(This is the documentation for SDL3, which is under heavy development and the API is changing! SDL2 is the current stable version!)

SDL_CreateWindow

Create a window with the specified dimensions and flags.

Header File

Defined in <SDL3/SDL_video.h>

Syntax

```
SDL_Window* SDL_CreateWindow(const char *title, int w, int h, SDL_WindowFlags flags);
```

Function Parameters

title	the title of the window, in UTF-8 encoding
w	the width of the window
h	the height of the window
flags	0, or one or more SDL_WindowFlags OR'd together

Return Value

Returns the window that was created or NULL on failure; call SDL_GetError() for more information.

Remarks

`flags` may be any of the following OR'd together:

- `SDL_WINDOW_FULLSCREEN`: fullscreen window at desktop resolution
- `SDL_WINDOW_OPENGL`: window usable with an OpenGL context
- `SDL_WINDOW_VULKAN`: window usable with a Vulkan instance
- `SDL_WINDOW_METAL`: window usable with a Metal instance
- `SDL_WINDOW_HIDDEN`: window is not visible
- `SDL_WINDOW_BORDERLESS`: no window decoration
- `SDL_WINDOW_RESIZABLE`: window can be resized
- `SDL_WINDOW_MINIMIZED`: window is minimized
- `SDL_WINDOW_MAXIMIZED`: window is maximized
- `SDL_WINDOW_MOUSE_GRABBED`: window has grabbed mouse focus

The SDL_Window is implicitly shown if SDL_WINDOW_HIDDEN is not set.

On Apple's macOS, you **must** set the NSHighResolutionCapable Info.plist property to YES, otherwise you will not receive a High-DPI OpenGL canvas.

The window pixel size may differ from its window coordinate size if the window is on a high pixel density display. Use SDL_GetWindowSize() to query the client area's size in window coordinates, and SDL_GetWindowSizeInPixels() or SDL_GetRenderOutputSize() to query the drawable size in pixels. Note that the drawable size can vary after the window is created and should be queried again if you get an SDL_EVENT_WINDOW_PIXEL_SIZE_CHANGED event.

If the window is created with any of the SDL_WINDOW_OPENGL or SDL_WINDOW_VULKAN flags, then the corresponding LoadLibrary function (SDL_GL_LoadLibrary or SDL_Vulkan_LoadLibrary) is called and the corresponding UnloadLibrary function is called by SDL_DestroyWindow().

If SDL_WINDOW_VULKAN is specified and there isn't a working Vulkan driver, SDL_CreateWindow() will fail because SDL_Vulkan_LoadLibrary() will fail.

If SDL_WINDOW_METAL is specified on an OS that does not support Metal, SDL_CreateWindow() will fail.

On non-Apple devices, SDL requires you to either not link to the Vulkan loader or link to a dynamic library version. This limitation may be removed in a future version of SDL.

Version

This function is available since SDL 3.0.0.

Code Examples

```c
// Example program:
// Using SDL3 to create an application window

#include <SDL3/SDL.h>

int main(int argc, char* argv[]) {

    SDL_Window *window;                     // Declare a pointer

    SDL_Init(SDL_INIT_VIDEO);               // Initialize SDL2

    // Create an application window with the following settings:
    window = SDL_CreateWindow(
        "An SDL3 window",                   // window title
        640,                                // width, in pixels
        480,                                // height, in pixels
```

```
        SDL_WINDOW_OPENGL                  // flags - see below
    );

    // Check that the window was successfully created
    if (window == NULL) {
        // In the case that the window could not be made...
        SDL_LogError(SDL_LOG_CATEGORY_ERROR, "Could not create window: %s\n", SDL_GetError()
        return 1;
    }

    // The window is open: could enter program loop here (see SDL_PollEvent())

    SDL_Delay(3000);  // Pause execution for 3000 milliseconds, for example

    // Close and destroy the window
    SDL_DestroyWindow(window);

    // Clean up
    SDL_Quit();
    return 0;
}
```

See Also

- SDL_CreatePopupWindow
- SDL_CreateWindowWithProperties
- SDL_DestroyWindow

CategoryAPI, CategoryAPIFunction, CategoryVideo