**PROYECTO FINAL 3**

**JOAN ALEJANDRO SANDOVAL ESCOBAR**

**LUISA FERNANDA RINCON PEREZ**

**PONTIFICIA UNIVERSIDAD JAVERIANA CALI**

**2020**

**MANUAL TECNICO**

Todos los códigos utilizados se colocan en los gameObjects que unity crea, para así darle la funcionalidad a los objectos, darle movimiento, colisión, etc. Estos códigos se encuentrar en la carpeta Assets.

Texto

Descripción generada automáticamente

Esta parte del código llamada “player.cs” es la encargada de iniciar la vida, la velocidad, la funcionalidad de movilidad del jugador, también a la hora de que el jugadro colisione con algo, que se encuentre “is trigger” inicia la batalla

Interfaz de usuario gráfica

Descripción generada automáticamente

Ya dentro del inspector, que es algo que nos ofrece unity, esta herramienta te ayuda a colocarle los valores a las variables colocadas en código, y como se ve tiene dos scripts que son “player.cs” y “Inventario.cs”

Texto

Descripción generada automáticamenteTexto

Descripción generada automáticamente

Ya dentro del script de inventario esta toda la lógica para así tomar el objeto y guardarlo dentro de un array para así mostrarlo dentro del inventario, el inventario se abre con la tecla “i”, mostrando todos los objetos que haya recogido, su funcionamiento es, primero mira si el objeto con “is trigger” tiene el tag de “Item” para así guardarlo dentro del inventario, mira si el espacio esta ocupado y si no está ocupado lo termina guardando.

Texto

Descripción generada automáticamente

El script de “battleState.cs” es el encargado de que cuando se detecta el “is trigger” con tag “Enemy” lo que hace es mostrar un canvas que es una funcionalidad de unity para mostrar el UI, con esto se mostraria la batalla que al darle al botón de attack haría la pelea automáticamente bajándole la vida al monstro y a el jugador, si gana se oculta el canvas de la pelea y quedaría el jugador. El problema de esto es que solo toma los datos del primer combate, del primer enemigo y al entrar a otro combate no hace nada el presionar el botón debido que no tiene ningún dato al cual restar vida quedándose siempre en el estado de batalla.

Texto

Descripción generada automáticamente

El script de “Item.cs” es el encargado de tomar los datos de los ítems, y colocándole esos datos a sus hijos.

Interfaz de usuario gráfica, Texto, Aplicación, Chat o mensaje de texto, Sitio web

Descripción generada automáticamenteInterfaz de usuario gráfica, Escala de tiempo

Descripción generada automáticamente con confianza media

Tiene una función abstracta la cual las tienen sus clases hijas que son “Espada.cs” y “healthPotion.cs” para crear un override de la función y dar así el dato que se solicite

Texto

Descripción generada automáticamente

El script “enemy.cs” es el encargado de tomar los datos de vida, velocidad, etc. También tiene un código el cual le da un movimiento a este enemigo cuando un personaje con tag “Player”, se mueva hacia el para así iniciar la pelea, esta parte del código funciona mal ya que el enemigo se mueve hacia el pero se queda estático en su lugar inicial.

Como ultimo están los pequeños scripts que están para complementar los scripts más grandes, o están dentro de cámaras o funcionalidades de botones.