Introduction

The purpose of the application was to be the basis for a simply dungeon crawling type game, with a few rooms across multiple floors, being randomised at the start of each floor, the application is a much simpler version of this style, with one pre-set floor to show the basic mechanics of this style, with pre-set rooms and layouts.

The main features of the application is a small player character with simple movement, who fires a ball straight up that is affected by gravity, making it come back down to the bottom.

The application has a simple start screen and 2 end screens, one for loss and another for victory.

Application Design

The code is designed using a basic set of game states; there is one for the start up one for the main floor developed, one for a game over and one for victory, that both return to the start, there is also a button in game to return to the start for if issues were to arise during playtime, if the game were to be expanded upon there would be a state for each individual floor with game over and victory messages specifically designed to match each one in style.

The main function starts the scene app, which then initialises the front end of the application, buy pressing the instructed button that then releases the front and begins to build the level, along with placing down the player and enemies.

All of the levels main geometry, the floor and the walls are all built in the initGround function, the walls are made first, each having their own collision built into it to prevent the player from walking through the walls and getting out of bounds, the floors are added last due to the fact that they do not interact with the player, and are just there to add polish and a “ground” to the room, to give the illusion of 3d movement on the 2d plane

The player character has a small amount of gravity to them, and the ball they fire does as well, this is to make it so the ball comes back down and can hit enemies around certain corners, even though the character can only shoot up, the enemies are more targets as they are stationary in the beginning, on later levels they would be designed to move and fire back with the same gravity affected orbs that the character fires.

The player characters movement is read in from the left joystick and is added to a linear velocity, the character would originally move by tiles, but this would made it harder to shoot while moving to curve shots., both of these are placed within the general update function, meaning that they both process at the same time as each other, rather than independent of one another.

In a full release of the game the rooms would have some level of procedural generation to them, meaning that the room’s layouts and positions in the level would be made to change every time the game is started.

User Guide

On the front end start up screen, simply press x to start, the left analogue stick controls movement of both the character and the camera, and you press square to fire, the start button returns you to the start screen of the application, and the x button is used at the game over screens to return to the start screen as well

Data Oriented Design

Some things that can be changed in the application to improve performance, one major thing that can be changed is the way the rooms are built, by instead of having a function that builds each wall and floor individually, instead have it so it is all one function called multiple times that is translated to different distances from the origin of the first rim and the start point.

Conclusions

During the course of this module I have learned of many ways to improve both accessibility and performance of my programmes, such as using game states being loaded in and out to prevent the running of too many background processes, and learned how to improve my functionality by using all methods of control availably to a user, such as the vitas touch controls. While going through my course work I have also learned the ways to divide up a program so that the work needed to run is divided among more threads, thus allowing for a faster program with better runtime and easier to identify errors