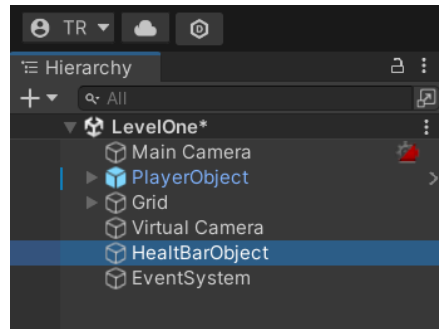


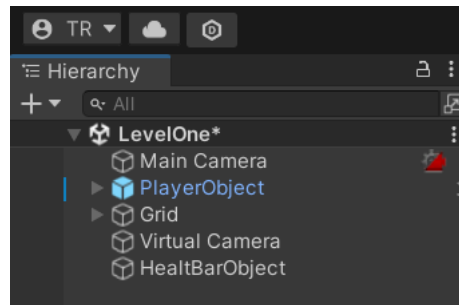
Construyendo Health Bar

Haga clic con el botón derecho en cualquier lugar de la vista [Jerarquía y seleccione UI ► Canvas](#).

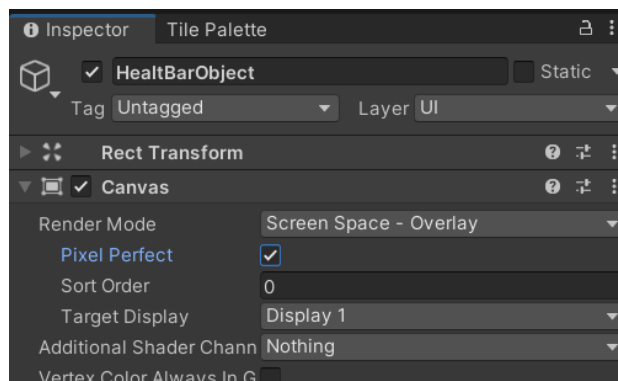
Esta crea dos objetos automáticamente: un [Canvas](#) y un [EventSystem](#). Renombrar el objeto Canvas, “**HealthBarObject**”.



El [EventSystem](#) es una forma para que el usuario interactúe directamente con objetos utilizando el ratón u otros dispositivos de entrada. No lo necesitamos en este momento, así borrelo.

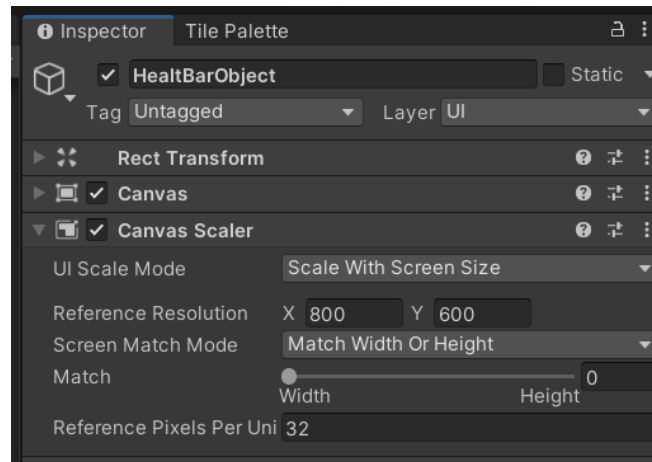


Seleccione [HealthBarObject](#) y busque el componente [Canvas en la ventana Inspector](#). Establecer el modo de renderizado en [Screen Space - Overlay](#) garantiza que Unity renderiza los elementos de la interfaz de usuario en la parte superior de la escena

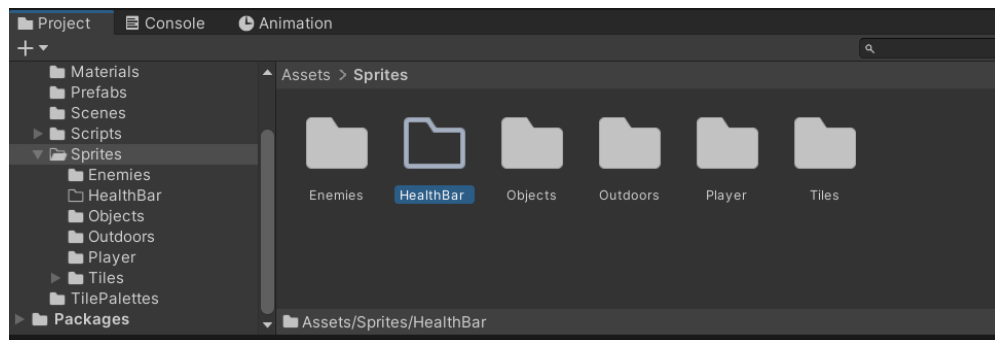


Gerardo Tonathiu Rojas Torres
Alexander Alonso Rodríguez

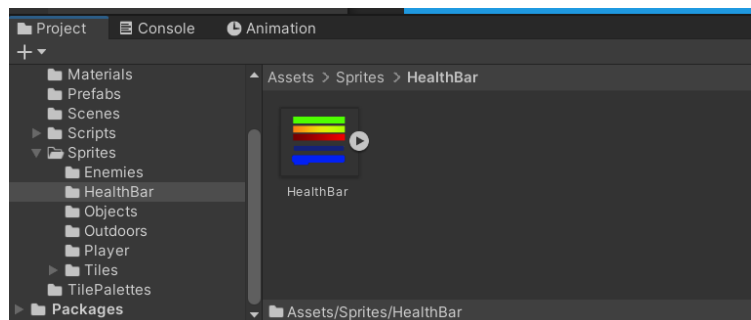
Seleccione **HealthBarObject** y busque el componente **Canvas Scaler**. Establezca **UI Scale Mode** en: **Scale With Screen Size** e igualmente establezca **Reference Pixels Per Unit** en **32**.



Es hora de importar los sprites que usaremos para la barra de salud. Crear una nueva subcarpeta en la carpeta **Sprites** llamada **"HealthBar"**.

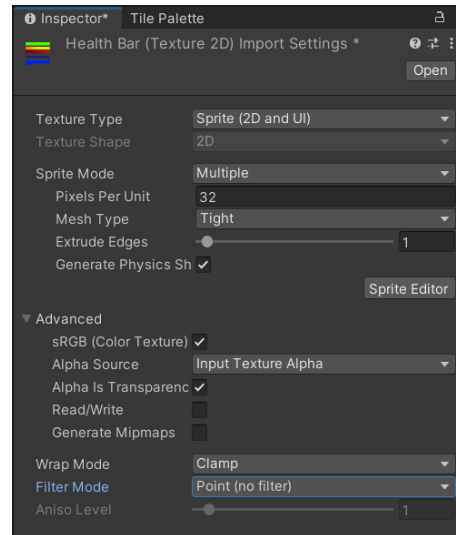


Pondremos todos nuestros Sprites relacionados con la barra de salud en esta carpeta. Ahora arrastre la hoja de sprites llamada, **"HealthBar.png"** en la carpeta que acabamos de crear.

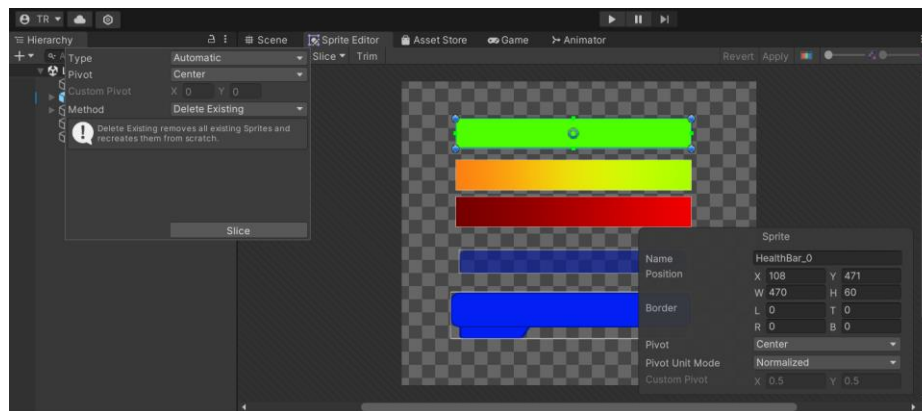


Gerardo Tonathiu Rojas Torres
Alexander Alonso Rodríguez

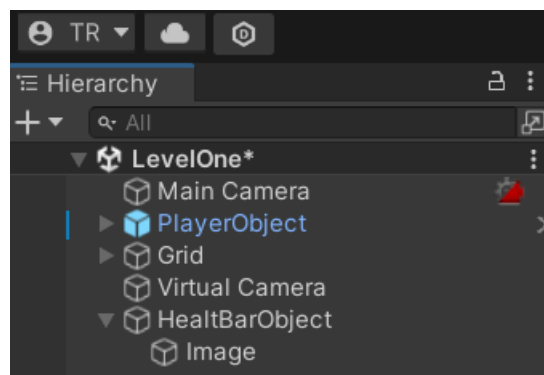
Seleccione la hoja de sprites **HealthBar** y use la siguiente configuración de importación en el Inspector:



En el menú **Slice**, asegúrese que el "Tipo" esté configurado en: **Automatic**. Vamos a permitir que Unity Editor detecte los límites de estos sprites. Presione **Apply** para cortar los sprites y luego cierre el **Editor Sprites**.

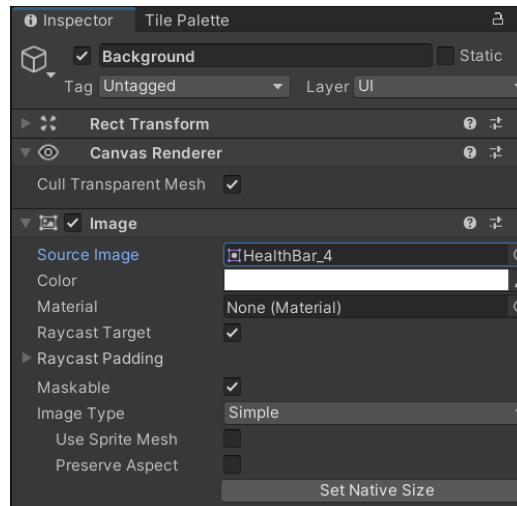


A continuación, agregaremos un objeto imagen, que es un elemento de la interfaz de usuario, al **HealthBarObject**. Seleccione **HealthBarObject**, haga clic con el botón derecho sobre el objeto y seleccione la opción **UI > Image** para crearla.

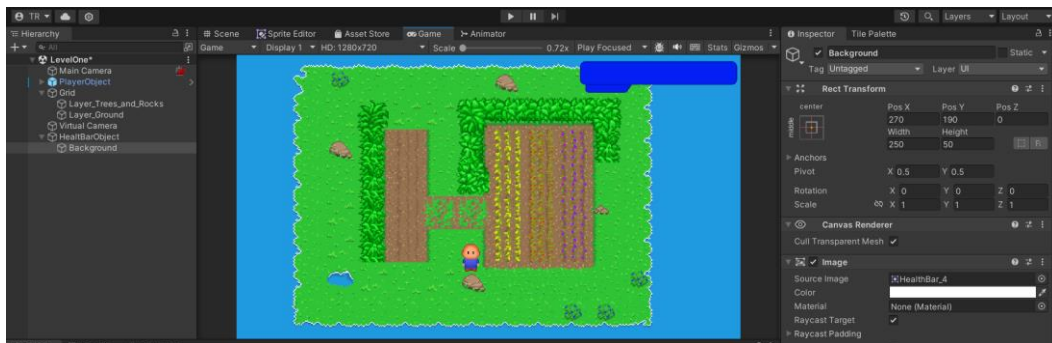


Gerardo Tonathiu Rojas Torres
Alexander Alonso Rodríguez

Este objeto de imagen actuará como imagen de fondo para nuestra barra de salud.
Cambie el nombre del objeto, "**Background**". Haga clic en el punto junto a la propiedad **Source Image** y seleccione la imagen titulada "**HealthBar_4**"

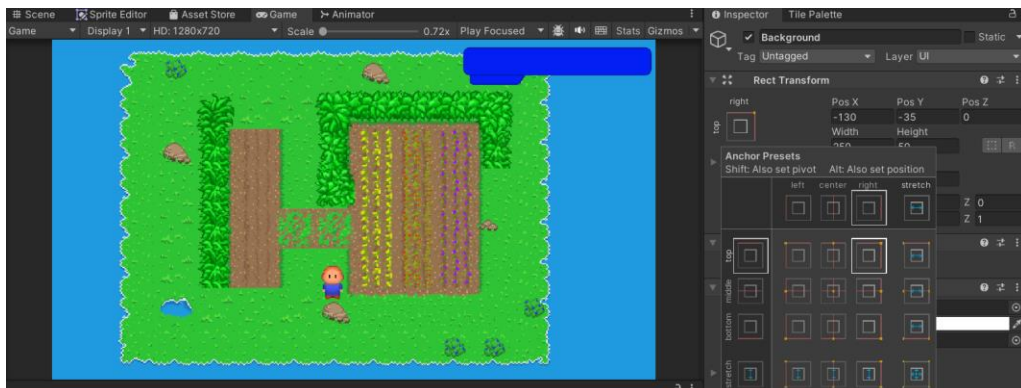


Con el objeto **Background** seleccionado, cambie la transformación de **rect Width: 250** y **Height** hasta: **50**.



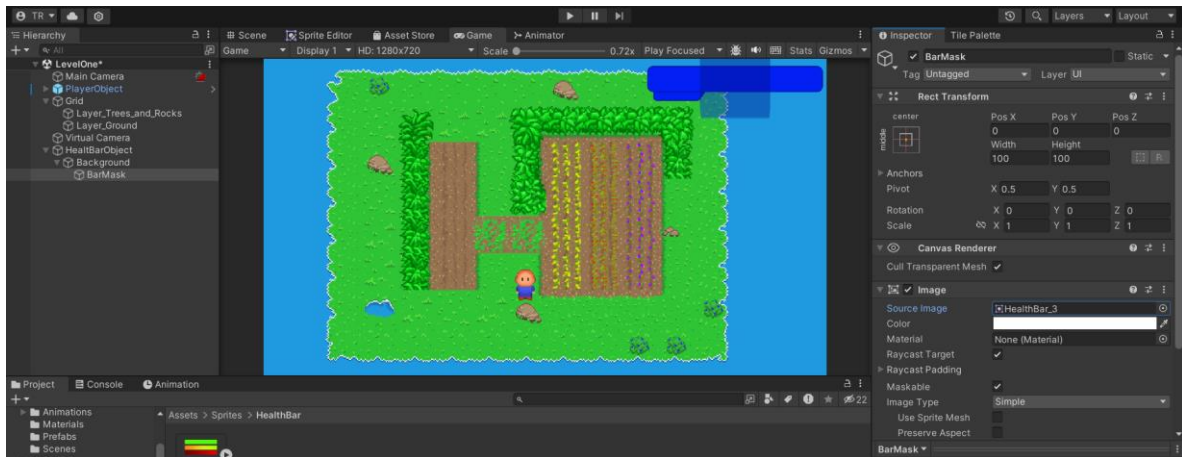
Seleccione el objeto **Background**. En el componente **Rect Transform** de la ventana Inspector, presione el icono de **Anchor Presets** resaltado

Queremos anclar la barra de salud en relación con la esquina superior derecha de la pantalla en todo momento. Seleccione la configuración de **Anchor Preset** de la columna titulada, "**Right**" y la fila titulada "**Top**".

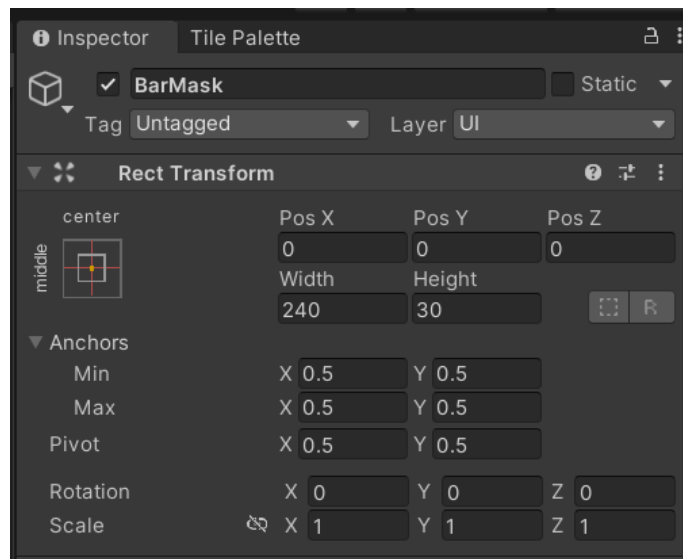


Máscaras de imagen de interfaz de usuario

Haga clic con el botón derecho sobre el objeto **Background** y cree otro objeto imagen, se creará como un objeto "hijo". El objeto **Child Image** actuará como una máscara. Esta máscara funciona un poco diferente a una máscara que podrías usar en Día de Muertos. Seleccione el objeto Imagen y cámbiale el nombre, "**BarMask**". Establece la fuente de la Imagen a: **HealthBar_3**.

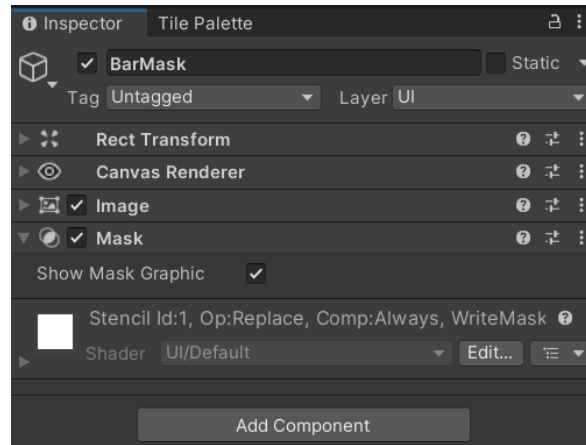


Los puntos de ancla de **BarMask** están centrados por defecto con respecto al objeto **Background**. Con el objeto **BarMask** seleccionado, cambie el tamaño de **Rect Transform** a **Width: 240** y **Height: 30**. Queremos hacer que la **BarMask** sea un poco más pequeña que la dimensiones de la barra de salud para mostrar un margen alrededor del medidor de salud real.

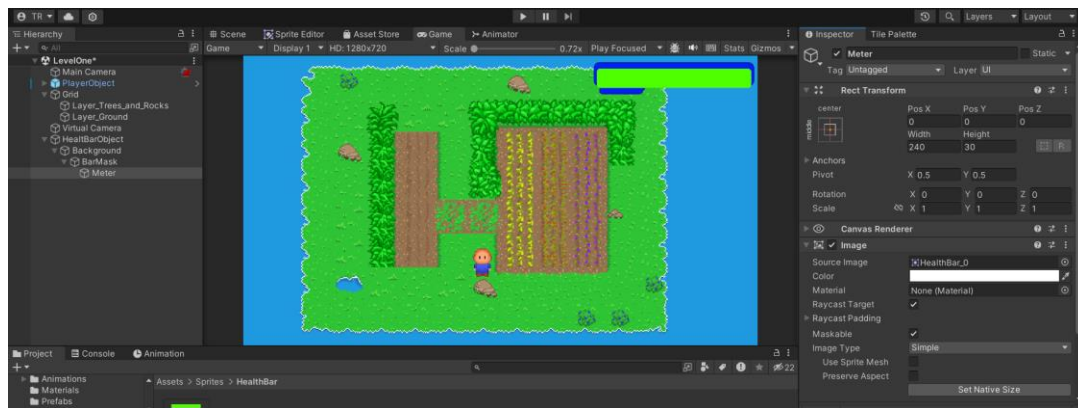


Gerardo Tonathiu Rojas Torres
Alexander Alonso Rodríguez

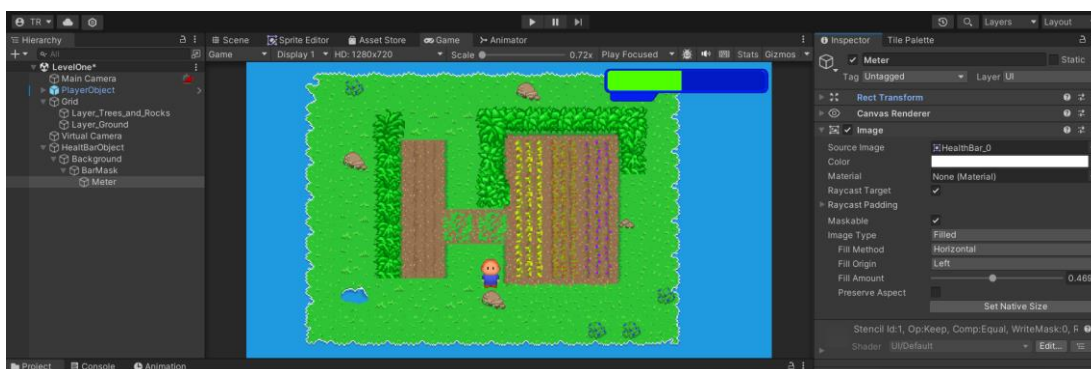
Con el objeto **BarMask** aún seleccionado, haga clic en el botón **Add Component** en la ventana Inspector y agregue un componente **"Mask"**



Haga clic con el botón derecho en **BarMask** y agregue un elemento de UI secundario de tipo: **Imagen**. Este es el mismo proceso que seguimos anteriormente cuando creamos BarMask. Llame a este objeto de imagen secundario: **"Meter"**. Establezca su imagen de origen en: **HealthBar_0** cambie el **Width** a: **240** y el **Height** a: **30**.



Seleccione el objeto **Meter** y, en el componente Imagen, cambie el Tipo de imagen a: **Filled**. Luego cambie el método de relleno a: **Horizontal**, y el Fill Origin a: **Izquierda**. Esta configuración garantizará que la barra de salud se llene desde izquierda a derecha, horizontalmente. Con el objeto **Meter** seleccionado, deslice el control **Fill Amount** hacia la izquierda y despacio.

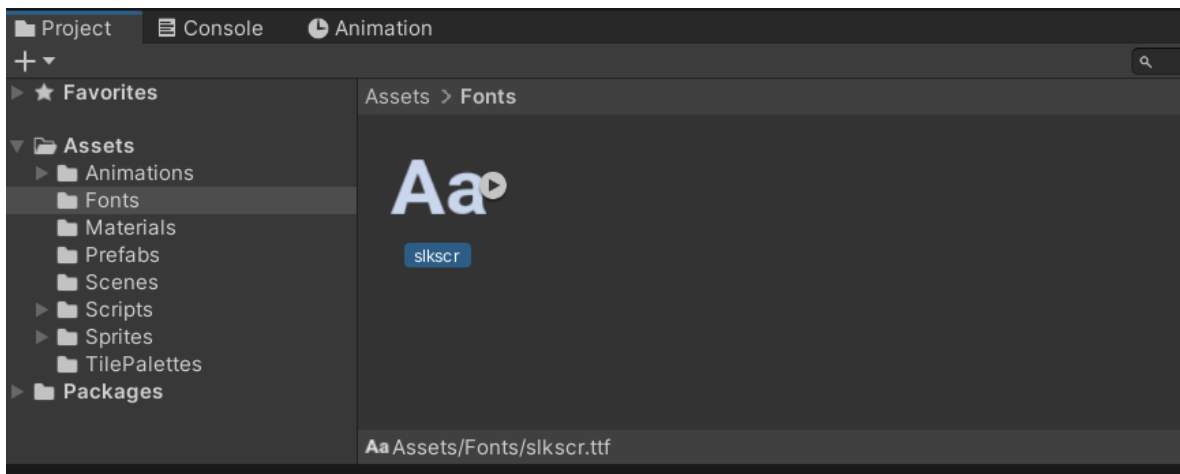


Importando Fuentes Personalizables

Es muy probable que desee utilizar fuentes personalizadas en su proyecto. Afortunadamente, es muy sencillo importar y utilizar fuentes personalizadas en Unity. Este proyecto incluye una fuente personalizada de libre acceso con un estilo retro llamado [Silkscreen](#). Silkscreen es un tipo de letra creado por Jason Kottke.

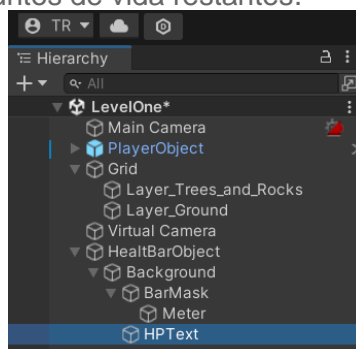
Haga clic con el botón derecho en la carpeta [Assets](#) en la vista Project y cree una nueva carpeta llamada "**Fonts**".

Arrastre y suelte el archivo de fuente, "[silkscr.ttf](#)", en la carpeta [Fonts](#) en su Proyecto de Unity para importarlo. Unity detectará el tipo de archivo y creará la fuente disponible en cualquier componente de Unity relevante.



Agregar texto de puntos de vida

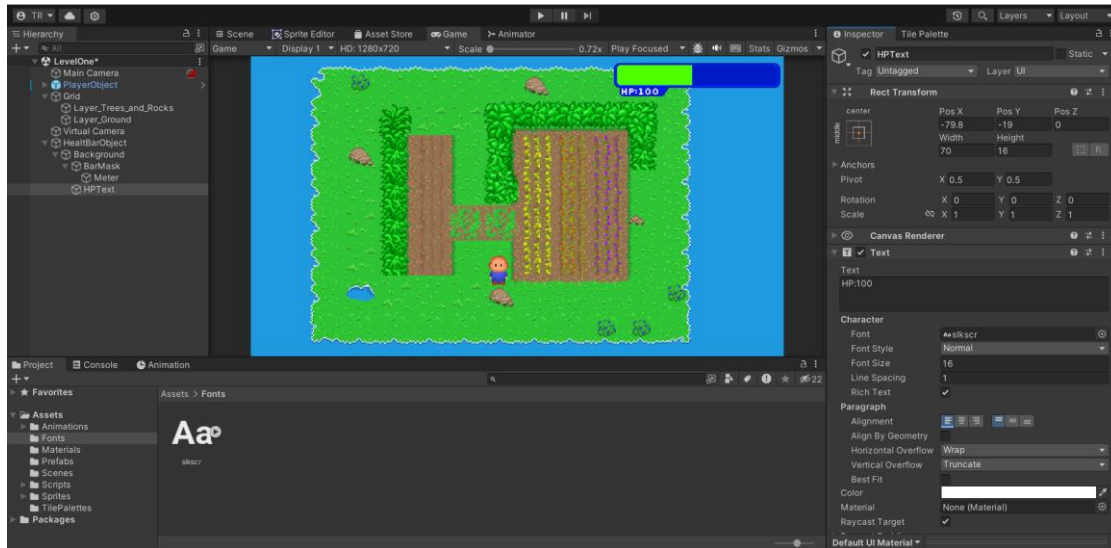
Haga clic con el botón derecho en el objeto [Background](#) y seleccione en el menú: **UI > Text**, para agregar un elemento de interfaz de usuario de tipo texto como elemento secundario del Background. Cambiar el nombre del objeto a, "**HPText**". Este objeto de texto mostrará el número de puntos de vida restantes.



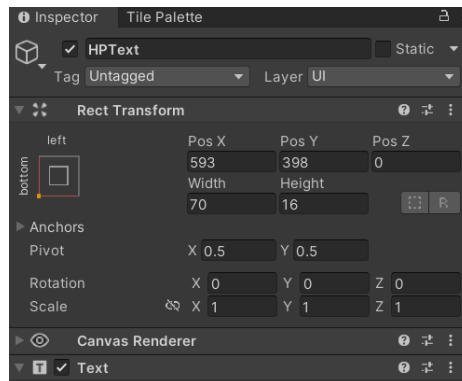
Gerardo Tonathiu Rojas Torres
Alexander Alonso Rodríguez

En el componente **Rect Transform** de **HPTText**, establezca el **Ancho** en: **70**, y **Altura** a: **16**. En el componente Texto de HPTText, cambie el **Tamaño de fuente** a **16** y el **Color** a **blanco**. Cambie la fuente a "**slkscr**", que es la fuente personalizada que acabamos de importar. Establecer el párrafo horizontal y vertical Alineación a la izquierda y al centro, respectivamente.

La imagen de la barra de salud tiene una pequeña pestaña en la parte inferior que proporciona un telón de fondo y mejora la visibilidad del texto. Mover el objeto HPTText en la bandeja para que se aparezca



Cambie los puntos de anclaje de HPTText para que estén en la parte inferior izquierda



Arrastre **HealthBarObject** a la carpeta Prefabs para crear un prefabricado y cambie el nombre del Prefab: **HealthBarObject**. No elimine **HealthBarObject** desde la vista Jerarquía

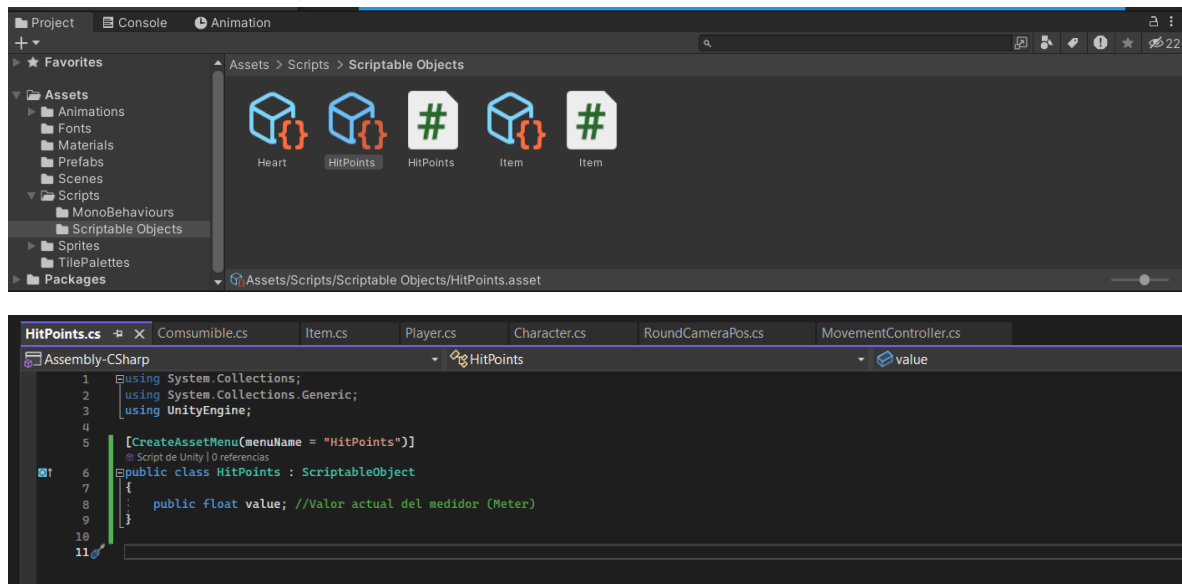


Gerardo Tonathiu Rojas Torres
Alexander Alonso Rodríguez

Scripting Barra de Vida

La clase **Player** hereda la propiedad: **hitPoints**, de la clase **Character**. En este momento, **hitPoints** es solo un tipo regular: **integer**. Los objetos programables comparten datos de puntos de vida entre **HealthBar** y la clase **Player**. El plan es crear una instancia de este objeto programable **HitPoints** y guarde el activo en la carpeta **ScriptableObjects**.

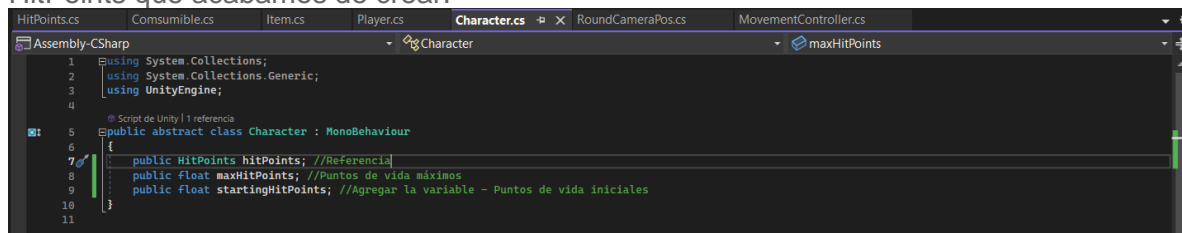
En la carpeta **ScriptableObjects**, haga clic con el botón derecho y cree un nuevo script llamado, **HitPoints**, y actualícelo para usar el siguiente código.



Utilice un float para mantener los puntos de golpe. Tendremos que asignar un float a la propiedad del objeto imagen: **Fill Amount**, en el objeto **Meter** de nuestra barra de salud, por lo que nos hace la vida un poco más fácil comenzar con un float.

Modifica el script de la Clase Character

Necesitamos hacer un pequeño cambio en el Script del Character para utilizar el Script de **HitPoints** que acabamos de crear.

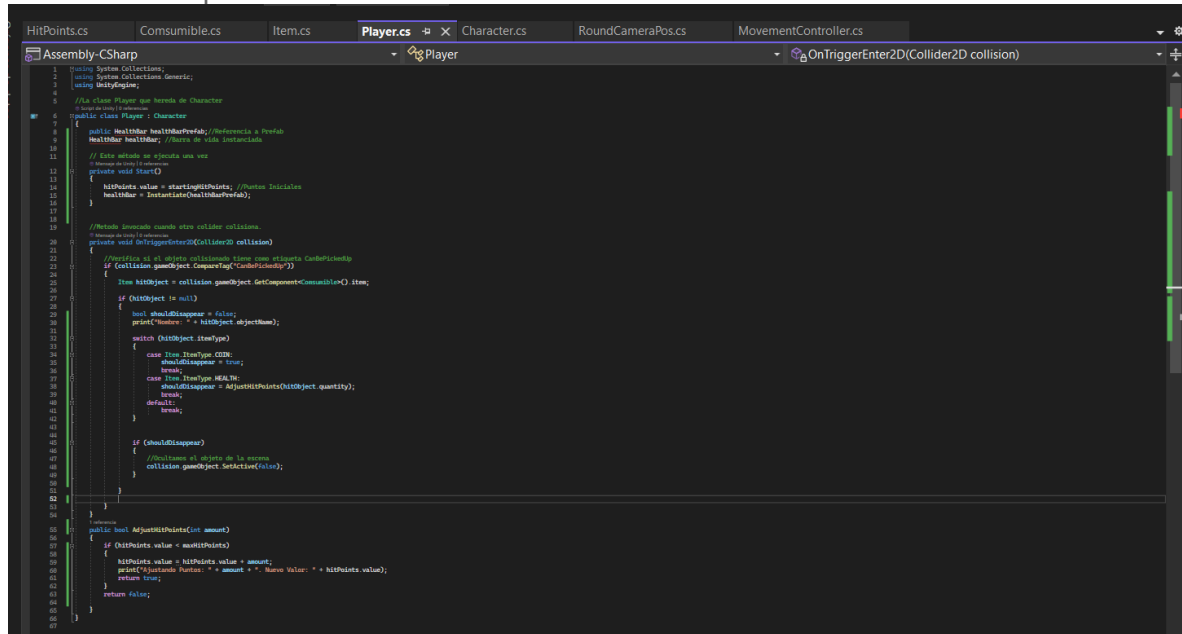


Gerardo Tonathiu Rojas Torres

Alexander Alonso Rodríguez

Clase Player

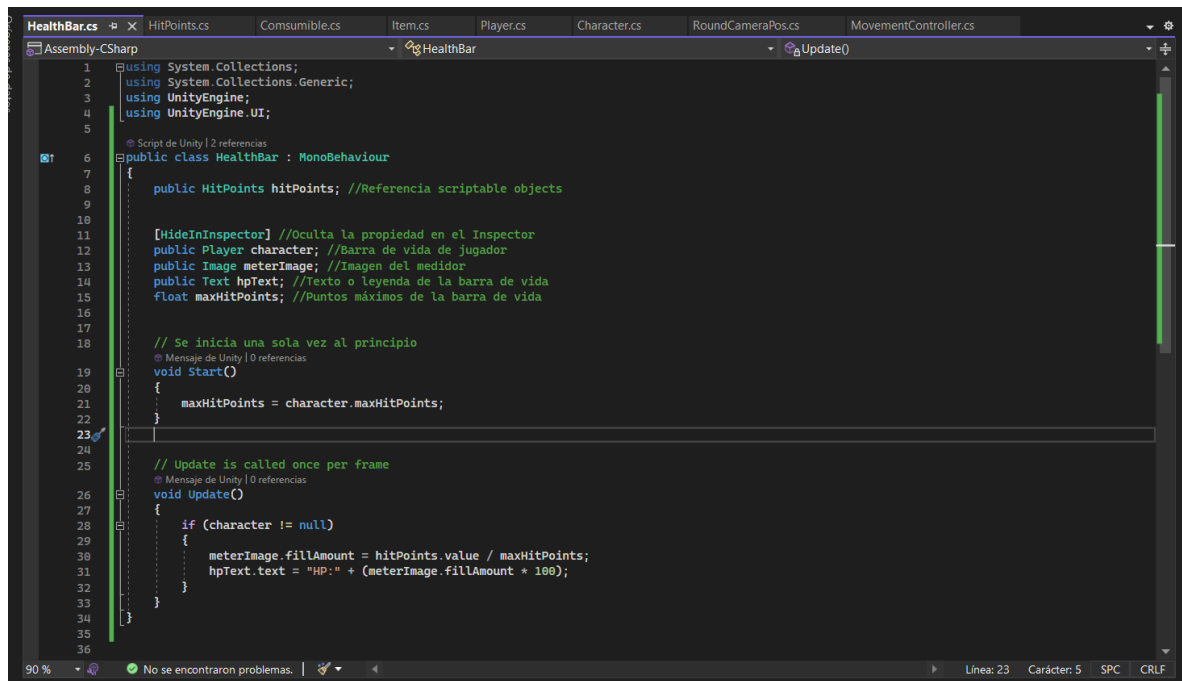
Modifica el script



```
1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4
5 //La clase Player que hereda de Character
6 public class Player : Character
7 {
8     //Este método se ejecuta una vez
9     //cuando se instancia el objeto
10    void Start()
11    {
12        hitPoints.value = startingHitPoints; //Puntos Iniciales
13        healthBar = Instantiate(healthBarPrefab);
14    }
15
16    //Método llamado cuando otro colisor colisiona.
17    //Mensaje de Unity | 2 referencias
18    private void OnTriggerEnter2D(Collider2D collision)
19    {
20        //Verifica si el objeto colisionado tiene una etiqueta Collider2D
21        if (collision.gameObject.CompareTag("Consumible"))
22        {
23            Item hitObject = collision.gameObject.GetComponent<ConsumibleItem>();
24
25            if (hitObject != null)
26            {
27                bool shouldDisappear = false;
28                print("Hit: " + hitObject.gameObject.name);
29
30                switch (hitObject.itemType)
31                {
32                    case Item.ItemType.CDIN:
33                        shouldDisappear = true;
34                        break;
35                    case Item.ItemType.HEALTH:
36                        shouldDisappear = AdjustHitPoints(hitObject.quantity);
37                        break;
38                    default:
39                        break;
40                }
41
42                if (shouldDisappear)
43                {
44                    //Destruye el objeto de la escena
45                    collision.gameObject.SetActive(false);
46                }
47            }
48        }
49    }
50
51    //Mensaje de Unity | 0 referencias
52    public bool AdjustHitPoints(int amount)
53    {
54        if (hitPoints.value < maxHitPoints)
55        {
56            hitPoints.value = hitPoints.value + amount;
57            print("Ajustando Hit: " + amount + ", Nuevo Valor: " + hitPoints.value);
58            return true;
59        }
60        return false;
61    }
62 }
```

Creando el script HealthBar

Haga clic derecho en la carpeta **MonoBehaviours|HealthBar**: script, creación y crear un nuevo C# llamado **HealthBar**

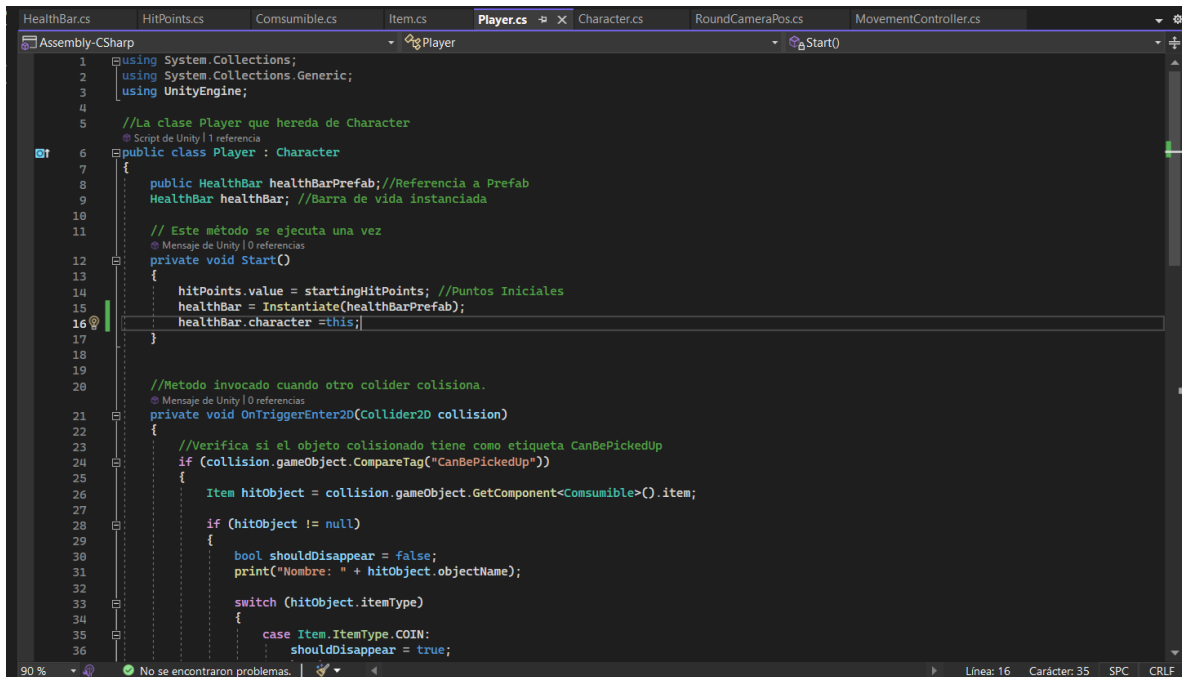


```
1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4 using UnityEngine.UI;
5
6 //Script de Unity | 2 referencias
7 public class HealthBar : MonoBehaviour
8 {
9     public HitPoints hitPoints; //Referencia scriptable objects
10
11     [HideInInspector] //Oculta la propiedad en el Inspector
12     public Player character; //Barra de vida de jugador
13     public Image meterImage; //Imagen del medidor
14     public Text hpText; //Texto o leyenda de la barra de vida
15     float maxHitPoints; //Puntos máximos de la barra de vida
16
17     // Se inicia una sola vez al principio
18     //Mensaje de Unity | 0 referencias
19     void Start()
20     {
21         maxHitPoints = character.maxHitPoints;
22     }
23
24     // Update is called once per frame
25     //Mensaje de Unity | 0 referencias
26     void Update()
27     {
28         if (character != null)
29         {
30             meterImage.fillAmount = hitPoints.value / maxHitPoints;
31             hpText.text = "HP:" + (meterImage.fillAmount * 100);
32         }
33     }
34 }
35
36
```

Gerardo Tonathiu Rojas Torres

Alexander Alonso Rodríguez

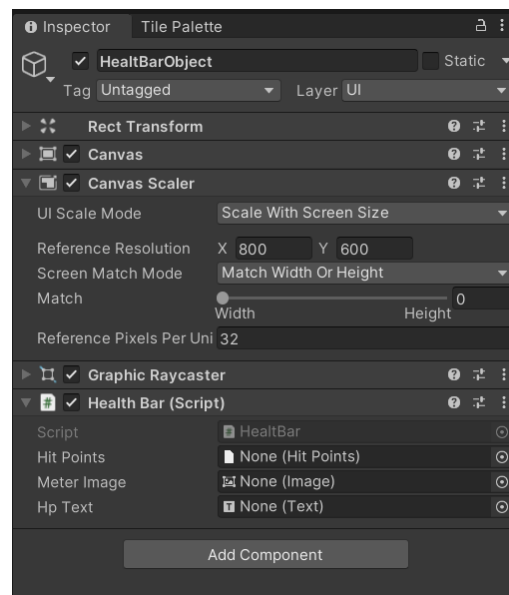
Modificamos método Start de Player



```
1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4
5 //La clase Player que hereda de Character
6 public class Player : Character
7 {
8     public HealthBar healthBarPrefab; //Referencia a Prefab
9     HealthBar healthBar; //Barra de vida instanciada
10
11     // Este método se ejecuta una vez
12     private void Start()
13     {
14         hitPoints.value = startingHitPoints; //Puntos Iniciales
15         healthBar = Instantiate(healthBarPrefab);
16         healthBar.character = this;
17     }
18
19     //Metodo invocado cuando otro colider colisiona.
20     private void OnTriggerEnter2D(Collider2D collision)
21     {
22         //Verifica si el objeto colisionado tiene como etiqueta CanBePickedUp
23         if (collision.gameObject.CompareTag("CanBePickedUp"))
24         {
25             Item hitObject = collision.gameObject.GetComponent<Consumible>().item;
26
27             if (hitObject != null)
28             {
29                 bool shouldDisappear = false;
30                 print("Nombre: " + hitObject.objectName);
31
32                 switch (hitObject.itemType)
33                 {
34                     case Item.ItemType.COIN:
35                         shouldDisappear = true;
36                 }
37             }
38         }
39     }
40 }
```

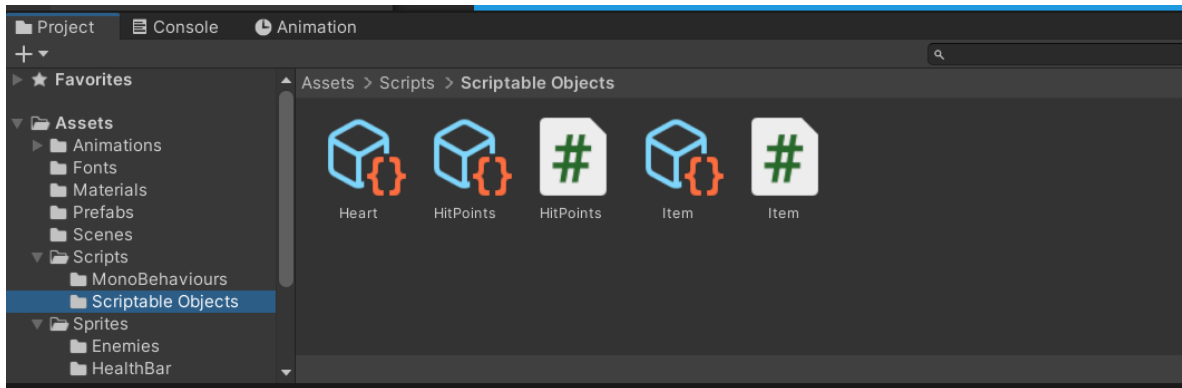
Configura el Health Bar

Vuelva al Editor de Unity y seleccione HealthBarObject de la carpeta Prefabs en la vista Proyecto. Modifique las propiedades del Objeto HealthBar

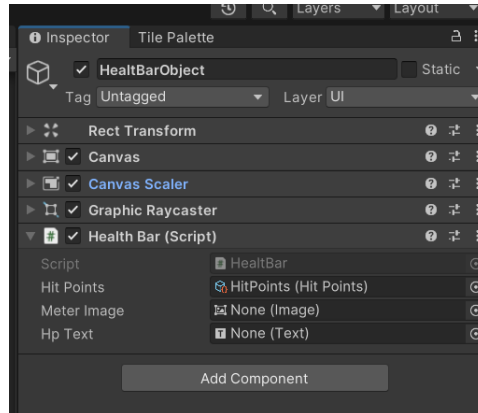


Gerardo Tonathiu Rojas Torres
Alexander Alonso Rodríguez

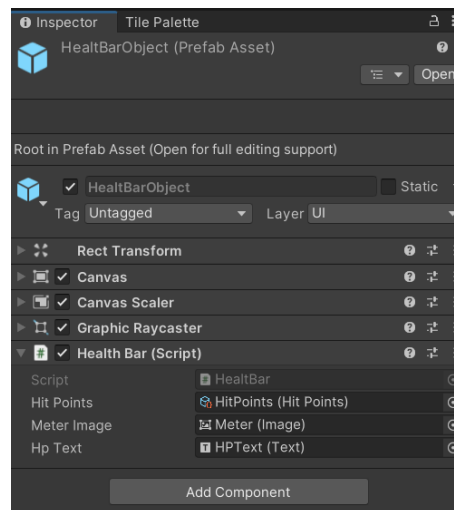
En la carpeta Scriptable Objects, haga clic con el botón derecho y use la opción de menú: Create ► HitPoints para crear una nueva instancia del objeto HitPoints. Cambiarle el nombre: "HitPoints",



Con HealthBarObject seleccionada, arrastre el objeto HitPoints a la propiedad HitPoints

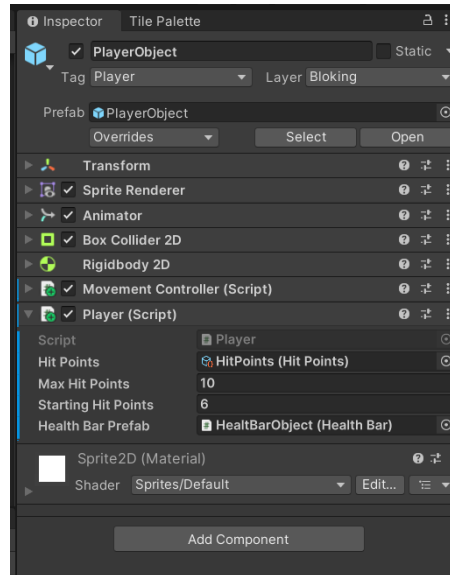


Seleccione HealthBarObject y haga clic en los pequeños puntos junto a cada uno de los propiedades en el script de la barra de salud. Seleccione el valor apropiado para cada propiedad



Gerardo Tonathiu Rojas Torres
Alexander Alonso Rodríguez

Cuando haya terminado, presione el botón Apply en el Inspector. Seleccione el Prefab del PlayerObject en la carpeta Prefabs. Arrastra los HitPoints del objeto programable que creamos en la propiedad HitPoints. Recuerde que estamos usando este mismo objeto HitPoints en Health Objeto de barra. Los datos de los puntos de vida se comparten entre dos objetos separados como magia



Presiona Play y camina con el jugador para recoger corazones. La barra de salud debe sumar 10 puntos cada vez que el jugador toma un corazón



