Programación 2D

Práctica 3: Sprites y animación

En prácticas anteriores hemos visto que cuando queremos dibujar elementos, debemos almacenar la posición, color, modo de mezclado a emplear y otras propiedades nosotros mismos. Esto nos obliga a definir un número elevado de estructuras de datos y gestionar manualmente mucha información.

En un juego, normalmente se renderizan multitud de sprites, y es por ello que el motor suele tener soporte para almacenar la información de este tipo de elementos sin que tengamos que crear estructuras adicionales.

Clase Sprite

Añadiremos esta clase al motor para incluir toda la información mencionada sobre un sprite, además de gestionar su animación. Tendrá la siguiente interfaz, y añadiremos las variables miembro que sea necesario:

- // Tipo de la función callback

typedef void (* CallbackFunc)(Sprite&, float);

CallbackFunc define el tipo para una función callback que será llamada automáticamente por el sprite si está definida con setCallback. La función no tiene valor de retorno, y tiene como parámetros una referencia al sprite y el deltaTime.

- // Indicamos el número de frames en horizontal y vertical

  // que tendrá la imagen del sprite

Sprite(const ltex_t* tex, int hframes = 1, int vframes = 1);

- // Establecemos puntero a la función callback

void setCallback(CallbackFunc func);

- // Puntero genérico a datos (normalmente introducimos aquí los datos

  // del sprite que se van a utilizar en la función callback) void* getUserData();

void setUserData(void* data);

- const ltex_t* getTexture() const;
- void setTexture(const ltex_t* tex, int hframes = 1, int vframes = 1);

- lblend_t getBlend() const;
- void setBlend(lblend_t mode);

- float getRed() const;
- float getGreen() const;
- float getBlue() const;
- float getAlpha() const;
- void setColor(float r, float g, float b, float a);

- const Vec2& getPosition() const;
- void setPosition(const Vec2& pos);

- float getAngle() const;
- void setAngle(float angle);

El ángulo del sprite indica una rotación en sentido natural (antihorario).

- const Vec2& getScale() const;
- void setScale(const Vec2& scale);

- // Tamaño de un frame multiplicado por la escala

Vec2 getSize() const;

- // Este valor se pasa a ltex_drawrotsized en el pintado

  // para indicar el pivote de rotación

const Vec2& getPivot() const;

void setPivot(const Vec2& pivot);

- int getHframes() const;
- int getVframes() const;

- // Veces por segundo que se cambia el frame de animación

int getFps() const;

void setFps(int fps);

- // Frame actual de animación

float getCurrentFrame() const;

void setCurrentFrame(int frame);

- void update(float deltaTime);

En el método update, si el puntero a la función callback no es nulo, debemos llamar a la función. Además, debemos actualizar el frame según los fps de animación. Si nos pasamos del final o del principio de la animación (ya que los fps pueden ser negativos para reproducir la animación al revés), debemos volver al principio o al final de la misma, respectivamente.

- void draw() const;

En el método draw, debemos calcular las coordenadas UV de textura según el frame de animación actual, establecer todas las propiedades de pintado almacenadas en el objeto (modo de mezclado y color de tintado), y dibujar la textura utilizando ltex_drawrotsized, calculando el tamaño según la escala y rotando adecuadamente la textura.

Programa principal

Al comienzo del programa, cargaremos la imagen "data/bee_anim.png", y generaremos un sprite con dicha imagen, utilizando 8 frames de animación en horizontal y 1 en vertical. Estableceremos una velocidad de animación de 8 frames por segundo.

En el bucle principal, el sprite se debe ir acercando desde su posición actual hacia las coordenadas del ratón, a una velocidad de 128 puntos por segundo. Se debe llamar a los métodos update y draw del sprite en el bucle principal para que se realice la actualización de animación y el pintado. Además, debemos rotar el sprite de la siguiente forma:

- Si el sprite se está moviendo a la derecha, debemos rotar desde el ángulo actual hasta -15º a una tasa de 32 grados por segundo.
- Si el sprite se está moviendo a la izquierda, debemos rotar desde el ángulo actual hasta 15º a una tasa de 32 grados por segundo.
- Si el sprite no se está moviendo, debemos rotar desde el ángulo actual hasta 0º a una tasa de 32 grados por segundo.
    - Añadir además algún otro objeto que esté libre por la ventana que reproduzca otra animación a vuestro gusto (recomendable que sea estático, pero puede comportarse parecido a la abeja)