**ANIMACIÓN EN UNITY**

En Unity una animación puede ser más que solo un efecto visual. Todas las propiedades que son visibles desde el inspector pueden ser animadas. Esto quiere decir que, por ejemplo, puedo animar el multiplicador de gravedad de un rigidbody, haciendo que un objeto comience a levitar.

**1 - Creación de animaciones**

Antes de detallar la creación de animaciones es importante explicar en qué consiste una. Una animación en Unity, de modo similar a otros sofwares, está compuesta por keyframes. Un keyframe es una marca en una línea de tiempo. Cada uno de estos keyframes tiene anotado una serie de valores para una o más propiedades del objeto siendo animado. Por ejemplo, un keyframe puede indicar que la posición en X del personaje es 3. A medida que la animación avanza en la línea de tiempo, **interpola** entre los keyframes. Esto quiere decir que si un keyframe indica que mi posición es 3 y más adelante otro keyframe indica que mi posición es 7, cuando esté en la mitad entre un keyframe y el otro tendré una posición de 5.

Es útil trabajar con keyframes, pues nos evita tener que detallar valores para cada posición en la línea de tiempo. Sólo tenemos que marcar los valores clave, y Unity automáticamente pasará por todos los valores intermedios.

Para crear una animación en Unity tenemos más de una opción. La más fácil y directa es seleccionar el objeto que queremos animar en la jerarquía y abrir la pestaña de animación. esta pestaña se puede encontrar en el menú “Window” de Unity.

Tras abrir esta ventana y tener el objeto seleccionado, aparecerá un botón que nos permitirá crear una nueva animación. Si lo seleccionamos podemos nombrar nuestra animación y guardarla en alguna carpeta dentro de nuestro proyecto. La animación creada es un asset, como los scripts y texturas de nuestro juego, y existe en nuestro proyecto.

Tras haber hecho esto se nos presentará en la ventana de animación una línea de tiempo y podemos empezar a animar.

Para animar debemos apretar el círculo rojo de “record” que se encuentra arriba a la izquierda de la ventana de animación. Al hacer esto estamos grabando, y todos los cambios que le hagamos al objeto se guardarán en el frame que esté actualmente seleccionado. De esta forma se crearán keyframes automáticamente.

Una vez nuestra animación esté terminada hay que asegurarse de dejar de grabar, apretando el botón rojo mencionado anteriormente.

Si ahora corremos nuestro juego podremos ver nuestra animación en acción, repitiéndose una y otra vez continuamente. También podemos apretar el botón de play que hay en la ventana de animación. La última opción es buena para la mayoría de los casos, pero si animamos propiedades físicas, como masa y gravedad, no podremos ver con este método los efectos de nuestra animación.

Por defecto las animaciones nuevas que creemos estarán en un loop infinito (se repetirán una y otra vez). Si queremos que nuestras animaciones se ejecuten una sola vez debemos seleccionar el asset en nuestro proyecto y cambiar el parámetro “loop time” en el inspector.

Para crear una segunda animación podemos, en la misma ventana, hacer click arriba a la izquierda donde aparece el nombre de la animación actual. Se nos desplegará una lista con todas las animaciones que hayamos creado. Al final de esta lista se encuentra la opción de añadir una nueva.

**2 - Creación de máquinas de estado**

En general, cuando queremos animar un objeto o un personaje al menos medianamente complejo, queremos que pueda pasar de una animación a otra de acuerdo a ciertas condiciones. Por ejemplo, queremos que pase de estar corriendo a saltando según el input del jugador. Cada animación que ejecutemos es necesariamente un estado en lo que se conoce como una máquina de estados de animación.

Una máquina de estados de animación puede visualizarse como un montón de animaciones conectadas con flechas unas con otras. El conjunto indica a qué animaciones puedo transitar según el estado (animación) en el que me encuentre. Por ejemplo, una máquina de estados para un personaje podría tener una transición de Corriendo a Disparando, pero no tener una transición de Saltando a Disparando, para evitar que el jugador pueda pasar a disparar mientras está saltando.

Una transición es algo más que una flecha con una dirección. La flecha puede indicar hacia dónde puedo transitar, pero no me dice **cuando** puedo transitar. Es por esto que cada transición (flecha) está asociada a una o más condiciones.

**2.1 - Máquinas de estado de animación**

Al crear cualquier animación, como se explica más arriba, automáticamente se crea una máquina de estados de animación. Si se revisan los componentes del objeto que se acaba de animar puede verse que tiene un nuevo componente llamado Animator, que alberga dentro suyo una máquina de estados de animación. Esta máquina de estados también es un asset, y está guardada por defecto en la misma carpeta donde se encuentra la animación que se creó anteriormente.

Si le hacemos doble click a este asset se nos abrirá una nueva ventana, llamada Animator. Acá se pueden visualizar los estados y transiciones de nuestra máquina. Por defecto siempre tenemos al menos una transición y un estado. Esta transición va desde un nodo que se llama Entry (el punto donde comienza nuestra máquina) hacia la única animación que tenemos.

Al añadir una segunda animación se nos creará un nuevo estado en nuestra máquina. Este estado estará desconectado del nodo de entrada y de nuestra animación anterior. Si queremos transitar de una a la otra debemos crear una transición.

**2.2 - Transiciones**

Para crear una transición debemos hacer click secundario en un estado, seleccionar crear transición y luego hacer click en el estado al que queremos ir. Esta transición se puede visualizar como una flecha.

Si hacemos click en esta flecha podemos ver en el inspector las propiedades de la transición. Este es el lugar donde definimos las condiciones de transición de las que hablábamos.

A continuación se describen las propiedades más importantes que podemos editar en las transiciones.

**Has Exit Time:** Este booleano define si para poder transitar debemos encontrarnos en un punto específico dentro de la animación. Si Has Exit Time está seleccionado (como lo está por defecto) significa que solo podemos pasar de una animación A a una B cuando lleguemos a algún punto dentro de A.

**Exit Time:** Si decidimos que queremos tener un tiempo de salida, acá podemos especificarlo. Este valor indica un punto dentro de la animación de origen desde donde podemos transitar. El valor está normalizado, lo que significa que un valor de 1 quiere decir el último frame de la animación y un valor de 0 el primero.

**Transition Duration:** Una transición no tiene por qué ocurrir de manera instantánea, puede demorarse un tiempo. Si un personaje está corriendo y transitamos a saltar es preferible que la transición no sea instantánea para que la animación no “salte”. Al asignarle una duración a una transición le decimos a Unity que interpole entre la animación de origen y la de destino por un tiempo, evitando saltos bruscos.

**Conditions:** Acá especificamos las condiciones para que la animación transite. Una condición puede representar por ejemplo cierto input del jugador, como cuando aprieta la tecla de saltar. No podemos vincular input directamente, en las condiciones solo podemos tratar con variables llamadas parámetros, que se describen a continuación. En esta lista de condiciones especificamos qué valor deben tener estas variables para que se cumpla nuestra condición.

**2.3 - Parámetros**

Un parámetro en una máquina de estados de animación es prácticamente lo mismo que una variable. Decidimos de qué tipo queremos que sea (int, bool, float) y le damos un nombre. Podemos hacer dos cosas con ellos.

Una de estas cosas es cambiarles el valor desde nuestro código, como se explicará en la siguiente sección.

Lo segundo que se puede hacer con un parámetro es utilizarlos para condiciones. Si creo un parámetro de tipo booleano llamado “corriendo” puedo utilizarlo para una condición como “corriendo es igual a falso”. Cuando este parámetro se vuelva falso, entonces, ejecutaremos una transición, por ejemplo desde una animación de correr a una en estado quieto.

**3 - Control de animación con código**

Teniendo una máquina de estados con transiciones y parámetros definidos, conectar esta misma con nuestro código es muy sencillo.

Primero necesitamos tener una referencia a un componente Animator. Esto lo podemos hacer a través de variables públicas o usando la función GetComponent. Este componente es el que alberga nuestra máquina de estados.

Una vez tenemos una variable con una referencia a nuestro componente, podemos llamar a una de varias funciones disponibles. Todas estas funciones comienzan con la palabra “Set”: SetBool, SetFloat, SetInt. Cada una cambia parámetros de un tipo concreto. Todas funcionan de la misma manera. Aceptan dos parámetros, un string con el nombre del parámetro que queremos cambiar y uno con el valor que queremos darle.

Por ejemplo para cambiar un parámetro booleano llamado “corriendo” a verdadero escribimos:

animator.SetBool (“moviendose”, true);