Detalles de código

Script de guardado.

Este es un script en C# para Unity que he creado para controlar la carga y el guardado de datos del juego, específicamente la posición del jugador. Además, he implementado una forma sencilla de cargar o guardar manualmente estos datos utilizando las teclas "C" y "G".

En la función Awake, el script busca al jugador utilizando su etiqueta "Player" y llama a la función CargarDatos para cargar los datos guardados previamente.

La función CargarDatos se encarga de comprobar si existe alguna clave de guardado previo en PlayerPrefs, y si es así, extrae el contenido de la posición del jugador en forma de un string. Luego, se convierte este string en un Vector3 y se actualiza la posición del objeto jugador con los datos cargados.

La función GuardarDatos se llama al presionar la tecla "G" y se encarga de guardar la posición más reciente del objeto jugador en PlayerPrefs.

Por último, en la función Update compruebo si se han presionado las teclas "C" o "G" y llamo a la función correspondiente.

En resumen, he creado este script para proporcionar una manera sencilla de cargar y guardar manualmente los datos de la posición del jugador utilizando PlayerPrefs y las teclas "C" y "G".

En el script "ControladorDatosJuego", se utilizan las teclas "G" y "C" para cargar y guardar manualmente los datos de la posición del jugador.

La tecla "G" llama a la función GuardarDatos, que guarda la posición actual del jugador en PlayerPrefs con la clave especificada en la variable "claveDeGuardado".

La tecla "C" llama a la función CargarDatos, que carga la posición del jugador desde PlayerPrefs utilizando la misma clave de guardado.

En resumen, la tecla "G" se utiliza para guardar los datos de la posición actual del jugador, mientras que la tecla "C" se utiliza para cargar los datos guardados previamente

EJ: CODIGO DE GUARDADO

```
/* este codigo controla la carga y el guardado de datos del juego, la posición del
jugador,
 también proporciona una forma de cargar o guardar manualmente estos datos
utilizando las teclas "C" y "G"*/

public class ControladorDatosJuego : MonoBehaviour
{
    public GameObject jugador;
    public string claveDeGuardado = "PosicionJugador";


    /*aqui buscaremos a nuestro jugador para cargar los datos que tiene*/

    private void Awake()
    {
        jugador = GameObject.FindGameObjectWithTag("Player");

        CargarDatos();
    }

    /*como su nombre lo dice cargaremos los datos que se recojan de nuestro
jugador*/
    private void CargarDatos()
    {
        if (PlayerPrefs.HasKey(claveDeGuardado))
        {
            string contenido = PlayerPrefs.GetString(claveDeGuardado);
            Vector3 posicion = Vector3.zero;
            string[] valores = contenido.Split(',');
            if (valores.Length == 3)
            {
                posicion.x = float.Parse(valores[0]);
                posicion.y = float.Parse(valores[1]);
                posicion.z = float.Parse(valores[2]);
            }
            else
            {
                Debug.LogError("Formato de datos incorrecto: " + contenido);
            }

            Debug.Log("Posicion Jugador :" + posicion);


             /*tendremos la posicion del objeto con los datos guardados*/

            jugador.transform.position = posicion;
        }
        else
        {
            Debug.Log("No se encontraron datos de guardado.");
        }
    }

    /*como el nombre lo indica se guardaran los datos de la posicion mas reciente de
nuestro objeto*/
```

```
    private void GuardarDatos()
    {
        Vector3 posicion = jugador.transform.position;
        string contenido = posicion.x + "," + posicion.y + "," + posicion.z;
        PlayerPrefs.SetString(claveDeGuardado, contenido);
        Debug.Log("Datos guardados.");
    }

    /*aqui comprobaremos eluso de las teclas y si se han usado,
     explica la funcion de cada una para guardar y cargar informacion*/
    private void Update()
    {
        if (Input.GetKeyDown(KeyCode.C))
        {
            CargarDatos();
        }
        if (Input.GetKeyDown(KeyCode.G))
        {
            GuardarDatos();
        }
    }
}
```

Aquí está la explicación de cada uno de los métodos utilizados en el siguiente código:

- Awake(): Este método se llama cuando el objeto se activa y se utiliza para inicializar cualquier valor o componente que necesite el objeto. En este caso, el método busca al jugador en la escena utilizando su etiqueta "Player" y luego llama a la función CargarDatos() para cargar los datos de guardado previamente almacenados.
- CargarDatos(): Este método se encarga de comprobar si existe alguna clave de guardado previo en PlayerPrefs, y si es así, extrae el contenido de la posición del jugador en forma de un string. Luego, convierte este string en un Vector3 y actualiza la posición del objeto jugador con los datos cargados.
- GuardarDatos(): Este método se llama al presionar la tecla "G" y se encarga de guardar la posición más reciente del objeto jugador en PlayerPrefs. Primero, recoge la posición actual del jugador y la guarda como un string en PlayerPrefs utilizando la clave proporcionada.
- Update(): Este método se llama en cada cuadro de la ejecución y se utiliza para comprobar si se han presionado las teclas "C" o "G" y llamar a la función correspondiente. Si el usuario presiona la tecla "C", la función CargarDatos() se llama para cargar los datos guardados. Si se presiona la tecla "G", la función GuardarDatos() se llama para guardar la posición actual del objeto jugador en PlayerPrefs.

¿Que son los player prefs?

PlayerPrefs es una clase en Unity que se utiliza para guardar datos persistentes entre sesiones de juego. Se pueden guardar y cargar valores de tipos de datos primitivos como enteros, flotantes, booleanos y cadenas. Estos datos se almacenan en la memoria del dispositivo en el que se ejecuta el juego y se pueden acceder en cualquier momento.

En el script que se me proporcionó, se utiliza PlayerPrefs para guardar y cargar la posición del jugador en el juego. Cuando se presiona la tecla "G", la posición actual del jugador se guarda en PlayerPrefs. Luego, cuando se inicia el juego de nuevo o cuando se presiona la tecla "C", la posición previamente guardada se carga de PlayerPrefs y se actualiza la posición del jugador en el juego.

El uso de PlayerPrefs en este script es una forma sencilla de guardar y cargar datos persistentes sin necesidad de utilizar bases de datos externas o archivos de texto.

Movimiento.

El script utiliza un Rigidbody2D para aplicar fuerzas y mover el objeto en el espacio 2D.

El objeto se mueve utilizando las teclas de flechas y la velocidad se puede ajustar a través de la variable "velocidad". El script también incluye la capacidad de aplicar una fuerza de derrape y una gravedad fija. La función Start obtiene la referencia del componente Rigidbody2D y la función FixedUpdate se llama en cada fotograma para aplicar las fuerzas y actualizar la posición del objeto.

El código no contiene ninguna función específica de teclas. Sin embargo, el código utiliza los valores de entrada de los ejes horizontal y vertical para controlar el movimiento del objeto a través de las flechas del teclado o del joystick. Es decir, cuando se presionan las teclas de flechas o se mueve el joystick en una dirección, el objeto se mueve en esa dirección. Además, el código utiliza el valor de entrada de las teclas para ajustar la velocidad del movimiento del objeto.

EJ: código de movimiento

```
public class movimiento : MonoBehaviour
{
    public float velocidad = 5f;

    private Rigidbody2D rb;

    public float fuerzaDerrape = 0.5f;

    public float gravedadFija = -9.81f;


    private void Start()
    {
        rb = GetComponent<Rigidbody2D>();
    }

    private void FixedUpdate()
    {
        rb.AddForce(Vector2.up * gravedadFija);

        float derrape = rb.velocity.magnitude * fuerzaDerrape;
        rb.AddForce(-rb.velocity.normalized * derrape);
```

```
        float movimientoHorizontal = Input.GetAxis("Horizontal");
        float movimientoVertical = Input.GetAxis("Vertical");

        Vector2 movimiento = new Vector2(movimientoHorizontal, movimientoVertical);

        rb.MovePosition(rb.position + movimiento * velocidad * Time.fixedDeltaTime);
    }

}
```

CONFIDENCIAL