Simple Direct Layer

www.libsdl.org

1.1. Obecné

- `#include "SDL.h"`
- Datové typy

SDL	C ekvivalent
SDL_bool	int
Uint8	unsigned char
Uint16	unsigned short
Uint32	unsigned int
Uint64	unsigned long long
Sint8	signed char
Sint16	signed short
Sint32	singed int
Sint64	signed long long

- SDL_bool = { SDL_TRUE / SDL_FALSE
- **SDL_Init()** – Inicializace

```
int SDL_Init(Uint32 flags);
```

Flagy (zadávat s bitovým OR):

  - SDL_INIT_TIMER, SDL_INIT_AUDIO, SDL_INIT_VIDEO, SDL_INIT_CDROM, SDL_INIT_JOYSTICK – zapnutí subsystémů pro časovač, audio, video, CD-ROM, joystick,
  - SDL_INIT_EVERYTHING – všechny předchozí dohromady,
  - SDL_INIT_NOPARACHUTE – ochrana před zachycením atálních signálů,
  - SDL_INIT_EVENTTHREAD – správa událostí ve zvlášním vláknu.

- **SDL_InitSubSystem(), SDL_QuitSubSystem()** – dodatečná inic. subsystému nebo jeho vypnutí

```
int SDL_InitSubSystem(Uint32 flags);
void SDL_QuitSubSystem(Uint32 flags);
```

- **SDL_WasInit()** – kontrola jestli byl subsystém inicializován

```
int SDL_WasInit(Uint32 flags);
```

- **SDL_GetError()** – vrátí řetězec s popisem chyby

```
char *SDL_GetError(void);
```

- **SDL_Quit()** – ukončení běhu SDL

```
void SDL_Quit(void);
```

1.2. Video

- Struktura **SDL_Surface** – definuje parametry okna video paměti

```
typedef struct SDL_Surface {
  Uint32 flags;                /* Read-only */
  /* Formát pixelů */
  SDL_PixelFormat *format;     /* Read-only */
  /* w - šířka, h - výška */
  int w, h;                    /* Read-only */
  /* délka obr. řádku v bytech */
  Uint16 pitch;                /* Read-only */
  /* ukatel na obr. data */
  void *pixels;                /* Read-write */
  /* ořezový obdélník */
  SDL_Rect clip_rect;          /* Read-only */
  int refcount;                /* Read-mostly */
} SDL_Surface;
```

- Struktura **SDL_Rect** – definuje parametry obdélníkové oblasti

```
typedef struct{
  Sint16 x, y; /* levý horní roh */
  Uint16 w, h; /* rozměry šířka x výška */
} SDL_Rect;
```

- Struktura **SDL_PixelFormat** – uložené parametry surface

```
typedef struct {
  /* ukazatel na 8bit. paletu */
  SDL_Palette *palette;
  /* počet bitů na pixel (8, 16, 24, 32) */
  Uint8  BitsPerPixel;
  /* počt bytů na pixel */
  Uint8  BytesPerPixel;
  /* přesnost barevných kanálů */
  Uint8  Rloss, Gloss, Bloss, Aloss;
  /* pousny uložení bar. kanálů */
  Uint8  Rshift, Gshift, Bshift, Ashift;
  /* maska barevných kanálů */
  Uint32 Rmask, Gmask, Bmask, Amask;
  Uint32 colorkey;
  /* celková hodnota alfa kanálu */
  Uint8  alpha;
} SDL_PixelFormat;
```

- **SDL_SetVideoMode()** – nastavení paremetrů video módu

```
SDL_Surface *SDL_SetVideoMode(int width,
int height, int bitsperpixel, Uint32 flags);
```

Flagy (zadávat s bitovým OR):

  - SDL_SWSURFACE – surface v hlavní paměti,
  - SDL_HWSURFACE – surface ve video paměti,
  - SDL_DOUBLEBUF – hardwarový double-buffering,
  - SDL_FULLSCREEN – fullscreen režim,
  - SDL_OPENGL – vykreslování přes OpenGL,
  - SDL_RESIZABLE – okno s proměnými rozměry,
  - SDL_ASYNCBLIT, SDL_ANYFORMAT, SDL_HWPALETTE, SDL_OPENGLBLIT, SDL_RESIZABLE, SDL_NOFRAME.

- **SDL_CreateRGBSurface()** – vytvoří prázdný surface

```
SDL_Surface *SDL_CreateRGBSurface(Uint32 flags,
  int width, int height, int bitsPerPixel,
  Uint32 Rmask, Uint32 Gmask, Uint32 Bmask,
  Uint32 Amask);
```

Flagy (zadávat s bitovým OR):

  - SDL_SWSURFACE, SDL_HWSURFACE
  - SDL_SRCCOLORKEY – klíčování barevných kanalů,
  - SDL_SRCALPHA – zapíná míchání alpha-kanálu.

- Pořadí bitové masky

SDL_BYTEORDER = { SDL_BIG_ENDIAN $RGBA$ / SDL_LIL_ENDIAN $ABGR$

- **SDL_CreateRGBSurfaceFrom()** – vytvořit surface s obrazovými daty.

```
SDL_Surface *SDL_CreateRGBSurfaceFrom(void *pixels, ...
```

- **SDL_SetClipRect()** – nastaví ořezový obdélník surface

```
void SDL_SetClipRect(SDL_Surface *surface,
                     SDL_Rect *rect);
```

- **SDL_MapRGB()** – mapování RGB(A) kanálů do formátu pixelu

```
Uint32 SDL_MapRGB(SDL_PixelFormat *fmt,
                  Uint8 r, Uint8 g, Uint8 b);
Uint32 SDL_MapRGBA(SDL_PixelFormat *fmt,
         Uint8 r, Uint8 g, Uint8 b, Uint8 a);
```

- **SDL_GetRGB()** – vrátí RGB(A) kanály z pixelu

```
void SDL_GetRGB(Uint32 pixel,
         SDL_PixelFormat *fmt,
         Uint8 *r, Uint8 *g, Uint8 *b);

void SDL_GetRGBA(Uint32 pixel,
         SDL_PixelFormat *fmt,
         Uint8 *r, Uint8 *g, Uint8 *b, Uint8 *a);
```

- **Kreslení pixelů** pomocí zadávání adresy uvnitř surface

  $$\text{Adresa} = \text{ukazatel} + Y \times \text{velikost řádku} + X \times \text{pixel}$$

  - ukazatel na počátek surface: `surface->pixels`
  - velikost obrazového řádku v bytech: `surface->pitch`
  - velikost obr. bodu v paměti: `surface->format->BytesPerPixel`

- **SDL_FillRect()** – vyplnění oblasti barvou, `dstrect` omezí velikost, při `NULL` celá plocha

```
int SDL_FillRect(SDL_Surface *dst,
             SDL_Rect *dstrect, Uint32 color);
```

- **SDL_SetAlpha** – nastavení průhlednosti, úplně průhledný 0, neprůhledný 255

```
int SDL_SetAlpha(SDL_Surface *surface,
         Uint32 flags, Uint8 alpha);
```

- **SDL_BlitSurface()** – rychlý přesun dat mezi surfacy

```
int SDL_BlitSurface(SDL_Surface *src,
                    SDL_Rect *srcrect,
                    SDL_Surface *dst,
SDL_Rect *dstrect);
```

- **SDL_LoadBMP(), SDL_SaveBMP()** – načtení a uložení obrazu

```
SDL_Surface *SDL_LoadBMP(const char *file);
int SDL_SaveBMP(SDL_Surface *surface,
                const char *file);
```

- **SDL_LockSurface(), SDL_UnlockSurface()** – uzamčení surface pro přímý přístup, udemčení surface. Uzamčení se provádí před zápisem obrazových dat.

```
int SDL_LockSurface(SDL_Surface *surface);
void SDL_UnlockSurface(SDL_Surface *surface);
```

- **SDL_UpdateRect** – obnoví oblast obrazovky, pokud jsou parametry 0 celou obrazovku

```
void SDL_UpdateRect(SDL_Surface *screen, Sint32 x,
                    Sint32 y, Sint32 w, Sint32 h);
```

- **SDL_FreeSurface** – uvolní surface z paměti

```
SDL_FreeSurface(SDL_Surface *surface);
```

1.3. Komunikace se správcem oken

- **SDL_WM_SetCaption** nastavení titulku okna

```
void SDL_WM_SetCaption(const char *title,
                       const char *icon);
```

- **SDL_WM_ToggleFullScreen** – přepínání mezi oknem a celou obrazovkou (za běhu pouze pod X11)

```
int SDL_WM_ToggleFullScreen(SDL_Surface *surface);
```

- **SDL_GrabMode** – chování kurzoru myši;

```
SDL_GrabMode SDL_WM_GrabInput(SDL_GrabMode mode);
```

Mód: při `SDL_GRAB_ON` kurzor pohlcen v okně, vypnout `SDL_GRAB_OFF`, zjistit stav pomocí `SDL_GRAB_QUERY`

1.4. Správa událostí

- Union SDL_Event pro zpracování událostí vyvolaných uživatelem, obsahuje struktury se stavem jednotlivých zařízení
  - Klávesnice

```
typedef struct{
  Uint8 type; /* SDL_KEYDOWN, SDL_KEYUP */
  Uint8 state; /* SDL_PRESSED, SDL_RELEASED */
  SDL_keysym keysym; /* klávesa */
} SDL_KeyboardEvent;
```

  - Pohyb kurzoru myši

```
typedef struct{
  Uint8 type; /* SDL_MOUSEMOTION */
  Uint8 state; /* stav tlačítek */
  Uint16 x, y; /* pozice */
  Sint16 xrel, yrel; /* relativní poz. */
} SDL_MouseMotionEvent;
```

  - Stav tlačítek myši

```
typedef struct{
  /* SDL_MOUSEBUTTONDOWN, SDL_MOUSEBUTTONUP */
  Uint8 type;
  Uint8 button; /* které tlačítko ? */
  Uint8 state; /* SDL_PRESSED,  SDL_RELEASED */
  Uint16 x, y; /* souřadnice při změně stavu */
} SDL_MouseButtonEvent;
```

    Tlačítko (`SDL_BUTTON_xxx`): `LEFT, MIDDLE, RIGHT, WHEELUP, WHEELDOWN`

  - Ukončení správcem oken, typ `SDL_QUIT`

```
typedef struct{ Uint8 type } SDL_QuitEvent;
```

- **SDL_WaitEvent()** – libovolně dlouhé čekání dokud nenastane nějaká událost

```
int SDL_WaitEvent(SDL_Event *event);
```

- **SDL_PollEvent()** – vrátí posledná událost ve frontě, nečeká na událost v případě prázdné fronty

```
int SDL_PollEvent(SDL_Event *event);
```

- **SDL_PumpEvents()** – shromažďuje události ze zařícení a tlačí je do fronty

```
void SDL_PumpEvents(void);
```

- **SDL_PushEvent()** – vložení události do fronty

```
int SDL_PushEvent(SDL_Event *event);
```

2. Kompilace v GNU/Linux

- Nastavení voleb kompilátoru

```
gcc ‘sdl-config --cflags‘ -c program.c
```

- Nastavení voleb linkeru

```
gcc ‘sdl-config --libs‘ -o program program.o
```