

User Manual

Installation Instructions

1. Download this Unity Package.
2. Open a Unity Project.
[IMPORTANT: Editor version MUST be 2022.3.11f or later. The package will NOT function otherwise.]
3. Right click in the assets folder and import the custom package.
4. Navigate to the download location, select and open it.
5. Select all files in the project, then click the import button.
6. If you don't already have it installed, you will be prompted by Unity to install the TextMeshPro package. Make sure to install this.

Quick Start

1. Begin by deleting the default Main Camera from the scene.
2. Navigate to the *Resources > Prefabs* folder. Select the GamePlane prefab, and drop it into the scene.
3. To ensure that the enemies behave correctly, edit the coordinates of the GamePlane prefab in the inspector under *Transform*, so that X, Y, and Z, are all 0.
4. From the *Prefabs* folder, drag in the Player prefab. IMPORTANT: Make sure there is only one Player in the scene.
 - a. The player can be controlled with the W, A, S, D or the arrow keys.
 - b. To ensure correct behavior, adjust the transform of the player so that the Y axis is 0.5.
5. From the *Prefabs* folder, navigate to the *Enemies* folder. Contained are prefabs of all enemies.
 - a. The BaseEnemy will follow the player constantly, and deal damage on contact.
 - b. The DashEnemy will periodically dash toward the player's last location, and deal damage on contact.
 - c. The ShootEnemy will stay a fixed distance from the player, and shoot bullets toward them, which deal damage on contact.
 - i. To ensure correct behavior, adjust the transform of all enemies once placed, so that their Y axis are 0.5.
6. Drag and drop these enemies into the scene as you desire. Once they are in the scene, their settings can be configured through the inspector.
7. Press Play!

Configuration

- All enemies have the following configurable values:
 - Move Speed: A float which dictates how quickly the enemy moves.
 - Rotate Speed: A float which dictates how quickly the enemy rotates to face the player.
 - Respawn Time: An integer that dictates, in seconds, how long it will take for the enemy to respawn after it has died.
 - Respawn: A boolean which dictates whether or not the enemy will respawn.
- The Base Enemy / Slow Enemy has the following additional configurable values:
 - Damage To Deal: Will dictate how much health the player loses on contact with the enemy.
- The Dash Enemy has the following additional configurable values:
 - Time Between Dashes: An integer which dictates, in seconds, the time between spent resting after dashing at the player.
 - Dash Multiplier: A float which controls the speed that the enemy dashes toward the player at, as multiplied by the Move Speed.
 - Damage To Deal: Will dictate how much health the player loses on contact with the enemy.
- The Shooter Enemy has the following additional configurable values:
 - Enemy Distance: A float which dictates how far the enemy will stay from the player.
 - Enemy Shot Timer: An integer which dictates (in seconds) how long to wait between each shot being fired.
 - Bullet Speed: A float which dictates how fast the bullets will travel in the world.
 - Bullet Damage: An integer which dictates how much damage the player will take on contact with a bullet.

Application Programming Interface (API)

- Inside Player.cs, the public function TakeDamage(int dmg); is used by the enemy scripts to adjust the player's health, without directly accessing the data.
- Inside Player.cs the public function GetHealth(); is used by the UI to display the player's remaining health as an integer on the screen.
- Inside all enemy classes is the public function SetEnemyType();. This is public as a result of it being necessitated by the IEnemy interface. It stores a reference to its own enemy prefab as a variable to be accessed by the respawn manager.
- Inside RespawnManager.cs is a public function called DoRespawn(); which is accessed by enemy classes when they are destroyed. It takes in:

- An integer set by each enemy which references their location in the array of enemy prefabs.
- An integer retrieved from the enemy class which dictates the time it will take them to respawn, in seconds.
- The respawn manager will select a random place on the GamePlane, and instantiate the enemy when the respawn timer has expired.
- When the object is instantiated, it is added to a list, so the boolean value which dictates whether or not the enemy will respawn, is set to true. This is necessary as the objects are instantiated at their default values, where the boolean for respawn is false.