

DualSense Windows API

Version 0.1

Ludwig Füchsl

November 23, 2020

Contents

1	Important information	2
1.1	Trademarks and affiliation	2
1.2	Sources	2
1.3	License	3
2	Introduction	4
3	Features	5
3.1	Overview	5
3.2	Feature List	6
4	Installation	11
4.1	Prebuild installation	11
4.2	Self build installation	12
5	Quick start	14

1 Important information

1.1 Trademarks and affiliation

”PlayStation”, ”PlayStation Family Mark”, ”PS5 logo”, ”PS5”, ”DualSense” and ”DUALSHOCK” are registered trademarks or trademarks of Sony Interactive Entertainment Inc. ”SONY” is a registered trademark of Sony Corporation.

The Author is not affiliated in any kind with Sony Interactive Entertainment Inc.!

Using this library may void your / your clients / your users / your customers controllers warranty! You as the redistributor of the precompiled or self compiled library have to make sure the controller will not be damaged by the functionality you use or at least point out the possible risk to your users / clients / customer!

Probably no damage or failure at all. This statement is just for my own safety!

1.2 Sources

This work is derivative from others work. Special thanks goes to:

- GitHub user dogtopus:
<https://gist.github.com/dogtopus/894da226d73afb3bdd195df41b3a26aa>.
- Reddit user ginkgobitter: https://www.reddit.com/r/gamedev/comments/jumvi5/dualsense_haptics_leds_and_more_hid_output_report/
- GitHub user Ryochan7: <https://github.com/Ryochan7/DS4Windows/tree/dualsense-integration>
Copyright (c) 2019 Travis Nickles - MIT License: <https://github.com/Ryochan7/DS4Windows/blob/jay/LICENSE.txt>
- And the amazing community at DS4Windows <https://github.com/Ryochan7/DS4Windows/issues/1545>

1.3 License

MIT License

Copyright (c) 2020 Ludwig Füchsl

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

2 Introduction

Welcome to the DualSense on Windows API documentation. This API will help you integrating the Sony PS5 DualSense controller into your application or game for Windows. This document will guide you through the complete flow of integration. Starting with learning the DualSense Features up to deeply understanding every feature of this api.

The documentation is structured as follows:

- Descriptions of the DualSense's features
- installation and self compiling
- getting started guide
- api references

We recommend starting with reading through all features as far as you are not familiar with the DualSense controllers feature. Continuing with the installation and getting started guide to get your own demo application up and running. Then you can use the API references to integrate the api into your application.

3 Features

In the following section the features of the DualSense controller will be explained.

3.1 Overview

- **Connectivity** The DualSense controller can be used via Bluetooth or USB (USB C).
- **Integrated battery** Featuring an integrated battery the DualSense controller is best used via Bluetooth. The controller can be charged via USB type-C.



(a) Front View



(b) Rear View

Figure 1: The DualSense controller

The DualSense controller features the following peripherals:

- Two XY-Axis analog sticks with integrated push button.
- Two adaptive triggers (are able to provide feedback).
- Two shoulder buttons.
- DPad with the ability to press two neighbor buttons simultaneously.
- The default Square, Cross, Circle and Triangle PlayStation buttons.
- Dual-touch touchpad with integrated push button, surrounded with five player indication LEDs on the bottom and RGB-LED lightbar on the sides.
- Menu, share, microphone mute and PlayStation button.
- 3-Axis Accelerometer and Gyroscope.

- Two rumble motors (Hard and Soft one). Can alternatively used as haptic feedback (not supported yet).
- Integrated speaker and microphone.
- Stereo audio jack.

3.2 Feature List

Analog sticks Each analog stick has two axis with 8-Bit precision each. The analog sticks will automatically return to their center position if released. They are mapped to the range -128 to 127 where 0 means center, -128 means left/bottom on X/Y-Axis and 127 means right/top on X/Y-Axis. Using the analog values requires the correction of the dead zones, because a released stick will most likely not have the value $R_{xy}(0; 0)$ it will be a bit off. Same goes for the extreme values which will also be off and not be exactly $T_{xy}(0; 127)$, $L_{xy}(-128; 0)$, etc..



Figure 2: Analog sticks

Adaptive trigger The DualSense controller features two 8-Bit analog triggers. It is possible to read the trigger values as 8-Bit continuous values or alternatively as binary button input. Aside from the normal trigger operation the adaptive triggers can be configured to simulate various force feedback effects. It is possible for example to simulate a gun trigger.



Figure 3: Adaptive triggers

Bumpers The two L/R Bumpers located over the adaptive triggers can be read as normal button inputs.



Figure 4: L/R Bumpers

DPAD and PS Buttons The DualSense controller feature a DPAD and the default well know PlayStation Square, Cross, Circle and Triangle buttons. The DPAD is capable of registering two simultaneously pressed buttons, however the two buttons must be neighbors. The PS-Buttons are being registered as four individual binary values.



Figure 5: DPAD and PS-Buttons

Other Buttons The DualSense controller feature several more buttons. These are:

- **Menu button** Should be used to open the in-game menu.
- **Share button** Should be used to open the in-game photo mode.
- **PlayStation button** Can be used to open a in-game overlay (Look at the know issues to get an additional use case of this button).
- **Mic button** Should be used to mute the microphone.

All the listed buttons are readable through individual binary values.



Figure 6: Left to right, top to bottom: Share, Menu, PlayStation and Mic Button

Touch Pad The touch-pad and the surrounding lightbar are featuring several functions:

- **Dual finger touch** The touch-pad itself is able to track two fingers simultaneously.
- **Integrated push button** The touch-pad integrates as momentary push button operated by pushing the pad down.
- **Lightbar** The left and right surrounding is able to light up in full 8-Bit RGB colors. The lib is providing several helpers to convert other color formats to the 8-Bit RGB UCHAR formate.
- **Player indication LEDs** On the bottom of the touch-pad are five player indication LEDs located. These LEDs are group in one left, three middle and one right led. The brightness of the LEDs is controllable and the LEDs are able to fade in.

When using the touch-pad make sure to implement hysteresis, dead zones and tolerances. It may also helpful to accumulate the values over multiple frame to get a more stable result but this will also increase latency.



Figure 7: Touch Pad

Accelerometer and Gyroscope The DualSense controller is able to measure its acceleration (By moving the controller around) and to track its rotation. Measurements are done with 16-Bit precision in all three XYZ-Axis. Make sure to implement hysteresis, dead zones and tolerances. It may also helpful to accumulate the values over multiple frame to get a more stable result but this will also increase latency.

- **Accelerometer** Measures the acceleration.
- **Gyroscope** Measures the controllers rotation. Currently you have to implement calibration in your own codebase.

Rumble motors / Haptic feedback The DualSense feature two Haptic feedback devices. Thees device work similar like normal speaker, but they are not good in producing tones, they are good in producing vibration. It is possible to send an audio signal directly to those haptic speakers (However currently not supported by this API).

The controller supports simulating the normal soft and hard rumble motors using the haptic speakers. When using this mode both motors can be controlled with the usual 8-Bit values. The left rumble feels hard, the right one soft.

Integrated speaker and microphone Featuring two microphones and one mono speaker, the DualSense is able to produce and pickup audio. This

features will be supported in the future. However it is possible to address thees devices with the default WASAPI independently.

Stereo audio jack Directly under the little microphone icon is a stereo headphone audio jack. It is possible to retrieve the connection status of this jack. However just like the speaker it is currently not supported to produce audio through the API.

4 Installation

We recommend using the prebuild dynamically linked release. But it is also possible to build the API from source. This might be the only way if you want to...

- statically link the library.
- get the latest (maybe unstable and undocumented) development release.
- sign and supply your company info inside the DLL info.
- make changes to the source code to better fit your application.

4.1 Prebuild installation

Before stating the integration process please make sure you have the latest release. You can find all releases here: <https://github.com/Ohjurot/DualSense-Windows/releases>

Inside the downloaded zip file you will find the following files:

DualSenseWindows.zip
└── ds5w.h
└── ds5w_x64.dll
└── ds5w_x64.lib
└── ds5w_x86.dll
└── ds5w_x86.lib
└── DualSenseWindows.pdf
└── LICENSE

The PDF file is the latest version of the document your are currently reading and LICENSE is a copy of the repository license file. All other files are required for integration into your project, thees steps will be explained next.

Including the header file When using the releases from the github page the single header `ds5w.h` is used. You can copy paste this file into the vendor directory of your project (The exact directory doesn't matter as far as it is accessible as a include directory).

Copying the binary DLL The API is coming with two different DLLs:

- **ds5w_x64.dll** Used when building x64 / 64-Bit projects.
- **ds5w_x86.dll** Used when building x86 / Win32 / 32-Bit projects.

Copy the DLL matching your target build to the output directory of your project.

Referencing the LIB file In order to compile your project successfully it is required to add the included lib file to the linker input. First you need to copy the lib file to your projects lib directory. Chose the lib file according to the name of the DLL you picked. After copying the file you have to add the files name to the linker input.

4.2 Self build installation

First you need to download or clone the project to your computer (We recommend to use the github tags to get the code that matches a specific release). Please check if you have the following prerequisites:

- VisualStudio 16.8.0 or later
- Windows SDK 10.0.19041.0 (Other version will probably work... Try to match the SDK version with the SDK version of your project!)
- Platform toolset **Visual Studio 2019 (v142)** Other version will probably work... Try to match with your project!)
- Windows Driver Kit (WDK) Version 10.0.19041.1 or later minor versions. (Should match your major SDK version. This part matters: 10.0.19041)

After downloading the project you can open the project by double-clicking the file **VS19_Solution\VS19_Solution.sln**

Building the project is very easy! Just execute a batch-built by selecting **Build → Batch Build → Select All → Build**. After building the library you can find several configurations under the following path

VS19_Solution\bin\DualSenseWindows\...

- **DebugDll-x64 / DebugDll-x86** Debug DLL version for 64/32-Bit
- **ReleaseDll-x64 / ReleaseDll-x86** Release DLL version for 64/32-Bit

- `Debug-x64 / Debug-x64` Debug static link version for 64/32-Bit
- `Release-x64 / Release-x86` Release static link version for 64/32-Bit

Every self build version will be based on several header files located in `VS19_Solution\DualSenseWindows\include\DualSenseWindows`.

Attention: When using the static link version your application needs additional linking to `hid.lib` and `setupapi.lib` from the Windows Driver Kit. It is also required to define `DS5W_USE_LIB` to prevent the DLL import attempt.

5 Quick start

Quick start