

原 SDL2源代码分析7：显示（SDL_RenderPresent()）

2014年11月09日 00:59:05 阅读数：9770

=====

SDL源代码分析系列文章列表：

[SDL2源代码分析1：初始化（SDL_Init\(\)）](#)

[SDL2源代码分析2：窗口（SDL_Window）](#)

[SDL2源代码分析3：渲染器（SDL_Renderer）](#)

[SDL2源代码分析4：纹理（SDL_Texture）](#)

[SDL2源代码分析5：更新纹理（SDL_UpdateTexture\(\)）](#)

[SDL2源代码分析6：复制到渲染器（SDL_RenderCopy\(\)）](#)

[SDL2源代码分析7：显示（SDL_RenderPresent\(\)）](#)

[SDL2源代码分析8：视频显示总结](#)

=====

上一篇文章分析了SDL纹理赋值给渲染目标的函数SDL_RenderCopy()。这篇文章分析SDL显示视频最后的一个函数：SDL_RenderPresent()。



SDL播放视频的代码流程如下所示。

初始化:

SDL_Init(): 初始化SDL。

SDL_CreateWindow(): 创建窗口（Window）。

SDL_CreateRenderer(): 基于窗口创建渲染器（Render）。

SDL_CreateTexture(): 创建纹理（Texture）。

循环渲染数据:

SDL_UpdateTexture(): 设置纹理的数据。

SDL_RenderCopy(): 纹理复制给渲染器。

SDL_RenderPresent(): 显示。

上篇文章分析了该流程中的第6个函数SDL_RenderCopy()。本文继续分析该流程中的最后一个函数SDL_RenderPresent()。

SDL_RenderPresent()

函数简介

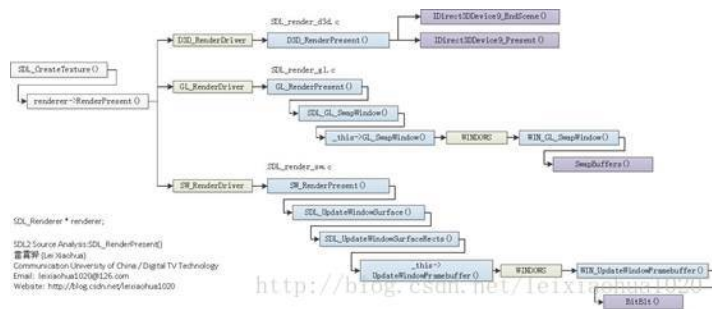
SDL使用SDL_RenderPresent()显示画面。SDL_RenderPresent()的原型如下。

```
[cpp]    
1. void SDLCALL SDL_RenderPresent(SDL_Renderer * renderer);
```

参数renderer用于指定渲染器。

函数调用关系图

SDL_RenderPresent()关键函数的调用关系可以用下图表示。



上面的图片不太清晰，更清晰的图片上传到了相册里面：

<http://my.csdn.net/leixiaohua1020/album/detail/1794103>

把相册里面的图片保存下来就可以得到清晰的图片了。

源代码分析

SDL_RenderPresent()的源代码位于render\SDL_render.c中。如下所示。

```
[cpp]
1. void SDL_RenderPresent(SDL_Renderer * renderer)
2. {
3.     CHECK_RENDERER_MAGIC(renderer, );
4.
5.
6.     /* Don't draw while we're hidden */
7.     if (renderer->hidden) {
8.         return;
9.     }
10.    renderer->RenderPresent(renderer);
11. }
```

从源代码中可以看出，SDL_RenderPresent()调用了SDL_Renderer的RenderPresent()方法显示图像。

下面我们详细看一下几种不同的渲染器的RenderPresent()的方法。

1.

Direct3D

Direct3D 渲染器中对应RenderPresent()的函数是D3D_RenderPresent()，它的源代码如下所示（位于render\direct3d\SDL_render_d3d.c）。

```
[cpp]
1. static void D3D_RenderPresent(SDL_Renderer * renderer)
2. {
3.     D3D_RenderData *data = (D3D_RenderData *) renderer->driverdata;
4.     HRESULT result;
5.
6.
7.     if (!data->beginScene) {
8.         IDirect3DDevice9_EndScene(data->device);
9.         data->beginScene = SDL_TRUE;
10.    }
11.
12.
13.    result = IDirect3DDevice9_TestCooperativeLevel(data->device);
14.    if (result == D3DERR_DEVICELOST) {
15.        /* We'll reset later */
16.        return;
17.    }
18.    if (result == D3DERR_DEVICENOTRESET) {
19.        D3D_Reset(renderer);
20.    }
21.    result = IDirect3DDevice9_Present(data->device, NULL, NULL, NULL, NULL);
22.    if (FAILED(result)) {
23.        D3D_SetError("Present()", result);
24.    }
25. }
```

从代码中可以看出，该函数调用了2个最关键Direct3D的API：



IDirect3DDevice9_EndScene()：结束一个场景。

IDirect3DDevice9_Present()：显示。



2.

OpenGL



OpenGL渲染器中对应RenderPresent()的函数是GL_RenderPresent(), 它的源代码如下所示（位于render\opengl\SDL_render_gl.c）。

```
[cpp]    
1. static void GL_RenderPresent(SDL_Renderer * renderer)  
2. {  
3.     GL_ActivateRenderer(renderer);  
4.  
5.  
6.     SDL_GL_SwapWindow(renderer->window);  
7. }
```

代码比较简单，只有两行。关键的显示函数位于SDL_GL_SwapWindow()函数中。下面看一下SDL_GL_SwapWindow()的代码（位于video\SDL_video.c。感觉这里调用关系稍微有点乱...）。

```
[cpp]    
1. void SDL_GL_SwapWindow(SDL_Window * window)  
2. {  
3.     CHECK_WINDOW_MAGIC(window, );  
4.  
5.  
6.     if (!(window->flags & SDL_WINDOW_OPENGL)) {  
7.         SDL_SetError("The specified window isn't an OpenGL window");  
8.         return;  
9.     }  
10.  
11.  
12.     if (SDL_GL_GetCurrentWindow() != window) {  
13.         SDL_SetError("The specified window has not been made current");  
14.         return;  
15.     }  
16.  
17.  
18.     _this->GL_SwapWindow(_this, window);  
19. }
```

从上述代码中可以看出，SDL_GL_SwapWindow()调用了SDL_VideoDevice的GL_SwapWindow()函数。我们看一下在“Windows视频驱动”的情况下，该函数的代码。在“Windows视频驱动”的情况下，调用GL_SwapWindow()实际上是调用了WIN_GL_SwapWindow()函数。看一下WIN_GL_SwapWindow()函数的代码（位于video\windows\SDL_windowsopengl.c）。



```
[cpp]    
1. void WIN_GL_SwapWindow(_THIS, SDL_Window * window)  
2. {  
3.     HDC hdc = ((SDL_WindowData *) window->driverdata)->hdc;  
4.  
5.  
6.     SwapBuffers(hdc);  
7. }
```

代码中调用了简单的一个函数SwapBuffers(), 完成了显示功能。

3.

Software

Software渲染器中对应RenderPresent()的函数是SW_RenderPresent(), 它的源代码如下所示（位于render\software\SDL_render_sw.c）。

```
[cpp]    
1. static void SW_RenderPresent(SDL_Renderer * renderer)  
2. {  
3.     SDL_Window *window = renderer->window;  
4.  
5.  
6.     if (window) {  
7.         SDL_UpdateWindowSurface(window);  
8.     }  
9. }
```

从代码中可以看出，SW_RenderPresent()调用了函数SDL_UpdateWindowSurface()。我们看一下SDL_UpdateWindowSurface()的代码（位于video\SDL_video.c）。

```
[cpp]
1. int SDL_UpdateWindowSurface(SDL_Window * window)
2. {
3.     SDL_Rect full_rect;
4.
5.
6.     CHECK_WINDOW_MAGIC(window, -1);
7.
8.
9.     full_rect.x = 0;
10.    full_rect.y = 0;
11.    full_rect.w = window->w;
12.    full_rect.h = window->h;
13.    return SDL_UpdateWindowSurfaceRects(window, &full_rect, 1);
14. }
```

SDL_UpdateWindowSurface()又调用了另一个函数SDL_UpdateWindowSurfaceRects()。继续看SDL_UpdateWindowSurfaceRects()的代码。

```
[cpp]
1. int SDL_UpdateWindowSurfaceRects(SDL_Window * window, const SDL_Rect * rects,
2.                                 int numrects)
3. {
4.     CHECK_WINDOW_MAGIC(window, -1);
5.
6.
7.     if (!window->surface_valid) {
8.         return SDL_SetError("Window surface is invalid, please call SDL_GetWindowSurface() to get a new surface");
9.     }
10.
11.
12.     return _this->UpdateWindowFramebuffer(_this, window, rects, numrects);
13. }
```

SDL_UpdateWindowSurfaceRects()调用了SDL_VideoDevice的UpdateWindowFramebuffer()函数。在“Windows视频驱动”的情况下，相当于调用了WIN_UpdateWindowFramebuffer()。我们看一下该函数的代码（位于video\windows\SDL_windowsframebuffer.c）

```
[cpp]
1. int WIN_UpdateWindowFramebuffer(_THIS, SDL_Window * window, const SDL_Rect * rects, int numrects)
2. {
3.     SDL_WindowData *data = (SDL_WindowData *) window->driverdata;
4.
5.
6.     BitBlt(data->hdc, 0, 0, window->w, window->h, data->mdc, 0, 0, SRCCOPY);
7.     return 0;
8. }
```

经过一系列的寻找之后，终于找到了Software渲染器显示视频的“源头”：BitBlt()函数。

版权声明：本文为博主原创文章，未经博主允许不得转载。 <https://blog.csdn.net/leixiaohua1020/article/details/40895797>

文章标签： [SDL](#) [Direct3D](#) [OpenGL](#) [GDI](#) [显示](#)

个人分类： [SDL](#)

所属专栏： [开源多媒体项目源代码分析](#)

此PDF由spygg生成,请尊重原作者版权!!!

我的邮箱:liushidc@163.com