Introdução à SDL

por Bruno Bottino Ferreira
tinnus@gmail.com

O que é SDL?

- Simple Directmedia Layer
- Biblioteca multimídia voltada para jogos
- Vídeo
- Som
- Interface com o usuário (teclado, mouse, joystick)
- CD-Áudio
- Threading
- Controle de tempo

Por que SDL?

- Abstração do hardware

Programador

Por que SDL?

- Portabilidade

Requisitos

- Conhecimento de programação em C/C++
- Ambiente de programação C/C++
- Instalar a biblioteca no ambiente de programação
- Preferencialmente hardware de som, mouse, joystick e CD-ROM (para testes)

Instalando

- http://www.libsdl.org
- Download -> SDL x.x (mais recente)
- Development Libraries (para sua plataforma)

  ou
- Source code (código-fonte) (deve ser compilado)

Instalando no MinGW

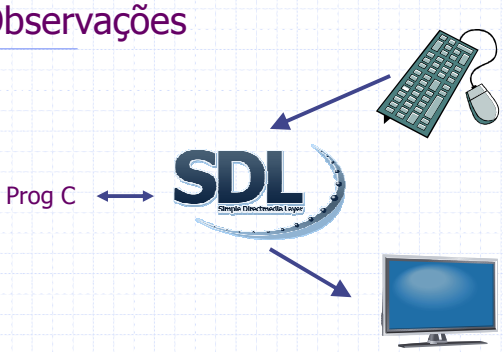
- Baixar o arquivo .tar.gz correspondente ao MinGW
- Copiar as pastas "Include" e "Lib" para a pasta do MinGW
- Para compilar:

```
gcc prog.c –o prog.exe -Wall
–lmingw32 –lSDLmain -lSDL
```

Observações

- Cuidados no Windows
  - A SDL redireciona `stdout` e `stderr` para os arquivos `stdout.txt` e `stderr.txt`
  - Tentar ler de `stdin` geralmente causa problemas

Observações



Conceitos básicos

- Uma *superfície* é uma matriz bidimensional de pontos, onde cada ponto representa uma cor composta por três componentes: R, G e B (vermelho, verde e azul)
- Um *PixelFormat* é uma estrutura que define o formato em que as cores são armazenadas em um pixel

Conceitos básicos

- SDL trabalha baseado em eventos.
- Normalmente o programa *não controla* o seu fluxo.
- Ele fica em um laço esperando eventos acontecerem.
- Um *evento* é uma mensagem enviada do sistema operacional ao seu programa, como o pressionamento de uma tecla, movimento do mouse ou uma mensagem de término.

Programa básico em SDL

- Incluir a biblioteca
  - `#include <SDL.h>`
- Declarar a função **`main`** como
  - `int main (int argc, char** argv)`
- Declarar variáveis de controle
  - `SDL_Surface* screen;`
    `SDL_Event event;`
    `int stop = 0;`

Programa básico em SDL

- Inicializar
  - `SDL_Init(flags);`
- "flags" é um ou-binário dos valores:
  - `SDL_INIT_VIDEO`
  - `SDL_INIT_AUDIO`
  - `SDL_INIT_JOYSTICK`
  - `SDL_INIT_CDROM`
  - `SDL_INIT_TIMER`
  - `SDL_INIT_EVERYTHING`

Programa básico em SDL

- Criar a janela
  - `screen = SDL_SetVideoMode(width, height, depth, flags);`
- "flags" é um ou-binário dos valores:
  - `SDL_SWSURFACE`
  - `SDL_HWSURFACE`
  - `SDL_ASYNCBLIT`
  - `SDL_ANYFORMAT`
  - `SDL_HWPALETTE`
  - `SDL_DOUBLEBUF`
  - `SDL_FULLSCREEN`
  - `SDL_OPENGL`
  - `SDL_OPENGLBLIT`

Programa básico em SDL

- Loop principal

```
while(!stop)
{
  while(SDL_PollEvent(&event))
  {
    if(event.type == SDL_QUIT)
    {
      stop = 1;
    }
  }
  SDL_Flip(screen);
}
```

Programa básico em SDL

- Encerrar

```
SDL_Quit();
return 0;
```

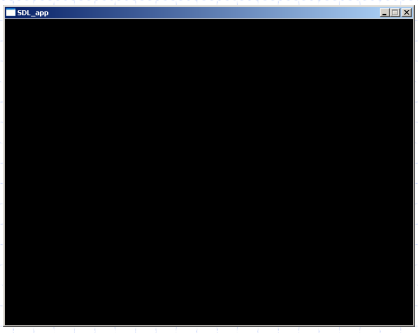
Programa básico em SDL

```
#include <SDL.h>
//====>  ex0.c
int main(int argc, char** argv) {
    SDL_Surface* screen;
    SDL_Event event;
    int quit = 0;

    SDL_Init(SDL_INIT_VIDEO);
    screen = SDL_SetVideoMode(640, 480, 8, 0);

    while(!quit) {
        while(SDL_PollEvent(&event))
                if(event.type == SDL_QUIT)
                        quit = 1;
        SDL_Flip(screen);
    }
    SDL_Quit();
    return 0;
}
```

Programa básico em SDL



Superfícies

- SDL usa estruturas chamadas superfícies (do tipo **SDL_Surface**) para representar dados gráficos.
- Uma superfície é somente um bloco de memória para armazenar uma região retangular de pixels.
- Cada superfície tem uma largura, altura e um formato específico para armazenar os pixels.

Superfícies

- SDL carrega arquivos de imagens diretamente nas estruturas de superficie.
- A tela (screen) também é uma superfície, embora especial.
- Superfícies podem ser copiadas para cima de outras em uma operação chamada blit (block image transfer).

Superfícies

- Declarando uma superfície
  - `SDL_Surface* my_surf;`
- Campos importantes
  - `format` – *PixelFormat* dos pixels
  - `w`, `h` – Largura e Altura
  - `pitch` – largura de uma linha (em bytes)
  - `pixels` – ponteiro para os dados dos pixels

Superfícies

- Criando uma superfície vazia
  - `my_surf = SDL_CreateRGBSurface(flags, width, height, depth, Rmask, Gmask, Bmask, Amask);`
- "flags" é um ou-binário dos valores:
  - `SDL_SWSURFACE`
  - `SDL_HWSURFACE`
  - `SDL_SRCCOLORKEY`
  - `SDL_SRCALPHA`

Superfícies

- Uma superfície com o PixelFormat da tela
  - `my_surf = SDL_CreateRGBSurface(flags, width, height, screen->format->BitsPerPixel, screen->format->Rmask, screen->format->Gmask, screen->format->Bmask, screen->format->Amask);`

Superfícies

- Criando uma superfície a partir de um arquivo
  - BMP
    - `my_surf = SDL_LoadBMP("file.bmp");`
  - BMP, PNM, XPM, LBM, PCX, GIF, JPEG, PNG, TGA, TIFF (usando a biblioteca SDL_image)

```
#include <SDL_image.h>

/* ... */

my_surf = IMG_Load("file.xxx");
```

  - http://www.libsdl.org/projects/SDL_image/

Blits

- Um *blit* é uma cópia de parte da imagem de uma superfície para outra
  - `src` é a superfície de origem
  - `dst` é a superfície de destino
  - `srcrect` é a área de origem
  - `dstrect` é a área de destino
  - Se `srcrect` ou `dstrect` forem NULL, significa toda a área da superfície
- `SDL_BlitSurface(src, srcrect, dst, dstrect);`
- Arquivo: `ex1.c`

Atualizando a tela

- Para atualizar parte da tela
  - `SDL_UpdateRect(screen, x, y, width, height);`
- Para atualizar toda a tela
  - Sem double buffering (`SDL_DOUBLEBUF`)
    - `SDL_UpdateRect(screen, 0, 0, screen->w, screen->h);`
  - Com ou sem double buffering
    - `SDL_Flip();`

Transparência

- Tornar uma cor transparente
  - `SDL_SetColorKey(surface, SDL_SRCCOLORKEY, SDL_MapRGB(surface->format, r, g, b));`
- Desligar a cor transparente
  - `SDL_SetColorKey(surface, 0, 0);`
- Dar um valor de transparência geral
  - `SDL_SetAlpha(surface, SDL_SRCALPHA, val);`
  - `val`: 0 (transparente) a 255 (opaco)
- Desligar a transparência geral
  - `SDL_SetAlpha(surface, 0, 0);`
- Arquivo: `ex2v2.c`

Cursor do Mouse

- Mostrar/esconder o cursor
  - `SDL_ShowCursor(toggle);`
    - SDL_ENABLE mostra o cursor
    - SDL_DISABLE esconde o cursor
    - SDL_QUERY retorna o estado atual
- Mover o cursor
  - `SDL_WarpMouse(x, y);`

Eventos

- Tipos de eventos
  - SDL_ACTIVEEVENT
  - SDL_KEYDOWN / SDL_KEYUP
  - SDL_MOUSEMOTION
  - SDL_MOUSEBUTTONDOWN / SDL_MOUSEBUTTONUP
  - SDL_JOYAXISMOTION
  - SDL_JOYBALLMOTION
  - SDL_JOYHATMOTION
  - SDL_JOYBUTTONDOWN / SDL_JOYBUTTONUP
  - SDL_QUIT
  - SDL_SYSWMEVENT
  - SDL_VIDEORESIZE
  - SDL_VIDEOEXPOSE
  - SDL_USEREVENT

Eventos

- Lendo eventos
  - `SDL_Event event;`

```
while(SDL_PollEvent(&event))
{
  switch(event.type)
  {
    case tipo1: /* ... */ break;
    case tipo2: /* ... */ break;
    /* ... */
    case tipon: /* ... */ break;
  }
}
```

Eventos

- Teclado
  - Dados do evento: `event.key`
    - `type`: `SDL_KEYDOWN` ou `SDL_KEYUP`
    - `state`: `SDL_PRESSED` ou `SDL_RELEASED`
  - Dados da tecla: `event.key.keysym`
    - `sym`: constante em `SDLKey`
    - `unicode`: caracter (formato Unicode/ASCII)
    - `mod`: modificadores (ou-binário de constantes em SDLMod)
- Arquivo: `ex3.c`

Eventos

- Movimentação do mouse
  - Dados do evento: `event.motion`
    - `type`: `SDL_MOUSEMOTION`
    - `state`: estado dos botões
    - `x`, `y`: novas coordenadas do mouse
    - `xrel`, `yrel`: movimento relativo
- Clique do mouse
  - Dados do evento: `event.button`
    - `type`: `SDL_MOUSEBUTTONDOWN` ou `SDL_MOUSEBUTTONUP`
    - `button`:
      - `SDL_BUTTON_LEFT`
      - `SDL_BUTTON_MIDDLE`
      - `SDL_BUTTON_RIGHT`
    - `state`: `SDL_PRESSED` ou `SDL_RELEASED`
- Arquivo: `ex4.c`

Eventos

- Outros eventos
  - Joysticks
  - Redimensionamento da janela
  - Encerramento
  - Eventos gerados pelo usuário

CD-Áudio

- Número de drives de CD
  - `n_drives = SDL_CDNumDrives();`
- Abrir um drive para acesso
  - `SDL_CD* cdrom = SDL_CDOpen(n);`
- Tocar uma trilha
  - `SDL_CDPlayTracks(cdrom, track, 0, 1, 0);`
  - Trilhas começam em zero
- Fechar um drive para acesso
  - `SDL_CDClose(cdrom);`
- Arquivo: `ex5.c`

Som

- Utilizando a biblioteca SDL_mixer
  - http://www.libsdl.org/projects/SDL_mixer/
- Inicializando
  - `SDL_Init( ... | SDL_INIT_AUDIO);`
  - `Mix_OpenAudio(freq, format, channels, bufsize);`
    - `freq`: frequência de saída (ex. 44100Hz)
    - `format`: formato de saída (ex. 16bits)
    - `channels`: 1 (mono) ou 2 (estéreo)
    - `bufsize`: tamanho do buffer (ex. 4096)
  - Tipicamente
    - `Mix_OpenAudio(44100, MIX_DEFAULT_FORMAT, 2, 4096);`

Som

- Carregando um som
  - `Mix_Chunk* sound = Mix_LoadWAV("kaboom.wav");`
- Ajustando o volume de um som
  - `Mix_VolumeChunk(sound, volume);`
  - `0 <= volume <= MIX_MAX_VOLUME`
- Liberando a memória
  - `Mix_FreeChunk(sound);`

Som

- Alocando canais de saída
  - `Mix_AllocateChannels(n_channels);`
- Tocando um som em um canal
  - `Mix_PlayChannel(channel, sound, loops);`
  - `channel`: número do canal
    - `channel = -1`: primeiro disponível
  - `loops`: número de repetições
    - `loops = 0`: tocar uma vez
    - `loops = -1`: tocar para sempre

Som

- Carregando uma música
  - `Mix_Music* music = Mix_LoadMUS("boogie.mp3");`
  - WAVE, MOD, MIDI, OGG, MP3
- Tocando uma música
  - `Mix_PlayMusic(music, loops);`
  - `loops`: número de repetições
    - `loops = 0`: tocar uma vez
    - `loops = -1`: tocar para sempre
- Ajustando o volume da música
  - `Mix_VolumeMusic(sound, volume);`
  - `0 <= volume <= MIX_MAX_VOLUME`

Som

- Encerrando
  - `Mix_CloseAudio();`
- Outras funções
  - Fade in/out
  - Panning
  - Distância
  - Posição (som 2D)
- Arquivo: `ex6.c`

Tempo

- Contagem de tempo
  - `time = SDL_GetTicks();`
  - Retorna o número de ms desde a inicialização
- Criando um temporizador
  - `id = SDL_AddTimer(interval, func, param);`
  - Chama a função `func` a cada `interval` ms passando os parâmetros `interval` e `param`
  - `Uint32 func(Uint32 interval, void *param);`
- Cancelando um temporizador
  - `SDL_RemoveTimer(id);`
- Arquivo: `ex7.c`

Encerrando

- Assuntos não abordados
  - Joysticks
  - Threads
- Onde conseguir ajuda
  - http://www.libsdl.org
    - http://www.libsdl.org/cgi/docwiki.cgi/
    - http://news.gmane.org/thread.php?group=gmane.comp.lib.sdl
  - http://gpwiki.org/index.php/C:SDL_tutorials

O Fim