

[Kernel](#) / [...](#) / [USB](#) / Building a Simple USB Driver

## Sample Code

# Building a Simple USB Driver

Set up and load a driver that logs output to the Console app.

[Download](#)

macOS 10.15+ | Xcode 12.0+

## Overview

This code project builds a simple USB driver. The driver logs events to the Console and responds to interrupts, such as when the user inserts or removes a device. Because it is a template driver with limited functionality, it can be used as a starting point for a more sophisticated driver.

## Configure the Sample Code Project

To run the sample code project in Xcode, make sure to use macOS 10.15.5 or later and Xcode 11.5 or later.

For the driver to load, System Integrity Protection (SIP) must be off. For more information on this process, see [Disabling and Enabling System Integrity Protection](#).

## Select a USB Test Device

This sample USB driver works by detecting when a USB device is plugged into a Mac and logging the event. The test device should be unsupported under macOS; otherwise, an existing driver will take precedence, recognize the device, and preempt this sample driver.

The driver detects the device based on a number of criteria inside of the Info.plist's `IOKitPersonalities > SimpleUSBInterfaceDriver` section, including configuration value, interface number, product identifier, and vendor identifier.

The sample driver requires the following command to find criteria that matches the keys in the Info.plist file for a selected USB device plugged into a Mac:

```
ioreg -w0 -rc IOUSBHostInterface -l
```

With the device's parameters identified and placed in the Info.plist file, the driver loads and initializes the device upon insertion.

## Build and Load the Driver

At build time, Xcode runs the "Launch Terminal" run script build phase, which launches the Terminal app in the directory where the newly-built driver resides.

The following commands copy the driver into the home folder, change the current directory to that location, set ownership and permissions, and load the driver.

```
sudo cp -R SimpleUSBInterfaceDriver.kext ~  
cd ~  
sudo chown -R root:wheel SimpleUSBInterfaceDriver.kext  
sudo chmod -R 755 SimpleUSBInterfaceDriver.kext  
sudo kextload SimpleUSBInterfaceDriver.kext
```

The kernel loads the driver and communicates the status through the logging system. The Console app shows the logging messages.

## Unload the Driver

When testing completes, the following command directs the kernel to unload the driver:

```
sudo kextunload SimpleUSBInterfaceDriver.kext
```

It is important to re-enable SIP once driver testing completes. Failure to re-enable SIP leaves the computer vulnerable to malicious code. See [Disabling and Enabling System Integrity Protection](#) for more information.

---

## See Also

### Device Communication

## IOUSBHostIOSourceClientRecordLink

A structure that represents a USB host input/output source client record entry.

## IOUSBHostIOSourceClientRecordList

A structure that represents a list of USB host input/output source client records.