Documentación Sistema de Spawn

Índice

Este es un trabajo para Unity 3D, pensado como un sistema de spawner que puede ser integrado y personalizado según los objetos, áreas y velocidad de aparición que necesite el programa.

Así mismo contiene sistema de paneles que proporciona información de estos spawners dentro del mismo juego.

Instalación

1- Acceder al siguiente link para descargar el package de archivos de forma rápida:

https://github.com/Sp0my/Sistema-de-Spawn/blob/main/Spawn System/Packages/SpawnerSystem_v1.0.1.unitypackage

2- Al descargarlo y abrirlo en tu juego de unity, los elementos se encontrarán dentro de en una carpeta bajo el nombre de: **Spawn System.** Dentro de ella veras las carpetas: Prefabs y Scripts; dentro de la carpeta **Scripts** es donde encontrarás los siguientes archivos y subcarpetas:

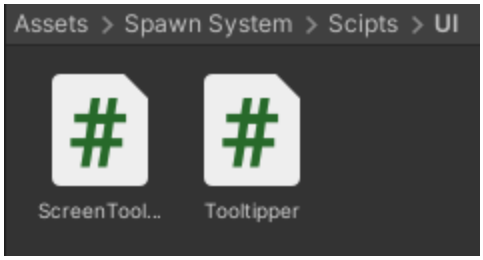
Contenido

▼ **Editor**

Esta carpeta contiene el script SpawnableEditor el cual sirve para controlar los prefabs que spawneen para poder reciclarlos.

▼ **UI**

En esta carpeta se encuentran los scripts: **ScreenTooltipManager** y **Tooltip**. Estos scripts son los controladores de la UI y la información que esta despliega.



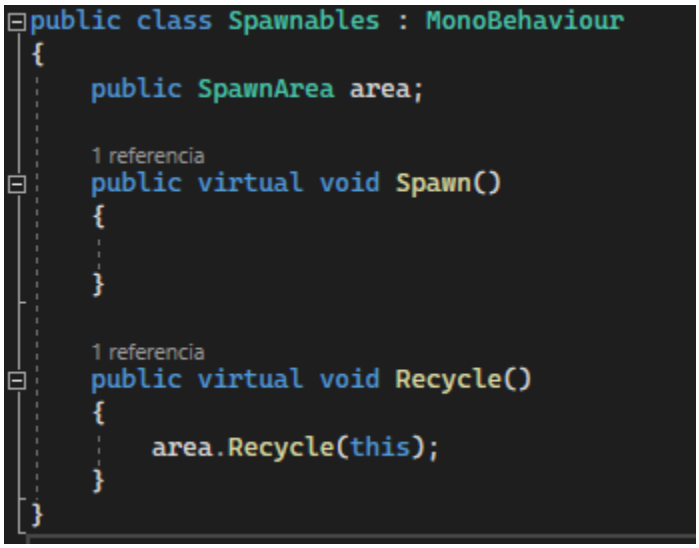
▼ **MathUtil**

Este es uno de los scripts más importantes pues es aquel que realiza los cálculos sobre el área en que los prefabs aparecerán.

```
public static class MathUtil
{
    1 referencia
    public static Vector3 PointInsideArea(Collider area)
    {
        Vector3 min = area.bounds.min;
        Vector3 max = area.bounds.max;
        Vector3 point = new Vector3();
        bool gotHit = false;
        RaycastHit hit;

        for (int i = 0; i < 100 && !gotHit; i++)
        {
            point.x = Random.Range(min.x, max.x);
            point.y = Random.Range(min.x, max.y);
            point.z = Random.Range(min.x, max.z);

            Ray ray = new Ray(point + new Vector3(0, 100, 0), Vector3.down);
            gotHit = area.Raycast(ray, out hit, 100.0f);
        }
        return point;
    }

    1 referencia
    public static Vector3 SnapToGround(Vector3 point, Collider ground)
    {
        Vector3 origin = new Vector3(point.x, ground.bounds.max.y + 10, point.z);
        Ray ray = new Ray(origin, Vector3.down);
        RaycastHit hit;


        if (ground.Raycast (ray, out hit, ground.bounds.extents.y + 20.0f))
        {
            point = hit.point;
        }
        return point;
    }
}
```

▼ Spawnables

Este script funciona como almacenador de variables y funciones base.



```
public class Spawnables : MonoBehaviour
{
    public SpawnArea area;

    1 referencia
    public virtual void Spawn()
    {

    }

    1 referencia
    public virtual void Recycle()
    {
        area.Recycle(this);
    }
}
```

Este script deberá ser colocado en el prefab que vayamos a colocar dentro del área de spawn.

▼ SpawnArea

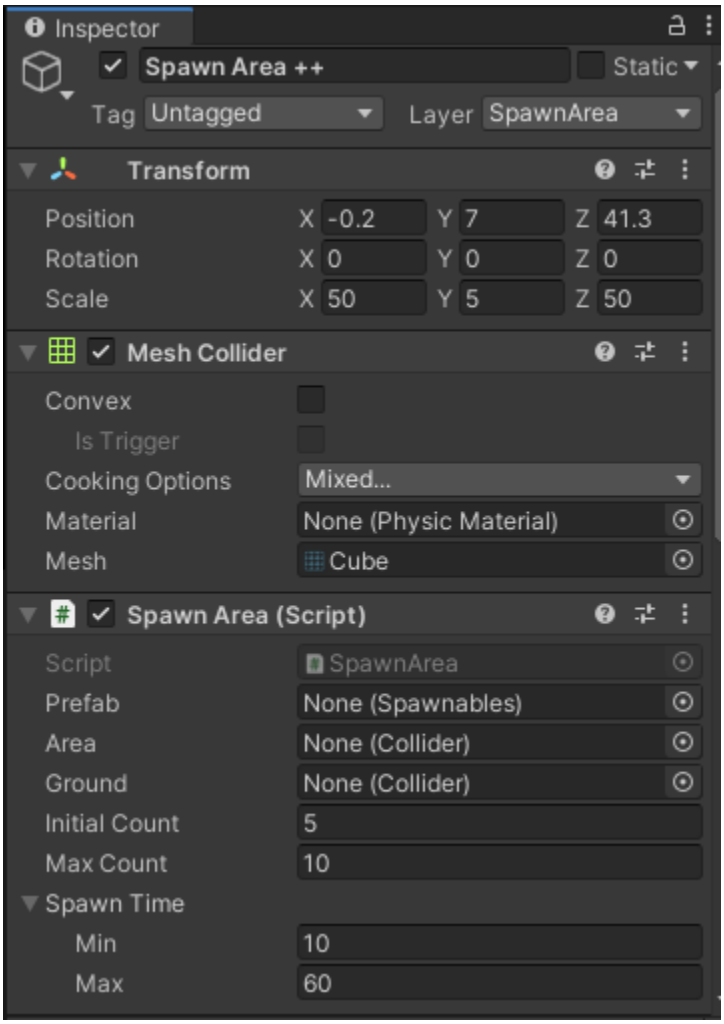
Este script es otro de los más importantes ya que es el que controlará y manejará la aparición y configuración de los spawnables; desde su velocidad de aparición hasta la cantidad que aparecerá.

```
public class Range
{
    public float min;
    public float max;

    0 referencias
    public Range() {min = 0; max = 0; }
    1 referencia
    public Range(float a, float b) {min = a; max = b; }
    2 referencias
    public float GetValueInRange()
    {
        return Random.Range(min, max);
    }
}
```

```
Script de Unity (3 referencias de recurso) | 1 referencia
public class SpawnArea : MonoBehaviour
{
    public Spawnables _prefab;
    public Collider area;
    public Collider ground;
    public int initialCount = 5;
    public int maxCount = 10;
    public Range spawnTime = new Range(10, 60);

    private List<Spawnables> _unUsed = new List<Spawnables>();
    private List<Spawnables> _used = new List<Spawnables>();
    private float timeUntilNextSpawn;

    Mensaje de Unity | 0 referencias
    void Start()
    {
        for (int i = 0; i < maxCount; i++)
        {
            Spawnables spawnables = Instantiate(_prefab);
```

Configuración

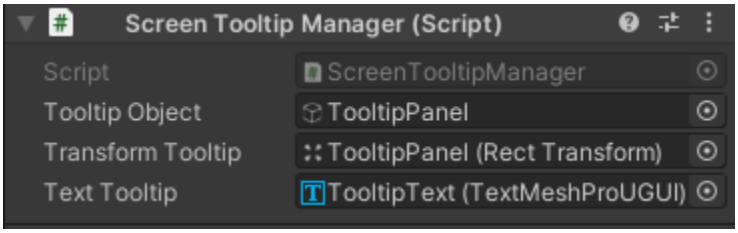
1- Para comenzar, deberás crear el área de spawn; esta puedes ser personalizada o un objeto 3D de unity. Lo importante será que este objeto, al ser colocado en el Hierarchy, solo deberá contener su **Mesh Collider**, a este objeto le agregaremos el script **Spawn Area.**



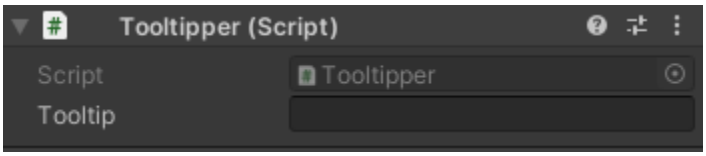
2- Aquí podremos comenzar a configurar nuestra área de spawn. En **Prefab** debemos arrastrar el objeto que queremos que aparezca en esta área. En **A rea** debemos arrastrar el Mesh de la zona que creamos. En **Ground** pondremos el objeto del piso sobre el que aparecerán los prefabs, este objeto puede ser irregular en su superficie, el Raycast se encargará de leer la superficie.

Seguido tendremos la cantidad inicial con la que queremos aparezcan los objetos, teniendo por defecto 5 como mínimo y 10 como máximo, pero pudiendo ser modificada al número que se desee. Finalmente tenemos el tiempo de aparición, estando por defecto en 10 segundos como mínimo y 60 como máximo, siendo este un intervalo aleatorio, pudiendo aparecer entre estas dos cantidades de tiempo.

3- Deberemos crear un Panel en el Hierarchy y en el Event System creado colocaremos el script: **ScreenTooltipManager**, en sus dos primeros apartados colocaremos el Panel y en el siguiente un objeto **TextMesh Pro** al cual haremos hijo del panel.



4- Ahora, ara obtener esta información en pantalla, usaremos los scripts dentro de la carpeta **UI**. Primero arrastraremos el script Tooltipper al objeto del área en el Hierarchy, dejando vacío el espacio de Tooltip.



Y seguido de esto colocaremos un Box Collider en este mismo objeto, este servirá para que el mouse detecte nuestra área y que active el panel en pantalla.