Introducción a los scripts Unity3D

1. Objetivos

Los scripts son una parte esencial de Unity3D ya que definen los comportamientos de tu videojuego. En este tutorial introduciremos los conceptos básicos de creación de scripts con el lenguaje Javascript. No son necesarios conocimientos previos de Javascript para seguirlo.

Además de Javascript, C# y Boo son otros lenguajes con los que poder crear scripts en Unity3D. En este tutorial también introduciremos los elementos clave de la API (Application Program Interface). La API es básicamente código que ha sido previamente escrito para ti de forma que puedas centrarte directamente en el diseño de tu videojuego acelerando el tiempo de desarrollo.

2. Prerequisitos

Vamos a centrarnos en la creación de scripts y asumimos que ya estamos suficientemente familiarizados con la GUI de Unity3D que trabajamos en el tema anterior.

Para hacer el scripting más fácil de entender, es preferible tener un editor de código que soporte un resaltado de sintaxis para Javascript. Esto significa que las palabras reservadas (La sintaxis que viene de serie con Javascript) sea coloreada de forma distinta que las palabras definidas por el usuario. Un editor con estas características sería SubEthaEdit.

Para facilitarte las cosas, nuevamente usaremos un guión (-) para prefijar todas las acciones que requieran tu intervención directa con el programa.

3. Convenios de nombre

Antes de empezar, vamos a mencionar varios convenios de nombrado de Unity3D:

Variables: Empiezan con letra minúscula. Las variables sirven para almacenar información sobre aspectos de algún estado del videojuego.

Funciones: Empiezan con letra mayúscula. Las funciones son bloques de código que se escriben una sóla vez y pueden ser reutilizadas tantas veces como sea necesario.

Clases: Empiezan con letra mayúscula. Son básicamente colecciones de funciones.

Siempre que leas ejemplos de código de Unity3D o la propia API de Unity3D, presta especial atención a la primera letra de cada palabra. Te ayudará a entender mejor las relaciones entre objetos.

4. Entrada de datos

En nuestro primer programa vamos a permitir al usuario moverse dentro de un mundo sencillo del videojuego.

4.1 Configurando la escena

- Arranca Unity3D

En primer lugar crearemos una superficie sobre la que pueda caminar el usuario. La superficie que vamos a usar es un cubo aplanado tal y como hicimos en el tema anterior:

- Crea un cubo y escala sus dimensiones x, y, z a 5,0, 0,1, 5,0, respectivamente, verás que se muestra como un gran plano. Renombra este objeto como Plano en la Vista de Jerarquía.
- Crea un segundo cubo y colócalo en el centro del Plano. Si no puedes ver los objetos en tu Vista de Videojuego, modifica la cámara principal para que sean visibles. Renombra este cubo como Cubo1.
- Deberías también crear una luz puntual y colocarla sobre los cubos de forma que sean más fáciles de ver.
- Guarda la escena haciendo clic en la pestaña File → Save As y dale un nombre al videojuego.

4,2 Nuestro primer script

Ya estamos listos para empezar a programar nuestro videojuego. Vamos a permitir al jugador moverse por el mundo controlando la posición de la cámara. Para hacerlo vamos a escribir un script que leerá la entrada de teclado, a continuación asociaremos el script con la cámara principal.

- Empieza creando un script vacío. Haz clic en la pestaña Assets → Create → Javascript y renombra el script como Mover1 en el panel de la Vista de Proyecto.
- Haz doble clic en el script Mover1 y se abrirá la función Update ( ) ya insertada en él, puesto que

es el comportamiento por defecto. Vamos a insertar el código dentro de esta función. Hay que tener en cuenta que todo el código dentro de la función Update ( ) se ejecutará en cada frame.

Para modificar la posición de un objeto del videojuego en Unity3D necesitamos alterar la posición en su Transform. El Transform el objeto que almacenan la posición, rotación y escala de un objeto, puedes encontrar más documentación sobre Transform aquí.

La función Translate de Transform nos permitirá hacer esto. Translate recibe 3 parámetros que son el movimiento en cada uno de los ejes x, y, z. Como queremos controlar la cámara con las teclas cursor, simplemente podemos añadir código para determinar si las teclas cursor están siendo pulsadas y ponerlas como parámetros x, z.

```
function Update () {
        Transform.Translate ( Input.GetAxis ( "Horizontal" ) , 0 , Input.GetAxis ( "Vertical" ) );
}
```

Input es una clase, y la función GetAxis que estamos invocando nos devuelve un valor entre -1 y 1. Por ejemplo en el caso del eje Horizontal, la tecla direccional izquierda nos devuelve -1 y la tecla direccional derecha nos devuelve 1.

Observa que para invocar una función de un objeto utilizamos el operador punto (.).

Finalmente, Transform es una clase, y la función Translate, como hemos dicho anteriormente, recibe 3 parámetros que indican el movimiento en cada eje. De esta manera el código anterior se encarga de modificar la posición en los ejes x, z con las teclas. No nos interesa movernos en el eje y, por eso ponemos un 0 en el parámetro correspondiente.

Los ejes horizontal y vertical están predefinidos en la configuración de entrada (Input Settings). Los nombres y las teclas mapeadas pueden cambiarse fácilmente si accedemos a la pestaña Edit → Project Settings → Input.

— Abre el script Mover1 y escribe en él el código que hemos mostrado anteriormente, presta especial atención a las letras mayúsculas.

4.3 Asociando el script

Ahora que ya tenemos escrito nuestro primer script, ¿Cómo podemos decirle a Unity3D cuál es el objeto del videojuego que queremos que tenga ese comportamiento?. Todo lo que tenemos que hacer es asociar el script al objeto del videjuego para el que queremos ese comportamiento.

- Para hacerlo, en primer lugar haz clic en el objeto del videojuego al que quieras asociar el comportamiento definido en el script. En nuestro caso, sería la cámara principal, puede seleccionarla bien sea en la Vista de Jerarquía o en la Vista de Escena.
- A continuación haz clic en la pestaña Componentes → Scripts → MoverI. Esto asocia el script a la cámara, observa que el componente MoverI ahora también aparece en la Vista de Inspector de la cámara principal.

También es posible asignar un script a un objeto simplemente arrastrando el script desde la Vista de Proyecto hasta el objeto en la Vista de Escena.

- Ejecuta el videojuego (Pulsa el botón Play), deberías ser capaz de mover la cámara utilizando las teclas direccionales o bien las teclas A,S,D y W.

Seguramente notarás que la cámara se mueve demasiado rápido, vamos a ver una manera más eficaz de controlar la velocidad de la cámara.

4.4 Delta Time (Tiempo Delta)