PROGRAMACION DE VIDEOJUEGOS III

Moviendo un sprite (parte I)

En el tutorial anterior aprendimos como crear una escena y agregar un sprite a la misma. A continuación veremos dar a los sprites movimiento utilizando el teclado.

La *Fig. 2* muestra el contenido del archivo *PlayState.hx*, el cual es muy similar al del ejemplo analizado antes. En el código correspondiente al nuevo ejemplo, se puede observar que se ha incluido una nueva clase/biblioteca (**FlxG**) y se han agregado algunas líneas de código al método *update()*.

La clase **FlxG** es una clase global de ayuda que facilita el acceso a dispositivos de audio, entrada, al sistema de cámaras, al depurador y otras propiedades globales.

En el ejemplo se utiliza la clase **FlxG** para poder acceder al objeto *keys*, el cual es de tipo **FlxKeyboard** y permite acceder al sistema de entrada por teclado. El atributo *pressed* de dicho objeto corresponde al tipo **FlxKeyList**, el cual consiste en una lista de valores que permite conocer si una tecla dada está o no presionada en un momento determinado.

```
import flixel.FlxState;
import flixel.FlxSprite;
import flixel.FlxG;

class PlayState extends FlxState
{
    private var sprite1: FlxSprite;

    override public function create():Void
    {
        super.create();
        sprite1 = new FlxSprite(100, 100,
"assets/images/ImpGuy_0.jpg");
        add(sprite1);
    }

    override public function update():Void
```

```
    {
        super.update();
        if(FlxG.keys.pressed.A)
        {
            sprite1.velocity.x = -200;
        }
        else if(FlxG.keys.pressed.D)
        {
            sprite1.velocity.x = 200;
        }
        else
        {
            sprite1.velocity.x = 0;
        }
    }
}
```

Fig. 1: Contenido del archivo PlayState.hx del ejemplo analizado

En el cuerpo del método *update()* se ha utilizado el atributo *keys* para mover el personaje hacia la derecha o a la izquierda mediante las teclas *A* y *D*: en caso de estar presionada la primera el personaje tendrá una velocidad negativa, mientras que si está presionada la segunda, indicará un movimiento hacia la derecha. En caso de que ninguna de estas dos condiciones se cumpla, la velocidad horizontal del sprite será nula.

Para conocer o modificar la velocidad del sprite, se puede operar directamente sobre el atributo *velocity* del mismo, el cual es de tipo **FlxPoint**. Éste está, a si vez, compuesto por dos atributos reales, *x* e *y*, que representan las componentes de un vector o punto en el plano, y que en éste caso son utilizados para representar la velocidad del sprite.

Es necesario destacar la llamada al método *update()* definido en la clase padre (invocada al principio de la función mediante la palabra reservada **super**. De no incluirse dicha llamada, se omitirá el comportamiento heredado de la clase **FlxState**, el cual consiste (entre otras cosas) en actualizar la posición de cada sprite en base a su velocidad en cada cuadro, y el sprite no se moverá.

En la *Fig. 2* se puede observar una captura de pantalla del ejemplo analizado sobre la que además puede hacerse click para probar el ejemplo en funcionamiento (utilizar las teclas *A* y *D* para mover al sprite).

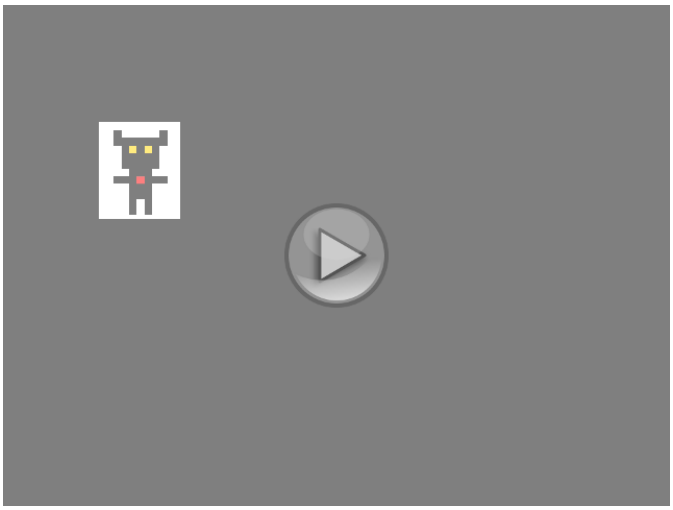

Fig. 2: Captura del ejemplo analizado a lo largo del tutorial

Descargar código del ejemplo

Nuevamente, finalizamos el tutorial con una breve reseña de los conceptos abordados durante el tutorial y recomendamos consultar a la documentación de referencia de las clases involucradas en el mismo:

- **FlxG**
- **FlxKeyboard**
- **FlxKeyList**
- **FlxPoint**
- **FlxSprite**

Resumen:

- La clase **FlxG** es una clase global que facilita el acceso a dispositivos de entrada/salida y a diversos datos y funcionalidades útiles.
- El atributo *keys* de tipo **FlxKeyboard** perteneciente a la clase **FlxG** permite tener acceso al teclado
- El atributo *pressed* del objeto **keys** permite saber si una tecla determinada está siendo presionada
- La clase **FlxPoint** permite representar un punto o una magnitud vectorial 2D
- El atributo *velocity* de un objeto de tipo **FlxSprite** permite conocer y modificar su velocidad
- Al redefinir el método *update()* en una escena es importante invocar al mismo método en la clase padre para que la lógica de actualización heredada siga funcionando

Volver al índice de tutoriales...