**ANIMACIÓN 3D EN UNITY**

**1 - Importación de animaciones y modelos con rig a Unity**

Las animaciones de objetos 3D, así como los meshes y materiales de estos modelos, vienen todos dentro de un mismo tipo de asset. El tipo de archivo de este asset puede variar, puede ser .blend, .obj o .fbx, pero tienen en común que son archivos que guardan modelos y animaciones 3D. Una vez se importe uno de estos archivos a Unity, serán tratados de la misma manera por el motor y podremos ver las mismas propiedades en el inspector. Por simplicidad, nos referiremos a estos paquetes de modelos y animaciones como FBX.

Un FBX se presenta en el inspector en 4 pestañas. Algunos tópicos de la pestaña de animación se analizan en detalle en la sección 2 de este mismo documento, porque están relacionados directamente con temas posteriores.

**1.1 - Pestaña de Modelo**

La primera pestaña, la pestaña de modelo, nos ofrece varios settings de importación que tienen que ver con el mesh. Desde acá se puede controlar el nivel de compresión del mesh o se puede activar la habilidad para poder modificar el mesh con código. Muy rara vez tendremos que jugar con los settings presentados acá, pero hay 1 en particular que es muy importante:

Scale Factor: El scale factor decide qué tamaño tendrá el mesh cuando lo veamos en la escena. Cuando añadamos este objeto a la escena se verá más grande o más pequeño de acuerdo a esta propiedad. Un error común es cambiar el tamaño de los modelos usando la propiedad de escala del transform del objeto. Esto puede causar problemas cuando utilizamos código para hacer que objetos se vuelvan padres o hijos de otros. Si un objeto se vuelve hijo de un objeto escalado, su escala será modificada automáticamente, y si luego hacemos que deje de ser hijo, nuestro objeto mantendrá la escala modificada y se verá deformado o de distinto tamaño. Intenten siempre cambiar el tamaño de los modelos usando el scale factor y mantener la escala del transform en (1, 1, 1).

**1.2 - Pestaña de Rig**

Esta pestaña contiene los settings acerca del rig (esqueleto) del modelo que estamos importando.

**Animation Type:** El animation type decide si nuestro esqueleto será tratado como humanoide, genérico o si no lo consideraremos (none). Genérico es cualquier tipo de esqueleto que no sea humanoide, como una araña o una grúa. Si nuestro modelo tiene un esqueleto humanoide es una buena idea seleccionar Humanoid como animation type. Un esqueleto humanoide tiene la ventaja de poder compartir sus animaciones con otros modelos que también sean humanoides.

**Avatar Definition:** Un avatar es lo que le permite a las animaciones del modelo mover los huesos apropiados del rig de este. Modificar esta propiedad es más relevante cuando tratamos con modelos humanoides. Cuando el animation type es “humanoid” el avatar funciona como una abstracción que le permite a los huesos de un modelo ser animados por la animación de otro. Al poner aplicar en la parte baja de esta pestaña, Unity intentará interpretar el esqueleto como un humanoide, identificando cada hueso de este con el hueso de un ser humano. Si Unity no logra hacer esta relación podemos hacer el mapeo manualmente apretando el botón “configure”.

Entonces, lo más relevante de la pestaña de rig es definir si tenemos esqueleto y si este es genérico o humanoide.

**1.3 - Pestaña de Animación, aspectos básicos**

La pestaña de animación es probablemente el lugar donde pasaremos más tiempo configurando cosas en un FBX. A continuación se presentan los aspectos básicos de esta pestaña.

Las animaciones que un FBX trae consigo están todas adyacentes en una sola línea de tiempo. Al importar el modelo esta línea de tiempo puede estar subdividida en varias sub-animaciones como puede no estarlo.

Cuando queramos extraer una animación de la línea de tiempo apretamos el botón de ‘+’ en la lista de animaciones (clips) y seleccionamos la sección que necesitamos utilizando los sliders en la línea de tiempo.

Al tener seleccionado un clip podemos decidir si la animación hará loop o correrá una sola vez activando o desactivando la propiedad “Loop Time” en esta misma pestaña.

**1.4 - Pestaña de Materiales**

Por defecto no podemos modificar los materiales o las texturas que un modelo trae consigo, pero a través de esta pestaña podemos extraerlos para que, en lugar de estar contenidos en el asset FBX, estén separados y sean editables. Si el material no se importó bien y nuestro modelo se ve transparente o de alguna manera extraño, extraer los materiales y configurarlos manualmente es muchas veces la solución.

**2 - Animación 3D**

Las bases de la animación en Unity son las mismas en 2D que en 3D: Tenemos controladores de animación, transiciones, estados, parámetros etc. En 3D eso sí, tenemos un tópico único llamado Root Motion.

Cuando tenemos un objeto con rig en la escena de Unity esté estará jerarquizado de tal manera que tendrá un objeto padre con un Animator. Este objeto tendrá como hijos una jerarquía de huesos y un objeto que contendrá el SkinnedMeshRenderer.

**Nota:** Cuando hablemos del movimiento del **objeto** en las siguientes secciones, nos estaremos refiriendo al objeto padre de los huesos y del renderer.

**2.1 - Root Motion**

Por definición las animaciones 3D mueven huesos que, en última instancia, son transforms en algún lugar de la escena. Los huesos están jerarquizados de tal manera que todos convergen en un mismo padre, llamado “root”, la raíz del objeto. En el caso de los esqueletos humanoides, este hueso raíz es la cadera. Si una animación 3D importada mueve este hueso raíz, decimos que esa animación tiene **root motion**.

Una animación con root motion puede resolverse de dos formas:

1 - Hacer que el objeto completo se desplace en el espacio si la animación mueve el objeto raíz. El desplazamiento del objeto lo realiza nuestra animación.

2 - Ignorar el movimiento del objeto raíz. La animación jamás será capaz de desplazar el objeto en la escena. El desplazamiento del objeto lo tendremos que escribir con código.

Al trabajar en 2D nunca hemos utilizado root motion. El desplazamiento de nuestros objetos lo hemos resuelto siempre con código. El root motion nos da la posibilidad de que nuestro objeto sea desplazado por su animación y no por código.

En el componente Animator tenemos una propiedad booleana llamada **Apply Root Motion**. Si queremos la opción 1 activamos este toggle, y si queremos la 2, lo desactivamos.

Queremos hacer que la animación desplace a nuestro personaje cuando buscamos movimientos realistas. La animación mueve a nuestros personajes de acuerdo a la posición de sus pies en el piso y nuestro personaje parecerá estar firme sobre él.

No queremos tener root motion cuando buscamos un control más fino sobre el movimiento de nuestros personajes. Puede que la animación lo desplace más lento de lo que quisiéramos o puede que queramos ir variando la velocidad a la que se mueve. Si queremos control total sobre la velocidad de un objeto, desactivamos apply root motion.

En caso de aplicar el root motion se volverá relevante conocer las propiedades de la pestaña de Animation de los FBX.

**2.2 - Pestaña de Animación, opciones avanzadas**

Hay tres propiedades acá que tienen que ver con root motion: Root Transform Rotation, Root Transform Position (Y) y Root Transform Position (XZ).

Estas propiedades descomponen el root motion del objeto en 3 partes: rotación, movimiento vertical y movimiento sobre el plano del piso, respectivamente.

Lo que nos interesa conocer respecto a cada uno de estos componentes del root motion es el significado de la opción “bake into pose” que tiene cada una de ellas.

Si estamos aplicando el root motion en nuestro componente Animator (ver más arriba), bake into pose **impide que este componente de root motion mueva el objeto**. Para poner un ejemplo, si pongo “bake into pose” en el componente de root motion “Root Transform Position (Y)”, entonces el movimiento de la raíz no controlará la posición en el eje Y del objeto. Esto quiere decir que la animación podrá seguir controlando su movimiento horizontal y su rotación, por ejemplo, pero no controlará cuanto se mueve hacia arriba o hacia abajo el personaje.

El resultado de esto es que, cuando “bake into pose” esté seleccionado, el esqueleto seguirá moviéndose, pero no acarreará su movimiento al objeto padre.

**2.3 - Blend Trees**

Aunque pueden utilizarse en 2D también, es más común encontrar blend trees en animaciones 3D.

Un blend tree existe como un estado en una máquina de estados de animación. La diferencia es que este estado, en lugar de contener una sola animación, contiene varias, y las mezcla.

El blend tree lo que hace es mezclar las animaciones que lo componen en distintas proporciones. Por ejemplo, si un blend tree está compuesto de 2 animaciones, podría dar como resultado una mezcla de un 80% de la primera y un 20% de la segunda.

Las proporciones en que un blend tree mezcla sus animaciones están dadas por uno o más parámetros float en el animator.

A continuación se explicará lo recién descrito con un ejemplo y se mostrará como crear blend trees en el proceso.

Imaginemos que queremos crear un solo estado que contenga todas las animaciones en que nuestro personaje está parado sobre el piso. Esto incluye las animaciones de estar quieto, caminando y corriendo, y deja fuera las animaciones de saltar y caer, por ejemplo.

Lo primero que hacemos es hacer click secundario en algún punto vacío en nuestra máquina de estados y seleccionar “from new blend tree”.

A continuación hacemos doble click en el estado recién creado. Esto nos llevará “dentro” del blend tree.

En el inspector podremos ver una lista vacía. Esta lista contendrá todas las animaciones que mezclaremos. En nuestro caso pondríamos ‘+’ en la lista 3 veces y asignaríamos a cada ítem las animaciones de “quieto”, “caminar” y “correr” respectivamente.

Tenemos un blend tree con 3 animaciones, pero para que Unity sepa en qué proporción mezclar cada una necesita un parámetro de tipo float. En este caso crearíamos uno llamado “speed”. En la parte superior del inspector seleccionamos como parámetro de mezcla el “speed” recién creado.

Ahora, a medida que nuestro speed cambie, nuestro blend tree mezclará las animaciones que lo componen en distintas proporciones. Esto dará como resultado que, a medida que el speed aumente, nuestro personaje pasará gradualmente de estar quieto a caminando y de caminando a corriendo.

Para que esto último se cumpla, debemos asignarle a cada animación en la lista del blend tree un valor de speed. Unity lo hace por nosotros pero podemos modificarlos en las propiedades de la lista si queremos.

Acabamos de hablar de blend trees de 1 dimensión, es decir, blend trees controlados por un solo parámetro. Podemos crear también blend trees de dos dimensiones, en la parte de arriba del inspector. Funcionan exactamente igual que los de 1 dimensión pero a cada animación de la lista hay que asignarle dos valores, uno para el primer parámetro y uno para el segundo.