THE ETERNAL RAMBLEINGS OF AN UNTIDY MIND

ERUM

Using Java libusb Wrapper to Control Dream Cheeky Thunder Turret

INTRODUCTION

After getting frustrated trying to get any Java USB library to work I figured it's probably worth writing about one that does (It seems many are old projects). Specifically if you want to control DreamCheeky's Thunder Turret, this guide is dedicated to you.

One of the first pitfalls I fell into is what documentation to read and where. With some of the libraries it seemed documentation was non existent. I'll attempt to link all the sources I used and comment the hell out of my code. I found myself asking "why?" A LOT as this is my first exposure Java USB and the libraries assume the reader has existing knowledge of USB protocols. I found many Python tutorials but here is one for Java.

BEFORE WE START

When writing this I was using Windows 7 64bit. Programming in NetBeans.

Read some of this: http://libusbjava.sourceforge.net/wp/

This is the project site for Java libusb wrapper we will be using. It details how to set-up your device for use with the libusb library. I will be attempting to go through everything in detail for Win7 first but I'm hoping to make another post in the near future about getting everything to work in Linux (Debian/Centos/Ubuntu likely).

FILES

libusb-win32-bin-1.2.6.0.zip:
http://sourceforge.net/projects/libusb-win32/files/libusb-win32-releases/1.2.6.0/

libusb jar library (ch.ntb.usb-0.5.9.jar):
http://sourceforge.net/projects/libusbjava/files/libusbjava-snapshots/20090517/

INSTALL LIBUSB DRIVER FOR THUNDER LAUNCHER

(SOURCE)

In order to allow libusb to see the usb device we need to create a driver for it. This is done with a tool provided with libusb-win32. Inside "bin" directory in the zip file run the inf-wizard.exe, select next and find the device we want to use in your Java program. For the Thunder Turret the Vendor ID is 0x2123 and the Product ID is 0x1010. Select this and click next. Click next again and select a place to save the driver. Install the driver then check that the USB device is using it.

Creating the driver will automatically copy over some dlls but one needs to be copied over manually:

- LibusbJava.dll -> windowssystem32

By default the device manage will not use this driver as it is not digitally signed. We may need to force the device manager to use our driver. In the start menu right click "Computer" and select manage. On the left pane select Device Manager. The launcher will be under "Human Interface Devices". Find the right device with the right product id and vendor id.

Right click device > Properties > Details tab > Property: "Hardware Ids"

VID = Vendor ID (1010)
PID = Product ID (2123)

Select the driver tab and update the driver. Locate the driver manually and explicitly pick the driver you made. (It will warn you about the driver not being digitally signed)

Your missile launcher is ready for some Java!

JAVA THUNDER LAUNCHER

Lets start by creating an object to hide all of the grizzly bits from us so we can just send a command and a length of time to execute it in milliseconds.

Create a new Java Application NetBeans Project and import the ch.ntb.usb-0.5.9.jar file.

To control the launcher we are sending commands in the data packet for the control message. To control the motors the first byte is 0x02. For the LED it is 0x03. The second byte is the command and the other 6 bytes are 0x00. For reference the commands we are passing in the data packet are as follows:

DOWN = 0x01

UP = 0x02

LEFT = 0x04

RIGHT = 0x08

FIRE = 0x10

STOP = 0x20

LEDON = 0x01

LEDOFF = 0x00

Thanks to: https://github.com/codedance/Retaliation

650 pages of fun! (Seriously though, look up Control Transfers to get an idea what is going on. Pages, 225-227, 248-250) http://sdphca.ucsd.edu/Lab_Equip_Manuals/usb_20.pdf

The first thing to do is create an object we can use to control the launcher. Here's mine:

```
 package thunderlauncher;

import ch.ntb.usb.Device;
import ch.ntb.usb.USB;
import ch.ntb.usb.USBException;

/**
 *
 * @author James Green
 * @version 1.0.0
 * @website http://coffeefueledcreations.com/
 */
public class Launcher {

    private Device dev = null;
    private final byte[][] COMMANDS = {{0x02, 0x20, 0x00,0x00,0x00,0x00,0x00,0x00},{0x03, 0x01, 0x00,0x00,0
    public enum Command {STOP,LEDON,LEDOFF,UP,DOWN,LEFT,RIGHT,FIRE}

    Launcher(){
        usbSetup();
    }

    /**
     * Resets the turret to a guess at middle position
     */
    private void zero(){ // Reset to bottom left and return to center-ish
            execute(Launcher.Command.DOWN, 3000);
            execute(Launcher.Command.LEFT, 6000);
            execute(Launcher.Command.RIGHT, 3000);
            execute(Launcher.Command.UP, 200);
    }

    /**
     * Open the USB Thunder Launcher
     */
    private void usbSetup(){
        dev = USB.getDevice((short) 0x2123, (short) 0x1010); // Vendor ID, Product ID
        try {
            dev.open(1, 0, -1); // Open the device (Configuration(default), Interface(Control), Alternative
        } catch (USBException ex) {
            System.out.println("Please check the driver for device VID:0x2123, PID:0x1010");
            ex.printStackTrace();
        }
    }

    /**
     *
     * @param c Use the public "Command" enum to control the turret.
     * @param Duration The number of milliseconds to execute the command.
     * The FIRE command takes about 3300 milliseconds.
     */
    public void execute(Command c, long Duration){
        try{
        if(dev.isOpen()){
            switch(c){
                case STOP: // Send a control message
                    dev.controlMsg(0x21, 0x09, 0,0, COMMANDS[0], COMMANDS[0].length, 2000, false);
                    break;
                case LEDON:
                    dev.controlMsg(0x21, 0x09, 0,0, COMMANDS[1], COMMANDS[1].length, 2000, false);
                    break;
                case LEDOFF:
                    dev.controlMsg(0x21, 0x09, 0,0, COMMANDS[2], COMMANDS[2].length, 2000, false);
                    break;
                case UP:
                    dev.controlMsg(0x21, 0x09, 0,0, COMMANDS[3], COMMANDS[3].length, 2000, false);
                    break;
                case DOWN:
                    dev.controlMsg(0x21, 0x09, 0,0, COMMANDS[4], COMMANDS[4].length, 2000, false);
                    break;
                case LEFT:
                    dev.controlMsg(0x21, 0x09, 0,0, COMMANDS[5], COMMANDS[5].length, 2000, false);
                    break;
                case RIGHT:
                    dev.controlMsg(0x21, 0x09, 0,0, COMMANDS[6], COMMANDS[6].length, 2000, false);
                    break;
                case FIRE://The fire command needs about 3300 milis to execute
                    dev.controlMsg(0x21, 0x09, 0,0, COMMANDS[7], COMMANDS[7].length, 2000, false);
                    break;
            }
            long wait = System.currentTimeMillis()+Duration;
            while(wait > System.currentTimeMillis()){
                //Burn Time and let the Launcher execute.
            }
            dev.controlMsg(0x21, 0x09, 0,0, COMMANDS[0], COMMANDS[0].length, 2000, false);
        }
        }catch(Exception e){
            e.printStackTrace();
        }
    }

}
```

As a simple demo, lets attach some keyboard controls.

```
 package thunderlauncher;

import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.KeyEvent;
import java.awt.event.KeyListener;
import javax.swing.JFrame;
import javax.swing.JLabel;

/**
 *
 * @author James Green
 * @version 1.0.0
 * @website http://coffeefueledcreations.com/
 */
public class KeyController extends JFrame implements KeyListener, ActionListener  {
    Launcher l = null;
    KeyController(Launcher l){
        this.l = l;
        javax.swing.SwingUtilities.invokeLater(new Runnable() { //Thread safe GUI initilazation
            public void run() {
                createGUI();
            }
        });
    }

    private void createGUI() {
        this.setName("Key Controller Demo");
        this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        JLabel label = new JLabel();
        label.setText("Use arrow keys to move launcher, space to fire and z,x to toggle led");
        this.add(label);
        this.setSize(410, 200);
        this.setVisible(true);
        this.addKeyListener(this);
    }

    @Override
    public void keyPressed(KeyEvent e) {
        System.out.println(e.getKeyCode());
        switch(e.getKeyCode()){
            case 38:
                l.execute(Launcher.Command.UP, 25);
                break;
            case 40:
                l.execute(Launcher.Command.DOWN, 25);
                break;
            case 37:
                l.execute(Launcher.Command.LEFT, 25);
                break;
            case 39:
                l.execute(Launcher.Command.RIGHT, 25);
                break;
            case 32:
                l.execute(Launcher.Command.FIRE, 3300);
                break;
            case 90:
                l.execute(Launcher.Command.LEDON, 1);
                break;
            case 88:
                l.execute(Launcher.Command.LEDOFF, 1);
                break;
        }
    }

    @Override
    public void keyReleased(KeyEvent e) {

    }

    @Override
    public void actionPerformed(ActionEvent e) {

    }

    @Override
    public void keyTyped(KeyEvent e) {

    }
}
```

And finally this is the main to start it all off.

```
 package thunderlauncher;

/**
 *
 * @author James Green
 * @version 1.0.0
 * @website http://coffeefueledcreations.com/
 */
public class ThunderLauncher {
    public static void main(String[] args) {
        Launcher l = new Launcher();
        KeyController kc = new KeyController(l);
    }
}
```

RESULTS

NEXT PART: FACE DETECTION

[Next, Face Detection]

Note: You can roll back the driver by updating the driver and let windows find it automatically.

ONE RESPONSE TO *USING JAVA LIBUSB WRAPPER TO CONTROL DREAM CHEEKY THUNDER TURRET*

Vinícius Baesso *says:*

21ST JULY 2014 AT 12:08 PM

Hello, I'm from Brazil and I'm trying to do something with libusb and Java, but I have difficulty installing the libusb. I'm using Linux, could you help me? I would be grateful!

REPLY