**PUTT IN THE HOLE**

This game is basically Minigolf, but the gameplay elements were inspired from the game called Balatro.

Quick description about the game is your main goal is to putt in the hole as the name suggests but you should also score points to finish the level. In order to boost your score you need to transform yourself into a ball that gives you special boosters for points to finish the level.

The catch is the level modifier, each level you start, you get a random modifier that stops you from scoring. In order to get over the modifier you need to choose the right ball for the right modifier to finish the level.

**Credits**

All the sprites for the game are done in photoshop by me.

Music : <https://assetstore.unity.com/packages/audio/sound-fx/sci-fi-ui-sound-effects-hove-audio-263876#description>

-> HoveAudio

**Controls and build details**

Press and hold SPACEBAR to Charge the shot for the ball and release the key to shoot.

Use the mouse for direction, the ball moves where the mouse pointer is.

Build the files in Debug mode ONLY and Hit play. Game works in just debug mode for now.

Main Menu for the game is pretty straightforward with all the details given.

**Game Features**

* **Golf Shooting Mechanincs:**

Instead of the regular press and drag shooting I’ve decided to give a charging mechanic for the player to shoot the ball in the desired force. Pretty cliché but works.

* **Level Modifier System:**

My level modifier system is really modular as you can add n number of modifiers you want, and the system decides which one to give in one level. SPOILER My favorite is the Fog Of War Modifier where you can barely see your path to the hole. Good Luck with that.

* **Point and result calculation like Balatro**

For The point calculation I wanted to do like Balatro with a score multiplier. They basically use jokers to multiply score, but I use the golf ball itself. (Inspiration).

* **Main Menu and Instructions screen:**

Like Every other game there is a Main Menu and instructions screen for you to look at the controls and play the game.

**Notable Coding Features**

* **ECS and Event System:**

My ECS workflow is inspired from a website which I attached below. Just like all other ECS it has Entity Manager and Component Container. The notable part is the componentIterator where I create a iterator class to iterate through entities which has specific components. Save some Load in the memory. The event system handles all type of events and open to update more in the future. Ref added below.

* **Physics Collision Normal Calculation:**

My Collision calculation does not stop with AABB collision. We can add more collision calculations in the engine and my ECS workflow will allow me to add future upgrades without any interdependent or spaghetti code. Ref added below

* **System Factory and Independency:**

My System classes work independently. Meaning if you remove one system from the list the game still works as none of system needs to know what other system is doing. You can comment on any system in the code and the game will still run. It won’t run if you comment on the gameplay system obviously.

* **Future Upgrades and Features:**

My Engine is open to future upgrades which can handle object pooling and raycasting. Infact there are code for those in the engine but there wasn’t any need for me to use atleast for now

**References:**

[**https://www.david-colson.com/2020/02/09/making-a-simple-ecs.html**](https://www.david-colson.com/2020/02/09/making-a-simple-ecs.html)

[**https://youtu.be/NsGGuapf-Ms?si=13RRkalFB1BefLBe**](https://youtu.be/NsGGuapf-Ms?si=13RRkalFB1BefLBe)

[**https://youtu.be/Z-CILn2w9K0?si=TKCpRHhpgA1LZEdQ**](https://youtu.be/Z-CILn2w9K0?si=TKCpRHhpgA1LZEdQ)

[**https://youtu.be/u-J3SbW3cqE?si=3-cJM\_nPZ6E1Nf-1**](https://youtu.be/u-J3SbW3cqE?si=3-cJM_nPZ6E1Nf-1)