



SDL In Practice

Keep It Small and Simple

Draw BMP Image

Draw Image At A Position

Draw A Part Of Image

Move It Around Screen

Draw BMP Image

SDL_Surface is a structure that contains a collection of pixels.

SDL_Blitsurface is used to perform a surface copy to a destination surface.

To draw an image on a window, we load image to a surface and perform it to window surface.

draw_image_bmp_01.cpp

```
#include <SDL2/SDL.h>

int main()
{
    //init
    SDL_Init(SDL_INIT_VIDEO);
    SDL_Window* pWindow = SDL_CreateWindow("BMP", 100, 100, 400, 400, 0);
    SDL_Surface* pWindowSurface = SDL_GetWindowSurface(pWindow);

    //load image
    SDL_Surface* pImageSurface = SDL_LoadBMP("assassin.bmp");

    //draw background
    SDL_FillRect(pWindowSurface, NULL, 0x3b5999);

    //draw image
    SDL_Blitsurface(pImageSurface, NULL, pWindowSurface, NULL);

    SDL_UpdateWindowSurface(pWindow);
    SDL_Delay(5000);

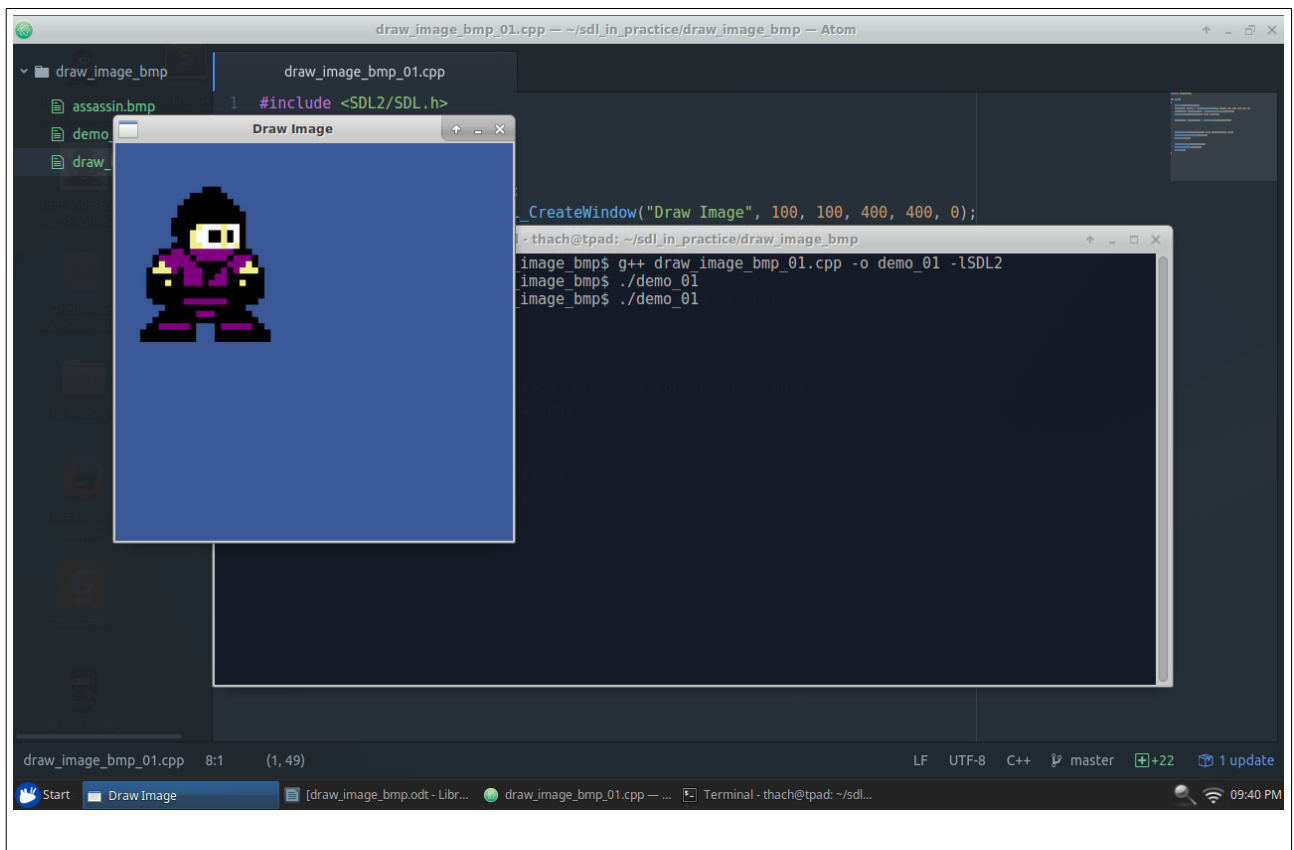
    //destroy
    SDL_FreeSurface(pImageSurface);
    SDL_DestroyWindow(pWindow);
    SDL_Quit();
}
```

Compile

```
g++ draw_image_bmp_01.cpp -o demo_01 -lSDL2
```

Run

```
./demo_01
```



Draw Image In Action

Draw Image At Another Position

Declare a variable store position on screen and pass it to `SDL_BlitSurface()` function.

```
SDL_Rect dstRect = {x, y, 0, 0};
SDL_BlitSurface(pImageSurface, NULL, pWindowSurface, &dstRect);
```

dstRect: represent the position on screen.

```
draw_image_bmp_02.cpp

#include <SDL2/SDL.h>

int main()
{
    //init
    SDL_Init(SDL_INIT_VIDEO);
    SDL_Window* pWindow = SDL_CreateWindow("Move", 100, 100, 500, 500, 0);
    SDL_Surface* pWindowSurface = SDL_GetWindowSurface(pWindow);
    SDL_FillRect(pWindowSurface, NULL, 0x3b5999);

    //load image
    SDL_Surface* pImageSurface = SDL_LoadBMP("assassin.bmp");
    SDL_Rect dstRect = {200, 200, 0, 0};

    //draw image
    SDL_BlitSurface(pImageSurface, NULL, pWindowSurface, &dstRect);
    SDL_UpdateWindowSurface(pWindow);
    SDL_Delay(5000);

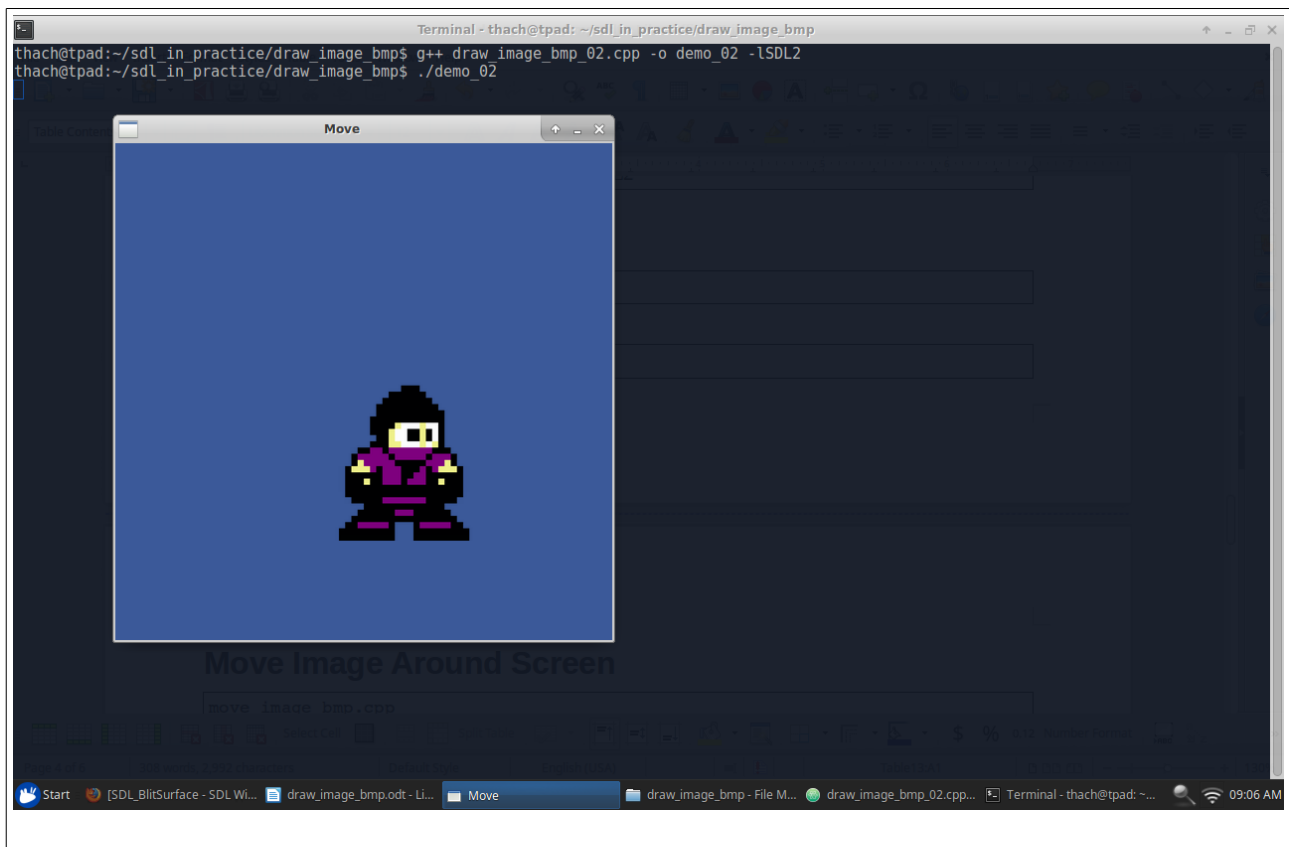
    //destroy
    SDL_FreeSurface(pImageSurface);
    SDL_DestroyWindow(pWindow);
    SDL_Quit();
}
```

Compile

```
g++ draw_image_bmp_02.cpp -o demo_02 -lSDL2
```

Run

```
./demo_02
```



Draw A Part Of Image

Declare a variable store which part of image that need to draw and pass it to `SDL_BlitSurface()` function.

```
SDL_Rect srcRect = {x, y, w, h};  
SDL_BlitSurface(pImageSurface, &srcRect, pWindowSurface, NULL);
```

srcRect: represent the part of image that need to draw.

```
draw_image_bmp_03.cpp  
  
#include <SDL2/SDL.h>  
  
int main()  
{  
    //init  
    SDL_Init(SDL_INIT_VIDEO);  
    SDL_Window* pWindow = SDL_CreateWindow("Move", 100, 100, 500, 500, 0);  
    SDL_Surface* pWindowSurface = SDL_GetWindowSurface(pWindow);  
    SDL_FillRect(pWindowSurface, NULL, 0x3b5999);  
  
    //load image  
    SDL_Surface* pImageSurface = SDL_LoadBMP("assassin.bmp");  
  
    SDL_Rect srcRect = {30, 30, 120, 120};  
  
    //draw image
```

```

    SDL_BlitSurface(pImageSurface, &srcRect, pWindowSurface, NULL);
    SDL_UpdateWindowSurface(pWindow);
    SDL_Delay(5000);

    //destroy
    SDL_FreeSurface(pImageSurface);
    SDL_DestroyWindow(pWindow);
    SDL_Quit();
}

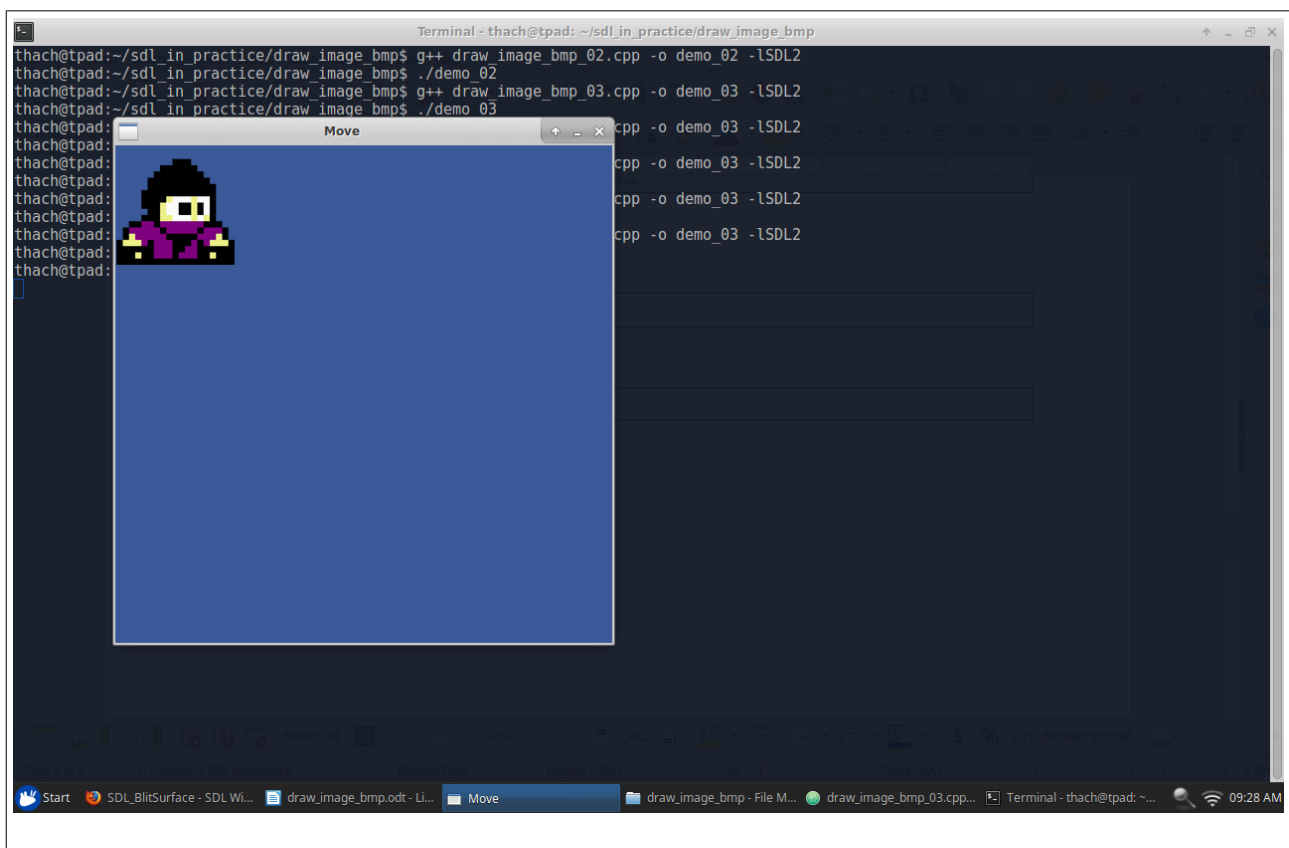
```

Compile

```
g++ draw_image_bmp_03.cpp -o demo_03 -lSDL2
```

Run

```
./demo_03
```



Move Image Around Screen

Add drawing image to game loop and update it position.

move_image_bmp.cpp

```
#include <SDL2/SDL.h>

int main()
{
    SDL_Init(SDL_INIT_VIDEO);

    SDL_Window *pWindow = SDL_CreateWindow("Move", 100, 100, 800, 480, 0);
    SDL_Surface *pWindowSurface = SDL_GetWindowSurface(pWindow);

    SDL_Surface* pImageSurface = SDL_LoadBMP("assassin.bmp");

    SDL_Rect dstRect = {0, 0, 0, 0};
    int speed = 1;

    bool isRunning = true;
    SDL_Event event;
    while(isRunning)
    {
        //draw
        SDL_FillRect(pWindowSurface, NULL, 0x3b5999);
        SDL_BlitSurface(pImageSurface, NULL, pWindowSurface, &dstRect);
        SDL_UpdateWindowSurface(pWindow);

        //update
        dstRect.x += speed;
        if (dstRect.x > 700 || dstRect.x < 0)
            speed = -speed;

        //handle input
        SDL_PollEvent(&event);
        if (event.type == SDL_QUIT)
        {
            isRunning = false;
        }
    }

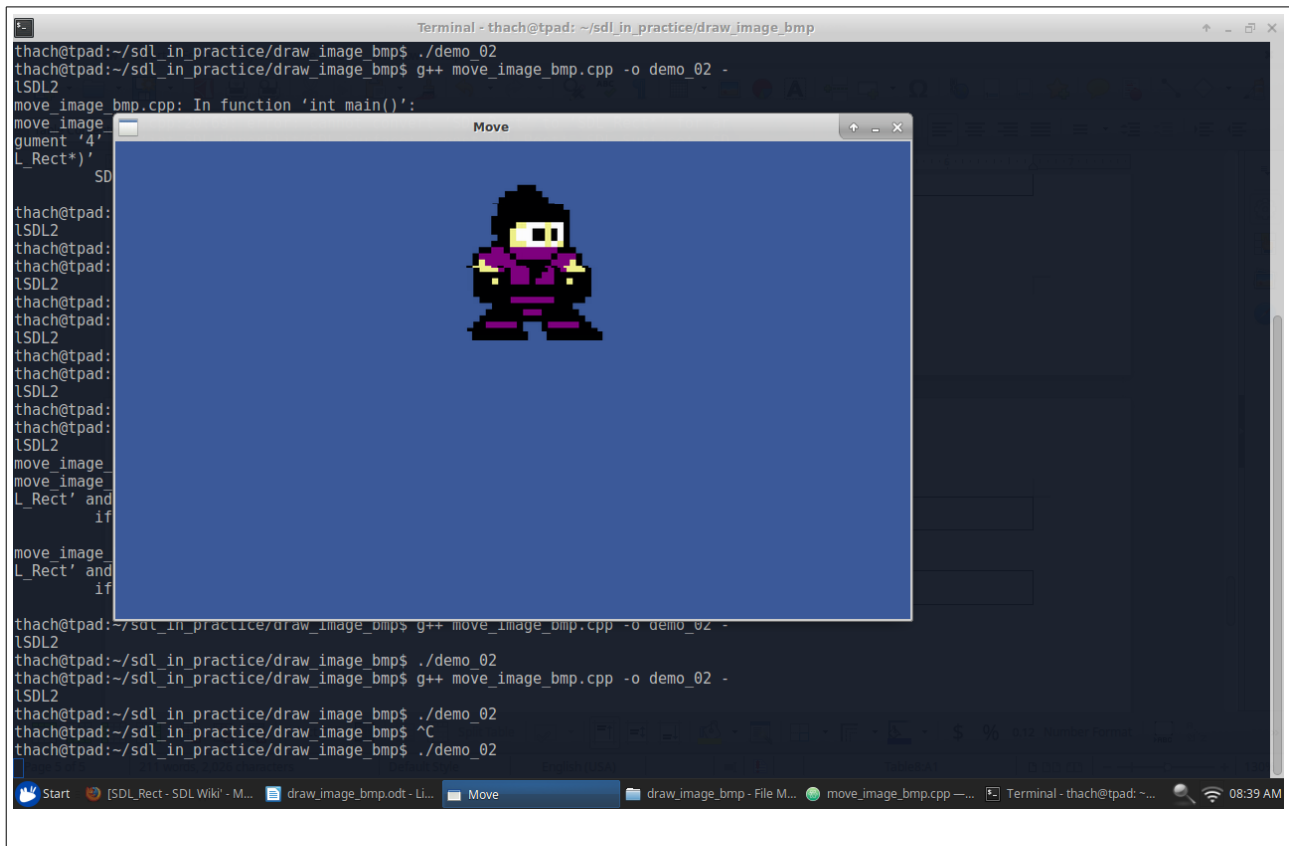
    //destroy
    SDL_FreeSurface(pImageSurface);
    SDL_DestroyWindow(pWindow);
    SDL_Quit();
}
```

Compile

```
g++ move_image_bmp.cpp -o demo_02 -lSDL2
```

Run

./demo_02



```
Terminal - thach@tpad: ~/sdl_in_practice/draw_image_bmp
thach@tpad:~/sdl_in_practice/draw_image_bmp$ ./demo_02
thach@tpad:~/sdl_in_practice/draw_image_bmp$ g++ move_image_bmp.cpp -o demo_02 -
LSDL2
move_image_bmp.cpp: In function 'int main()':
move_image_bmp.cpp:14: error: 'SDL_Rect' does not name a type
   14 |     SDL_Rect rect = {0, 0, 100, 100};
      |     ^~~~~~
thach@tpad:~/sdl_in_practice/draw_image_bmp$ g++ move_image_bmp.cpp -o demo_02 -
LSDL2
thach@tpad:~/sdl_in_practice/draw_image_bmp$ ./demo_02
thach@tpad:~/sdl_in_practice/draw_image_bmp$ g++ move_image_bmp.cpp -o demo_02 -
LSDL2
thach@tpad:~/sdl_in_practice/draw_image_bmp$ ./demo_02
thach@tpad:~/sdl_in_practice/draw_image_bmp$ ^C
thach@tpad:~/sdl_in_practice/draw_image_bmp$ ./demo_02
thach@tpad:~/sdl_in_practice/draw_image_bmp$
```

The screenshot shows a terminal window with a dark background. A window titled "Move" is open in the foreground, displaying a pixelated character with a yellow face and purple body on a blue background. The terminal window shows the user attempting to compile and run a C++ program. The first compilation attempt fails due to an error with the `SDL_Rect` type. After correcting the code, the program is successfully compiled and executed, displaying the character in the "Move" window. The terminal shows the user pressing `^C` to stop the program.

Draw Image In Depth

Function SDL_BlitSurface()

```
int SDL_BlitSurface(SDL_Surface*    srcSurface,
                    const SDL_Rect*  srcRect,
                    SDL_Surface*    dstSurface,
                    SDL_Rect*        dstRect)
```

Suppose, we are playing picture pasting game. We have to paste an ninja picture into a blue sky picture.

srcSurface: is ninja picture

dstSurface: is blue sky picture.

dstRect: is the location on blue sky picture where we need to put ninja on.

srcRect: is the part of ninja picture that we need to use. We can use the entire of ninja picture or a part of it.

If dstRect is NULL, ninja will be put on upper left corner (position: $x = 0$, $y = 0$).

In dstRect, only the position parameters (x,y) are used. The size parameters (width, height) are ignored.

If srcRect is NULL, the entire of ninja picture will be used.