PROGRAMACION DE VIDEOJUEGOS III

Desplegando texto en pantalla

En el siguiente tutorial, veremos un simple ejemplo que muestra el uso de la clase **FlxText**, la cual nos permitirá mostrar texto en los juegos que desarrollemos con HaxeFlixel.

Comenzaremos el tutorial mostrando el ejemplo terminado para ilustrar algunas de las cosas que pueden lograrse utilizando dicha clase y luego examinaremos brevemente el código fuente del mismo. El ejemplo puede observarse y probarse en la *Fig. 1*, el mismo consiste en varios textos con distintas características que son mostrados en pantalla.

Fig. 1: Captura del ejemplo analizado a lo largo del tutorial

Descargar código del ejemplo

Tal como adelantamos antes, en HaxeFlixel disponemos de la clase **FlxText** para mostrar texto en pantalla. La misma hereda de **FlxSprite**, por lo cual muchas de las operaciones realizadas anteriormente con sprites son también aplicables a los objetos de tipo **FlxText**, como por ejemplo moverlos o escalarlos.

En la *Fig. 2* se muestra el código fuente del ejemplo de la *Fig. 1*, el cual analizaremos brevemente a continuación.

Como puede verse en las primeras líneas del código, al importar la clase **FlxText** se debe tener en cuenta que la misma pertenece al subpaquete *text* dentro del paquete *flixel*.

En el constructor de la clase **PlayState** se inicializan cuatro objetos de tipo **FlxText**, que son atributos de la escena, y se agregan a la misma tal como se ha hecho con sprites en los ejemplos anteriores.

```
import flixel.FlxState;
import flixel.FlxG;
import flixel.text.FlxText;

class PlayState extends FlxState
{
    private var text1: FlxText;
    private var text2: FlxText;
    private var text3: FlxText;
    private var text4: FlxText;
    private var elapsedTime: Float = 0;

    override public function create():Void
    {
        super.create();

        text1 = new FlxText(200, 200);
        text1.size = 20;
        add(text1);

        text2 = new FlxText(0, 20, 320, "texto
centrado");
        text2.alignment = "center";
        add(text2);

        text3 = new FlxText(0, 60, 50, "Éste
texto está alineado a la derecha...");
        text3.alignment = "right";
        text3.borderSize = 0.5;
        text3.borderColor = 0xffff00ff;
        text3.borderStyle =
FlxText.BORDER_OUTLINE;
        add(text3);

        text4 = new FlxText(0, 200);
        text4.font =
"assets/data/RiseStarHandRegular.ttf";
        text4.size = 20;
        add(text4);
    }
```

```
    override public function update(): Void
    {
        super.update();
        elapsedTime += FlxG.elapsed;
        text1.text = "hola";
        switch(Std.int(elapsedTime)%3)
        {
            case 0:
                text1.color = 0xffff0000;
                text1.text = "ROJO";
            case 1:
                text1.color = 0xff00ff00;
                text1.text = "VERDE";
            case 2:
                text1.color = 0xff0000ff;
                text1.text = "AZUL";
        }
        text1.y = 100 + 20 * Math.sin(3 *
elapsedTime);
        text4.text = "Tiempo transcurrido: " +
Std.int(elapsedTime*100)/100;
    }
}
```

Fig. 2: Contenido del archivo PlayState.hx del ejemplo analizado

El constructor de la clase **FlxState** recibe como parámetros: dos valores reales con la posición inicial del objeto, un real adicional con el ancho del campo de texto (cuyo significado explicaremos a continuación) y el texto que será mostrado inicialmente.

El ancho de campo de un objeto de tipo **FlxText** es un valor importante en diversas situaciones, por ejemplo en caso de que se desee especificar una alineación para el texto. En la *Fig. 2* se muestra una captura de pantalla del ejemplo con los AABB de cada texto dibujados. En caso de especificar una alineación centrada para el texto, deberá tenerse en cuenta que el centrado se realizará respecto al AABB del texto que, en el caso del objeto *text2*, es igual al ancho de la vista. De manera similar, puede observarse que el texto del objeto *text3* es cortado automáticamente en varias líneas para que su tamaño no exceda el ancho especificado. De no especificar un ancho de campo para el texto, el mismo será calculado automáticamente. Además de especificarlo en el constructor, se podrá modificar dicho calor mediante el atributo *fieldWidth*.

Fig. 3: Captura de pantalla del ejemplo mostrando los AABB de cada texto

Como se puede ver en el código del ejemplo, existen muchos atributos que permiten modificar la forma en que el texto se visualiza. El atributo *size* permite especificar el tamaño del texto. El atributo *alignment* es de tipo **String** y especifica la alineación horizontal del texto respecto a su AABB, sólo puede tomar tres valores posibles: "center", "left" y "right". Los atributos *borderColor* y *borderSize* definen el color y tamaño del borde, dichos atributos solo tendrán efectos para ciertos valores del atributo *borderStyle*, el cual define que tipo de borde se dibujará. El atributo *font* especifica el archivo de fuente que el texto utilizará (se utiliza una fuente embebida en caso de no especificar ninguno). El atributo *color*, al igual que con un spite, permite modificar el color del texto.

Finalmente, el atributo *text* permite modificar el texto que se mostrará. Como se puede apreciar en el código del ejemplo, en haXe, puede sumarse un valor numérico a una cadena mediante el operador *"+"*. Por otro lado, el método *int()* de la clase **Std** permite convertir un valor real a entero.

En éste tutorial, se han demostrado brevemente algunas de las capacidades de la clase **FlxText** para mostrar texto. Como siempre, para una explicación más detallada de los métodos y atributos de ésta clase, sugerimos revisar la documentacion de referencia de HaxeFlixel.

Asimismo, dado que en éste ejemplo se ha introducido brevemente la clase **Std** de haXe, recomendamos aprovechar a hechar una mirada a la página de documentación correspondiente. La clase **Std** posee algunas funciones que permiten convertir entre los distintos tipos primitivos de haXe.

Resumen:

- En HaxeFlixel, la clase **FlxText** nos permite reperesentar texto y opera de manera similar a la clase **FlxSprite**
- El atributo *fieldWidth* de la clase **FlxText** representa el ancho del AABB que contiene al texto y es un valor útil a la hora de alinear o ajustar el texto
- El atributo *text* de la clase **FlxText** permite modificar el texto que se muestra

Volver al índice de tutoriales...