**RentRace State and Readme**

**Trello:** https://trello.com/b/lUutTPMT/rentrace

In its current state, RentRace consists of 2 Levels, a title screen and a tutorial screen.

**Animation:**

In this project, animation of sprites was done frame-by-frame using custom written sprite swapping code. This code is called SpriteAnimation.cs and in the case of the player, PlayerSprite.cs. You attach it to an object with a sprite renderer and then load in the sprites to the script in Unity. In that code, you can set the ideal framerate for the project. It would probably be good to make the rate a public variable that can be set on an instance-by-instance basis.

**Player Movement:**

Player movement is controlled by the player swiping the in direction they want to go. Node prefabs should be placed on every corner of valid areas of the map. This allows the player to make more accurate moves and can queue actions if the player swipes too early.

**Pathing:**

Pathing is achieved using A\* pathing package.

**Spawns:**

In the code, any enemy type is called a “Mugger”.  
Muggers are any Mob/Entity that pursue the player. Muggers use A\* pathing to pursue the player, and can also chose to head in a random direction every now and then to try and flank the player. Muggers are activated by a mugger spawner object which also manages their behaviour.

Coins are created at load-time by a coinspawner object which creates many copies of a single coin placed in the scene. These coins are dynamically assigned a location by randomly selecting a coordinate and then checking if the point intersects with other collision boxes. If so, then it selects another point randomly until a valid spot is found. This should be refined so that it doesn’t place coins in the centre inaccessible part of the maze.

**Win and Loss:**

Win and loss tracking is done by an object called the WinController.  
the WinController listens out for certain conditions to be met so that It can activate the notification windows. When those windows are activated, deltaTime is made equal to 0 to immobilise all entities.

**Text Displays:**

Text displays pull their information using a script called PullScore.cs.

In the case of the “-40” text that appears when a player hits an enemy, this text is activated from the player in ScoreManagerAndInteraction.cs, the text object itself then runs a script that waits for 2 seconds to pass and then turns itself off again.

**Maze:**

The maze in this project is an image that I have overlayed collision boxes onto. This approach is good for rapid development and accurate hitboxes, but not really ideal for expansion. Ideally the maze could be rebuilt using individual objects, but for the sake of rapid development, I opted for the quicker method. Nodes are placed on every corner to refine player movement. If not for the nodes, then players would make more invalid moves.

**Finish Goals:**

This game idea has evolved beyond the original design documents. The new idea is that the player takes turns as each character (Tenant, Landlord and Property Manager). Each character has a different game mode. The tenant is like traditional pacman (like the level presented), the landlord pursues the tenant and maintains a property, and the property manager is like the landlord but with more tenants and properties.

The game could benefit from a level select screen if the game is longer than 4 levels.

The Game’s art assets are not entirely completed yet, as more animations would need to be made for the land lord and property manager. The buildings should also be redesigned to look more like properties and less like non-descript buildings.

**Unused Assets:**

Buildings:

* Bank: Upon player collision with the hitbox, the player will deposit the money they are carrying into the bank. The bank generates interest over time.
* Property: When the player hits the hitbox, the player will purchase the house if they have the required points. Once purchased the property will spawn coins every now and then. Occasionally the house can be damaged and then alerts the player.
* Agency: When the player hits the hitbox, they hire an agent who picks up coins on their behalf.

Police:

* Initially the enemies would be vandals and muggers. When they were present, three zeros would spawn and function like a power pellet from pacman when call collected. The player would transform into a police car, and could chase down the muggers.

Minimap:

* There is a minimap that is currently disabled, this feature would ideally make map navigation easier, but more testing is required to see if gameplay is better by forcing the user to search the maze rather than just follow the minimap.