to also use OpenInput for handling these functions. Finally, we intend to incorporate music and sounds for events like firing the portal gun, so we will also use OpenAL.

Describe the user input your program will take and how it will affect the state of the program:

Left mouse button - launch a portal to a location relative to the player and the mouse pointer.
A - move player left
W - jump
D - move player right
Q - switch to shooting blue portal.
E - switch to shooting yellow portal.
R - Close both portals.
F - pick up an object.

Esc/P - pause game.

Briefly describe plans for dynamic memory management and class inheritance structure:

Inheritance Structure:

Interface: Game Object (obeys physics like gravity and momentum)
- Abstract Class: BaseObject
  - Class: Block
    - Turret
    - Player
    - Wall/Structure
    - Portal

Interface: Projectile (doesn't obey certain physics like gravity)
- Abstract Class: BaseProjectile
  - Class: PortalProjectile
    - TurretProjectile

Class: Level

Memory Management:

As projectiles are created and destroyed, they are each represented by a new object. However, these objects are only necessary as long as the projectile is in flight, so the objects must be deleted once they are no longer necessary to prevent memory leaks. Additionally, each level that the player progresses through will have a unique set of objects. When a player leaves a level, every game object and projectile that exists in that level must be deleted in order to manage memory usage. Additionally, if the player dies while in a level, the old player object should be deleted from memory when a new player object is constructed.