MulticocoSDL

José Ladislao Lainez Ortega y José Molina Colmenero

15 de mayo de 2013

Índice

1. Introducción

MulticocoSDL es un juego arcade que emula al famoso Pacman realizado como proyecto para la asignatura Sistemas Multimedia del Grado de Ingeniería Informática de la Universidad de Jaén.

El objetivo de MulticocoSDL es alcanzar varias areas multimedia en un mismo programa haciendo uso de los tres elementos principales del software multimedia:

Emplazamiento espacial
: Los distintos elementos visuales (enemigos, escenario, pacman) son dibujados sobre un canvas en posiciones especficas y adems se puede mover por l.

Control temporal
: El jugador puede moverse por el escenario a una determinada velocidad as como los enemigos, estando todos ellos animados.

Interacción
: Mediante el teclado el jugador puede dar órdenes a pacman de forma que este se mueva en la dirección que el jugador le indica.

El código se encuentra alojado en GitHub en el siguiente enlace y contiene un archivo de proyecto de XCode para Mac OS X, si bien el código es portable a Windows y Linux.

`https://github.com/L4D15/MulticocoSDL`

2. Bibliotecas

Se ha hecho uso de la biblioteca Simple DirecMedia Layer así como de algunos submódulos de esta para trabajar con el renderizado de imágenes, texto y reproducción de audio. A continuación explicamos qué tareas ha realizado cada una.

Para más informacin sobre Simple DirecMedia Layer:

`http://www.libsdl.org/`

2.1. SDL

Biblioteca con las operaciones básicas para crear una ventana y dibujar en ella. Algunas de las utilidades más usadas han sido:

SDL_Rec
: Estructura para definir un rectángulo. Usado a la hora de recorta un área de un SpriteSheet y dibujar en un canvas.

SDL_Surface
: Superficie o canvas sobre el que dibujar, usado no solo en la ventana principal como contexto de renderizado, sino también para almacenar en la memoria de la tarjeta gráfica los distintos Sprites a renderizar.

SDL_BlitSurface(SDL_Surface*,SDL_Rec*,SDL_Surface*,SDL_Rec*)
: Mediante esta función se puede dibujar un área seleccionada de un canvas origen en un área de un canvas de destino. Se ha usado tanto para dibujar en la ventana principal como para separar los distintos Sprites del SpriteSheet.

SDL_LoadBMP(const char*)
: Como bien indica su nombre carga una imagen en formato BMP, la guarda en memoria gráfica y devuelve un puntero a una SDL_Surface, de modo que podamos usar la imagen cargada.

Esta biblioteca puede descargase desde el siguiente enlace:

`http://www.libsdl.org/download-1.2.php`

2.2. SDL_Mixer

Biblioteca modular de SDL que facilita el trabajo con la reproducción de sonido y música. Los elementos más destacados de esta biblioteca son:

Mix_Chunk
: Contenedor de sonido, igual que SDL_Surface lo era de imágenes.

Mix_Music
: Contenedor de sonido específico para música.

Mix_LoadWAV(const char*)
: Carga un archivo en formato WAV y devuelve un puntero a Mix_Chunk para poder trabajar con el audio del archivo.

Mix_LoadMUS(const char*)
: Carga un archivo en formato WAV, OGG, MP3 o FLAC devolviendo un puntero a Mix_Music. Esta función es específica para cargar música ya que SDL_Mixer trabaja de forma distinta los sonidos y la música.

La biblioteca se puede descargar desde:

`http://www.libsdl.org/projects/SDL_mixer/`

2.3. SDL_TTF

A la hora de mostrar la puntuación del jugador necesitábamos mostrar texto por pantalla, por lo que recurrimos a esta biblioteca (otro módulo del proyecto SDL) específica para mostrar texto por pantalla. Una curiosidad sobre esta biblioteca es que hace uso de fuentes TTF en lugar de recurrir a fuentes en archivos bitmap como sucede, por ejemplo, al renderizar texto en OpenGL.

De esta biblioteca hemos usado:

TTF_Font
: Contenedor para la información de la fuente a usar a la hora de renderizar el texto.

TTF_OpenFont(const char*, int)
: Carga una fuente desde el archivo en formato TTF especificado y usando el tamao indicado.

TTF_RenderText_Solid(TTF_Font,const char*,SDL_Color)
: Crea una superficie sobre la que dibuja el texto pasado usando la fuente indicada y el color deseado. Una vez tengamos esa superficie habrá que dibujarla usando el método que se mostró antes con SDL_BlitSurface(...).

3. Imagen

3.1. Window

3.2. Sprite

3.3. SpriteSheet

4. Audio

4.1. Sound

4.2. Music

5. Lógica

5.1. Vector2D

5.2. CollisionBox

6. Juego

6.1. Inicialización

6.2. Visualización

6.3. Eventos