HID API for Linux, Mac OS X, and Windows

About

HIDAPI is a multi-platform library which allows an application to interface with USB and Bluetooth HID-Class devices on Windows, Linux, and Mac OS X. While it can be used to communicate with standard HID devices like keyboards, mice, and Joysticks, it is most useful when used with custom (Vendor-Defined) HID devices. Many devices do this in order to not require a custom driver to be written for each platform. HIDAPI is easy to integrate with the client application, just requiring a single source file to be dropped into the application. On Windows, HIDAPI can optionally be built into a DLL.
Programs which use HIDAPI are driverless, meaning they do not require the use of a custom driver for each device on each platform.
HIDAPI provides a clean and consistent interface for each platform, making it easier to develop applications which communicate with USB HID devices without having to know the details of the HID libraries and interfaces on each platform.
The HIDAPI source also provides a GUI test application which can enumerate and communicate with any HID device attached to the system. The test GUI compiles and runs on all platforms supported by HIDAPI.

What Does the API Look Like?

Doxygen HTML documentation can be found here.
The API provides an easy method to enumerate HID devices attached to the system, and easy access to the functionality of the most commonly used HID functions including transfer of Input, Output, and Feature Reports. The sample program, which communicates with a modified version of the **USB Generic HID** sample which is part of the **Microchip Application Library** (in folder "`Microchip Solutions\USB Device - HID - Custom Demos\Generic HID - Firmware`" when the Microchip Application Framework is installed), looks like this (with error checking removed for simplicity):

```
#include <stdio.h>
#include <stdlib.h>

#include "hidapi.h"


int main(int argc, char* argv[])
{
        int res;
        unsigned char buf[65];
        #define MAX_STR 255
        wchar_t wstr[MAX_STR];
        hid_device *handle;
        int i;

        // Enumerate and print the HID devices on the system
        struct hid_device_info *devs, *cur_dev;
        devs = hid_enumerate(0x0, 0x0);
        cur_dev = devs;
        while (cur_dev) {
                printf("Device Found\n  type: %04hx %04hx\n  path: %s\n  serial_number:
%ls",
                        cur_dev->vendor_id, cur_dev->product_id, cur_dev->path, cur_dev-
>serial_number);
                printf("\n");
                printf("  Manufacturer: %ls\n", cur_dev->manufacturer_string);
                printf("  Product:      %ls\n", cur_dev->product_string);
                printf("\n");
                cur_dev = cur_dev->next;
        }
        hid_free_enumeration(devs);


        // Open the device using the VID, PID,
        // and optionally the Serial number.
        handle = hid_open(0x4d8, 0x3f, NULL);

        // Read the Manufacturer String
        res = hid_get_manufacturer_string(handle, wstr, MAX_STR);
        printf("Manufacturer String: %ls\n", wstr);

        // Read the Product String
        res = hid_get_product_string(handle, wstr, MAX_STR);
        printf("Product String: %ls\n", wstr);

        // Read the Serial Number String
        res = hid_get_serial_number_string(handle, wstr, MAX_STR);
```

```
        printf("Serial Number String: %ls", wstr);
        printf("\n");

        // Send a Feature Report to the device
        buf[0] = 0x2; // First byte is report number
        buf[1] = 0xa0;
        buf[2] = 0x0a;
        res = hid_send_feature_report(handle, buf, 17);

        // Read a Feature Report from the device
        buf[0] = 0x2;
        res = hid_get_feature_report(handle, buf, sizeof(buf));

        // Print out the returned buffer.
        printf("Feature Report\n   ");
        for (i = 0; i < res; i++)
                printf("%02hhx ", buf[i]);
        printf("\n");

        // Set the hid_read() function to be non-blocking.
        hid_set_nonblocking(handle, 1);

        // Send an Output report to toggle the LED (cmd 0x80)
        buf[0] = 1; // First byte is report number
        buf[1] = 0x80;
        res = hid_write(handle, buf, 65);

        // Send an Output report to request the state (cmd 0x81)
        buf[1] = 0x81;
        hid_write(handle, buf, 65);

        // Read requested state
        res = hid_read(handle, buf, 65);
        if (res < 0)
                printf("Unable to read()\n");

        // Print out the returned buffer.
        for (i = 0; i < res; i++)
                printf("buf[%d]: %d\n", i, buf[i]);

        return 0;
}
```

License

HIDAPI may be used by one of three licenses as outlined in LICENSE.txt. These licenses are:

- GPL v3 (see LICENSE-gpl3.txt),
- BSD (see LICENSE-bsd.txt),
- The more liberal original HIDAPI license (see LICENSE-orig.txt).

The user of HIDAPI (the developer who uses HIDAPI in their code) can choose to use HIDAPI under any of the three licenses at their discretion. For example:

1. An author of GPL software would likely use HIDAPI under the terms of the GPL.
2. An author of commercial, closed-source software would likely use HIDAPI under the terms of either the BSD-style license or the original HIDAPI license.

The idea is to make HIDAPI accessable to as many users as possible for both open- and closed-source applications, and to give users the flexibility to link with the code in any way they see fit.

Download

HIDAPI can be downloaded fromGitHub.A zip file is available from theDownload Page.To get the latest trunk revision (if you have git installed, run the following:

```
git clone git://github.com/signal11/hidapi.git
```

Build Instructions

Windows: Build the `.sln` file in the `windows/` directory using Visual Studio.**Linux:**Change to the `linux/` directory and run `make`.**Mac OS X:**Change to the `mac/` directory and run make.**To build the Test GUI:**

- On Windows, build the `.sln` file in the `hidtest/` directory. Make sure to first set up the externals (Fox Toolkit) as described in README.txt.
- On Linux and Mac, run `make` from the `hidtest/` directory. Make sure to first install fox-toolkit as described in README.txt

Contact

A mailing list is to be set up soon. For the time being, please direct inquiries and questions to alan@signal11.us.