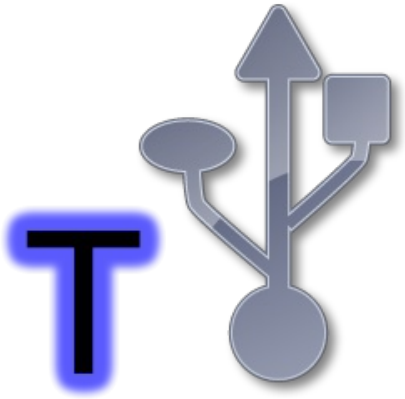
adafruit learning system

Trinket Fake USB Serial

Created by Frank Zhao



Last updated on 2014-09-01 11:45:10 AM EDT

Guide Contents

Overview

The Trinket has a USB port that is used for bootloading. But the Trinket can only become a low-speed USB device because of its limited hardware. USB standards prevents low speed USB devices to truly act as virtual serial ports, which is why we cannot use a serial terminal to communicate with the Trinket directly.

*However***,** there's a work-around for this problem. In Windows, you can emulate a fake serial port bridge using a utility named **com0com**. In this tutorial, you'll see how we can write a middle-man program that communicates with the Trinket and **com0com**, so that a serial terminal (such as the one built into Arduino IDE) can talk with the other end of **com0com.**

The com0com code will only work on Windows computers - but we have some code that *might* work for Mac/Linux, check the last step of this tutorial

This code requires running at 16MHz, so it is only suggested for use with 5V Trinktets - 16MHz is overclocking on 3V and may not work or may be flaky!

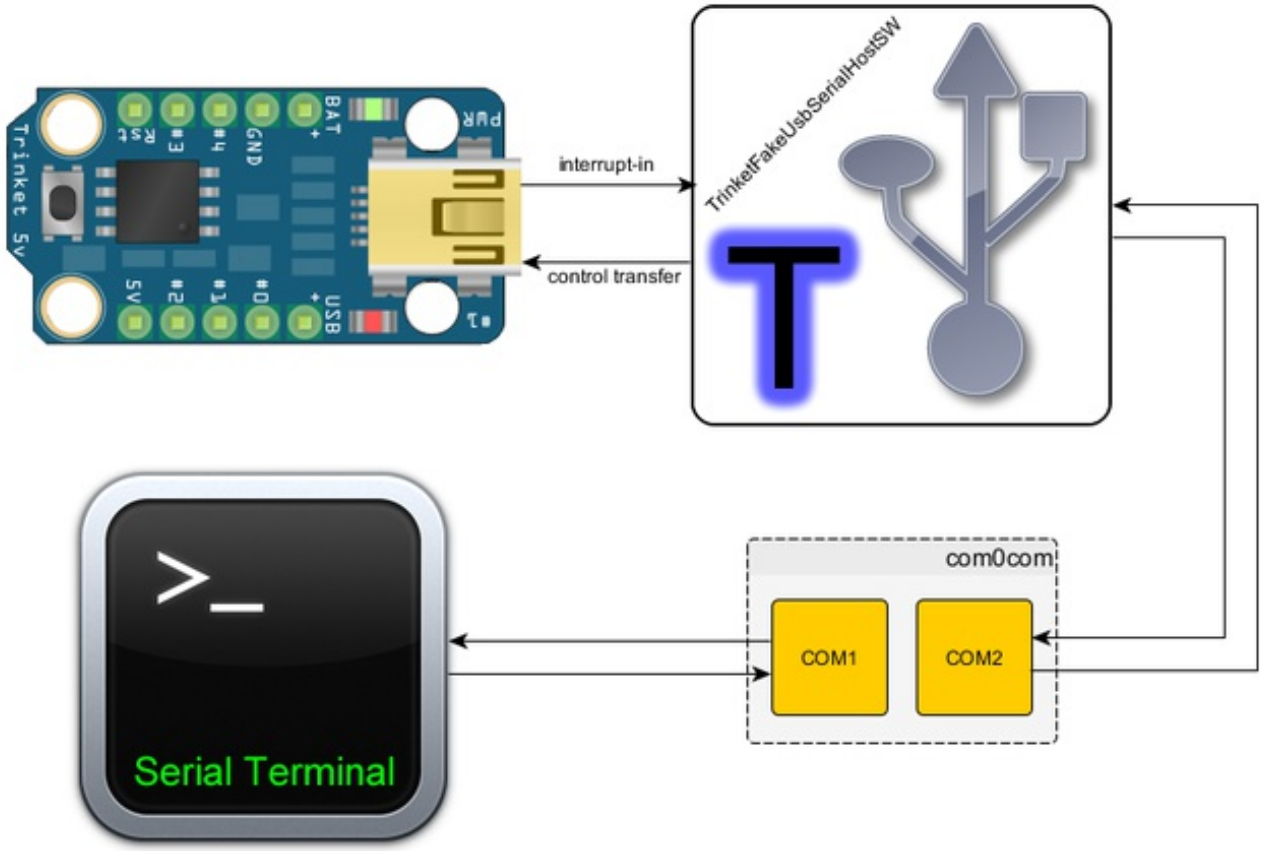
There are two pieces of code involved here:

- Arduino library named **TrinketFakeUsbSerial**
- A PC app named **TrinketFakeUsbSerialHostSW**

The other pieces of software involved

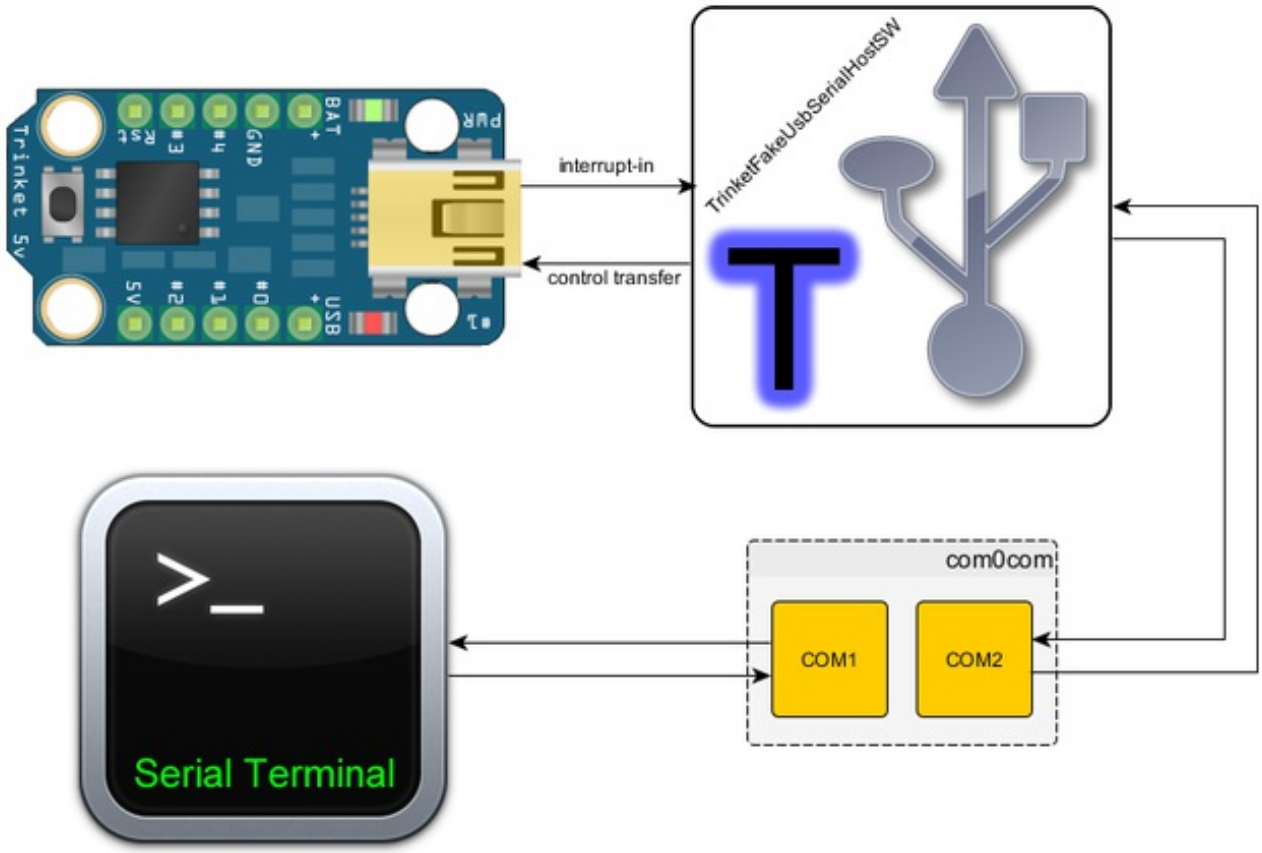
- **LibUsbDotNet**, which makes it easy for me to use libusb-win32 to write TrinketFakeUsbSerialHostSW for Windows
- **com0com**, which emulates the two fake serial ports (only required on Windows)
- Some sort of serial terminal (Arduino IDE, Hyperterminal, Teraterm, RealTerm, Putty, etc)

Below is a diagram showing how data flows between the different components:



Below is a demonstration video (the code sketch is in the Usage Demo section of this tutorial, the video should be viewed in 720p and full screen if you want to read the text on screen)

How it works



Since virtual serial port protocol isn't allowed for low speed devices. We have to find another way to communicate.

First, understand that all communication on a USB bus is always initiated by the host (the computer).

The most basic useful data transaction is a control transfer, which carries two parts: a setup packet and a data packet. Basically, the setup packet says how many bytes are in the data packet, and which way the data packet goes. We use control transfers for data going from the computer to the Trinket.

To get data going from the Trinket to the computer, we open an interrupt-in endpoint to send it. The computer will automatically ask "is there anything in the endpoint" once every 2 ms, and if Trinket answers "yes", then the transfer begins.

The middle-man software I wrote (named **TrinketFakeUsbSerialHostSW**) will communicate with the Trinket using LibUsbDotNet, which is a libusb-win32 wrapper for C#. The data is relayed to a fake serial port created by com0com.

Install for Windows

Everything you need to download is on github, click the button to download a zip

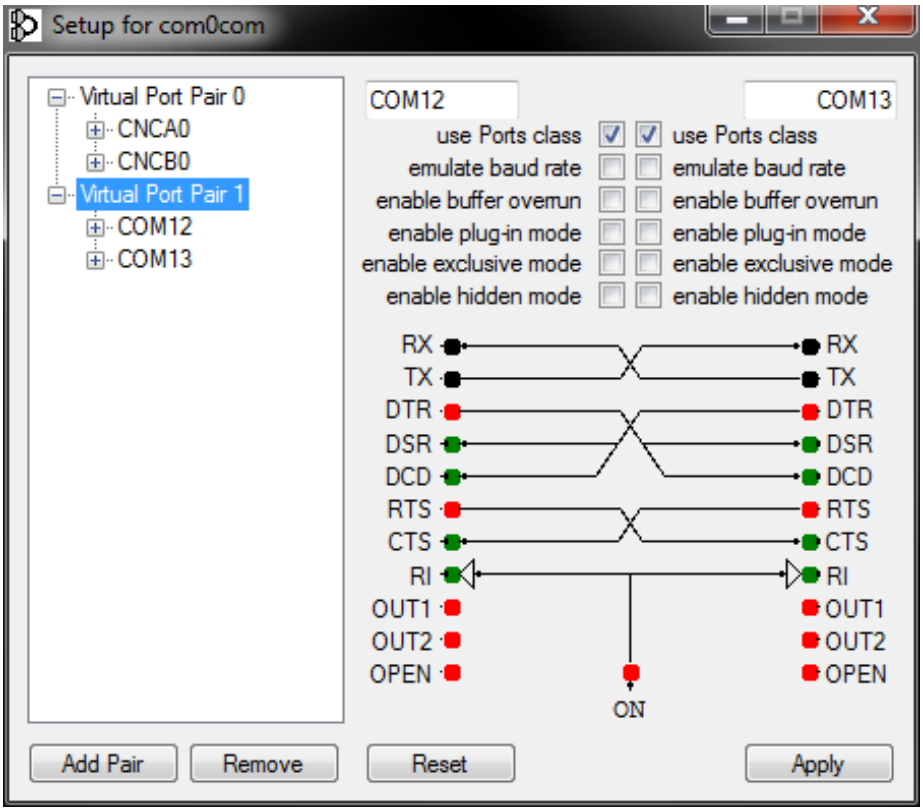
Download the Trinket Fake USB Files

http://adafru.it/cPA

(note: all the source code is available: https://github.com/adafruit/Adafruit-Trinket-USB/ (http://adafru.it/cJL))

Install the Arduino library (http://adafru.it/cJM) as if it was any other Arduino library, read the readme.txt file, the source code, and the example sketch to see how to use it.

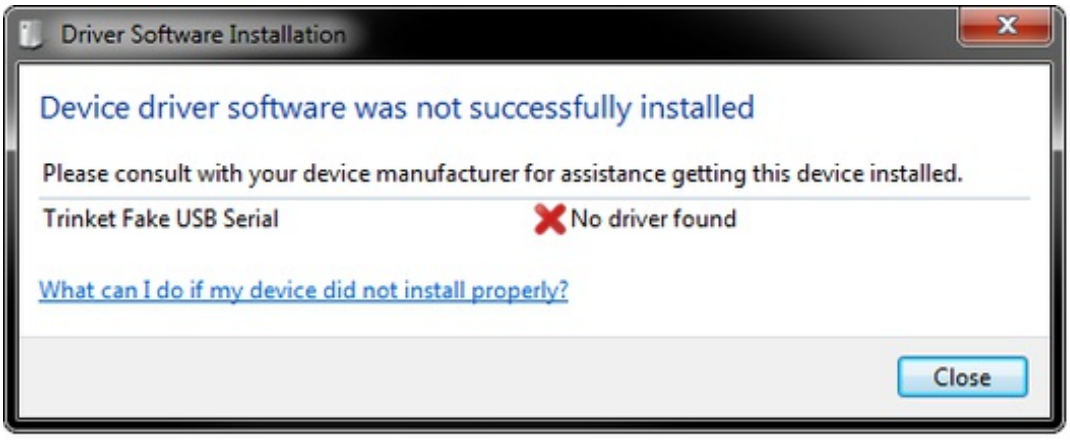
Install com0com (http://adafru.it/cfm), then use it to setup a pair of fake serial ports (see screenshot to see my configuration, you can use any port number you want)

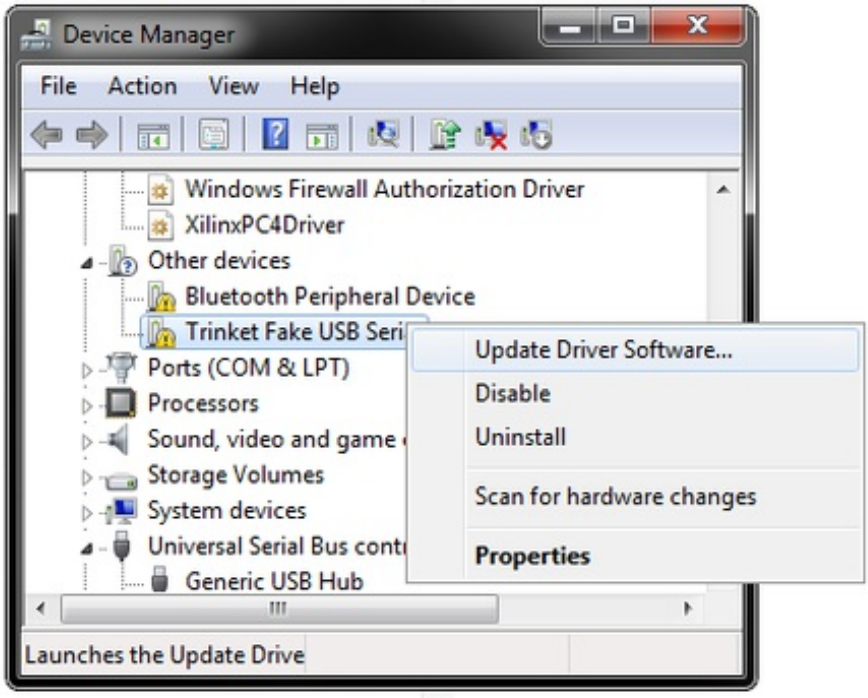


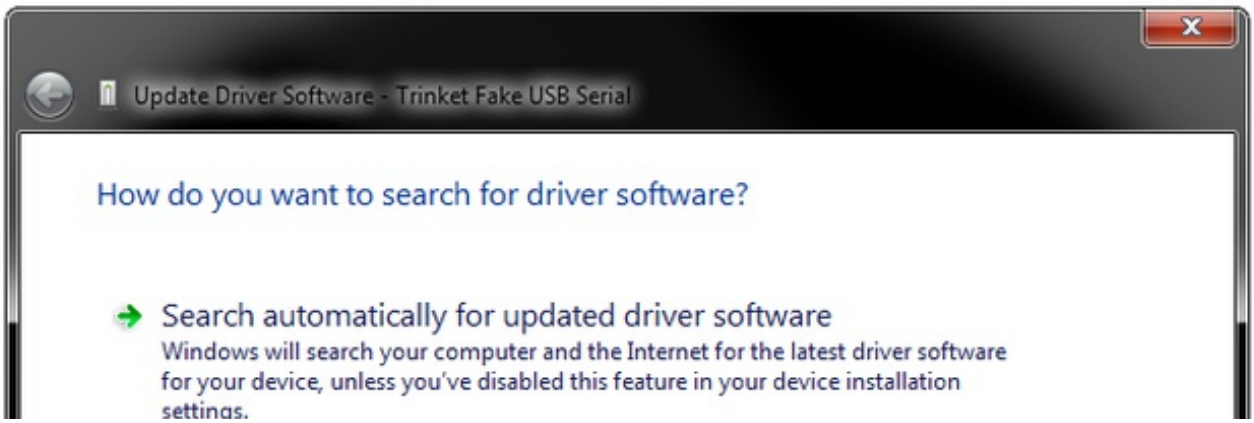
This is the point where you write your own sketches to use the TrinketFakeUsbSerial library in any way you'd like. Get creative!

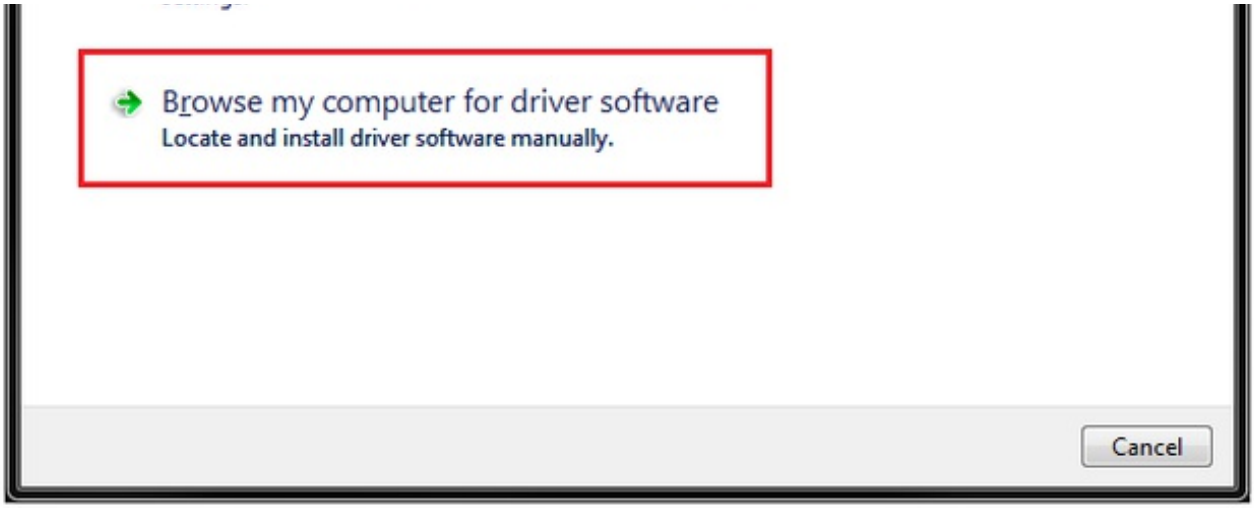
Please read the source code of the library, the readme file, and the example sketch. It will help you understand how to use the library. It works mostly like HardwareSerial, so you can use print and println, along with other inherited functions.

When you try to connect a Trinket loaded with a sketch containing the TrinketFakeUsbSerial library, Windows will fail to find a device driver. The zip file you downloaded contains the device driver you need, so you have to tell Windows where they are. The driver is unsigned so there might be a security warning, install it anyways.

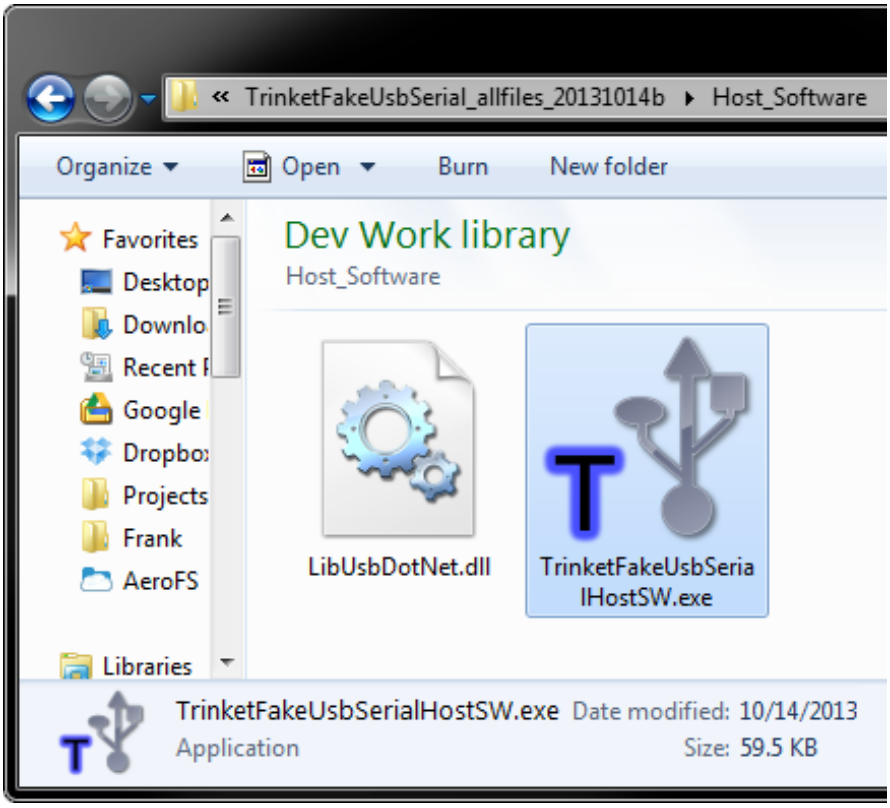


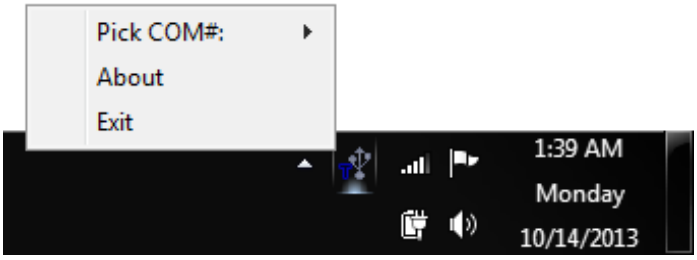






You've also downloaded the executable named TrinketFakeUsbSerialHostSW.exe , when you need to use the serial terminal with Trinket, this executable must be running in the background.





Remember, if you use com0com to setup COM12 and COM13, then use TrinketFakeUsbSerialHostSW to open COM12, and use your serial terminal to open COM13. The com0comsoftware will create a data-pipe between the two different # ports.

Don't try to open COM12 using both, it won't work. Don't try to open COM13 using both, it won't work.

(You don't have to use 12 or 13, you can really use any unused number)

Usage Demo

The video above should be viewed in 720p and full screen if you want to read the text on screen

The sketch used in the demo is posted below

```
#include "TrinketFakeUsbSerial.h"

void setup()
{
  TFUSerial.begin();
}

void loop()
{
  TFUSerial.task(); // this should be called at least once every 10 ms

  if (TFUSerial.available()) {
    char c = TFUSerial.read();
    if (c == 'a')
    {
      TFUSerial.println("hello world");
    }
    else if (c == 'b')
    {
      TFUSerial.println(millis(), DEC);
    }
  }
}
```

Install for Linux / Mac

I managed to write a Python script that does the same thing as the Windows application, except it's not a system tray application. It needs to be launched through the command line or using a script.

Since it's a Python script that uses cross platform modules, the whole script is also cross platform and thus it works on Linux and Mac OS X.

> I've tested this on Windows, and I am only assuming it will work on Linux and Mac OS X at this time. I will test more when I am able to.

First you need these items installed:

- Python 2.7 (http://adafru.it/cPB) (Python 3 will not work)
- PyUSB (http://adafru.it/cPC)
- pySerial (http://adafru.it/cLU)

(note: you can install the two modules using easy_install (http://adafru.it/cPD) , if you are having trouble installing these, please check Google to find some Python tutorials on installing modules!)

Second, to replicate the role of com0com, use this command:

```
socat PTY,link=COM8 PTY,link=COM9
```

to create COM8 and COM9, use any number you want

Finally, to run the Python script...

The download package is located: https://github.com/adafruit/Adafruit-Trinket-USB/blob/master/TrinketFakeUsbSerial_allfiles_20131014... (http://adafru.it/cPA) , the py file is inside the zip archive.

Using your command line terminal, navigate to where the py file is located, and then run this command:

```
python TrinketFakeUsbSerialHostSW.py -p <portname>
```

The script should then run forever. It should gracefully handle connections and disconnections for you, but errors will show up on the screen.

To view more command line options (options for silencing errors, or show more detailed

connection status) for the script, run this command:

```
python TrinketFakeUsbSerialHostSW.py -h
```

USB HID Terminal Alternative

Another way of communicating with the Trinket uses HID to pass raw USB messages back and forth. This technique was created by Ray's Hobby. The key difference between Ray's method and the method discussed above is:

- Ray's method uses a HID profile, thus no driver installations are required at all
- Compared to the Fake USB method uses a custom profile with a libusb driver
- Ray's method uses a customized serial terminal that he wrote himself
- The Fake USB method works with most generic serial terminal programs, but requires running a background application to act as a "bridge"

For a lot more information check out http://rayshobby.net/?p=7363 (http://adafru.it/cV6) where the technique is talked about in detail

Ray's code is not specifically designed for Trinket, so to make it work with Trinket, I have forked Ray's github for this project and made modifications to his code so that it also works with the Trinket.

Visit the forked and modified github repo

http://adafru.it/dVj

You can download all the files from the github repo above.

The only changes are:

- changed usbconfig.h to include different pin assignments
- added the oscillator calibration function required for ATtiny
- adjusted clock speed for Trinket

I did not modify Ray's host software at all. It works with his software without any modifications. His software is written using Processing.org (http://adafru.it/cV8), and while the executables are also included, you should have Java (http://adafru.it/cV9) installed in order to use them. The host software should work on any platform that can run Processing.org applications (or any platform that can run Java, meaning Windows, Linux, and Mac should all work).

When you have downloaded all the files, install the Arduino library as usual (http://adafru.it/cJM). You can then try compiling your own sketches using the library, or try running one of the provided example sketches.

Then to use the serial terminal, either run the host software that you've already downloaded. You may use Processing.org to open the host software, or use one of the provided precompiled executables specific to your operating system.

Ray's method must use his own serial terminal, it will not work with Arduino IDE's own serial terminal, it will not work with Hyperterminal or TeraTerm or anything else. It will only work with Ray's own serial terminal application.