Installation

Pieter P

Important: These are the installation instructions for developers, if you just want to use the library, please see these installation instructions for users.

I strongly recommend using a Linux system for development.
I'm using Ubuntu, so the installation instructions will focus mainly on this distribution, but it should work fine on others as well.
If you're using Windows, my most sincere condolences.

Installing the compiler, GNU Make, CMake and Git

For development, I use GCC 9, because it's the latest version, and its error messages and warnings are better than previous versions.

On Ubuntu, GCC 9 can be installed through the `ppa:ubuntu-toolchain-r/test` repository:

```
$ sudo add-apt-repository ppa:ubuntu-toolchain-r/test
$ sudo apt update
$ sudo apt install gcc-9 g++-9
```

You'll also need GNU Make and CMake as a build system, and Git for version control. Now is also a good time to install GNU Wget, if you haven't already, it comes in handy later.

```
$ sudo apt install make cmake git wget
```

Installing Python 2

The ESP32 Core Toolchain requires Python 2. On newer versions of Ubuntu, this is no longer installed by default.
You can install it from the repositories:

```
$ sudo apt install python
```

Installing the Arduino IDE, Teensy Core and ESP32 Core

To compile sketches for Arduino, I'm using the Arduino IDE. The library focusses on some of the more powerful Arduino-compatible boards, such as PJRC's Teensy 3.x and Espressif's ESP32 platforms.
To ensure that nothing breaks, tests are included for these boards specifically, so you have to install the appropriate Arduino Cores.

Arduino IDE

If you're reading this document, you probably know how to install the Arduino IDE.
Just in case, you can find the installation instructions here.

Teensyduino

The installation instructions for the Teensy Core can be found on the PJRC website: https://www.pjrc.com/teensy/td_download.html

ESP32 Arduino Core

On GitHub, you can find the installation instructions for the ESP32 Core: https://github.com/espressif/arduino-esp32/blob/master/docs/arduino-ide/boards_manager.md

Installing the Control Surface library

The next step is to download the Control Surface library itself. Just use Git to clone it into your `~/Arduino/libraries` folder. If you have write access to the repository, use SSH, otherwise, use HTTPS.

The `--recursive` option clones the Google Test submodule as well.

```
$ mkdir -p ~/Arduino/libraries && cd ~/Arduino/libraries
$ git clone --recursive git@github.com:tttapa/Control-Surface.git      # SSH
$ git clone --recursive https://github.com/tttapa/Control-Surface.git  # HTTPS
```

Installing the Arduino library dependencies

Some of the examples use the `Adafruit_SSD1306` and `Adafruit_GFX` libraries. You also need the Encoder and MIDIUSB libraries.
You can download them using the library manager in the Arduino IDE, or:

```
$ cd ~/Arduino/libraries
$ git clone https://github.com/adafruit/Adafruit-GFX-Library.git
$ git clone https://github.com/adafruit/Adafruit_SSD1306.git
$ git clone https://github.com/PaulStoffregen/Encoder.git
$ git clone https://github.com/arduino-libraries/MIDIUSB.git
```

Installing Visual Studio Code

I use Visual Studio Code (not to be confused with Visual Studio) to write code, format it, and run the build and documentation tasks.

Of course, you're free to use whatever IDE you prefer, but I do ask to run all tests and format all code using `clang-format` before committing, and this is much easier using VSCode, because there are VSCode tasks defined for building and running tests, and for generating and exporting the documentation.

You can download VSCode from the website: https://code.visualstudio.com/

Extensions

Once you've installed and launched VSCode, I'd recommend installing some extensions.
Open the extensions menu using `CTRL+SHIFT+X`.

- C/C++ - Microsoft
- CMake - twxs
- Code Spell Checker - Street Side Software
- GitLens - Eric Amodio
- Doxygen Documentation Generator - Christoph Schlosser

Installing Doxygen

Doxygen is used to generate the documentation of the library. I find it easiest to clone it from GitHub, and build it from source. The version in the Ubuntu repositories is pretty old.

```
$ sudo apt install flex bison
$ cd /tmp
$ git clone https://github.com/doxygen/doxygen.git
$ mkdir -p doxygen/build && cd doxygen/build
$ cmake \
      -G "Unix Makefiles" \
      -D "CMAKE_INSTALL_PREFIX=$HOME/.local" \
      -D "CMAKE_CXX_FLAGS=-O3 -march=native" \
      -D "CMAKE_C_FLAGS=-O3 -march=native" \
      -D "CMAKE_CXX_LINK_FLAGS=-flto" \
      -D "CMAKE_C_COMPILER=gcc-9" \
      -D "CMAKE_CXX_COMPILER=g++-9" ..
$ make -j$[$(nproc) * 2]
$ make install
```

Dot

Doxygen uses Dot to generate diagrams. Install the Graphviz package that contains Dot:

```
$ sudo apt install graphviz
```

Installing LCOV

The code coverage reports are generated using LCOV.
Install it from GitHub:

```
$ cd /tmp
$ git clone https://github.com/linux-test-project/lcov.git
$ cd lcov
$ make install PREFIX=$HOME/.local
```