view source

print?

```
#include <iostream>
#include <libusb.h>
using namespace std;

void printdev(libusb_device *dev); //prototype of the function

int main() {
    libusb_device **devs; //pointer to pointer of device, used to retrieve a list of devices
    libusb_context *ctx = NULL; //a libusb session
    int r; //for return values
    ssize_t cnt; //holding number of devices in list
    r = libusb_init(&ctx); //initialize a library session
    if(r < 0) {
        cout<<"Init Error "<<r<<endl; //there was an error
                return 1;
    }
    libusb_set_debug(ctx, 3); //set verbosity level to 3, as suggested in the documentation
    cnt = libusb_get_device_list(ctx, &devs); //get the list of devices
    if(cnt < 0) {
        cout<<"Get Device Error"<<endl; //there was an error
    }
    cout<<cnt<<" Devices in list."<<endl; //print total number of usb devices
        ssize_t i; //for iterating through the list
    for(i = 0; i < cnt; i++) {
                printdev(devs[i]); //print specs of this device
        }
        libusb_free_device_list(devs, 1); //free the list, unref the devices in it
        libusb_exit(ctx); //close the session
        return 0;
}

void printdev(libusb_device *dev) {
    libusb_device_descriptor desc;
    int r = libusb_get_device_descriptor(dev, &desc);
    if (r < 0) {
        cout<<"failed to get device descriptor"<<endl;
        return;
    }
    cout<<"Number of possible configurations: "<<(int)desc.bNumConfigurations<<"  ";
    cout<<"Device Class: "<<(int)desc.bDeviceClass<<"  ";
    cout<<"VendorID: "<<desc.idVendor<<"  ";
    cout<<"ProductID: "<<desc.idProduct<<endl;
    libusb_config_descriptor *config;
    libusb_get_config_descriptor(dev, 0, &config);
    cout<<"Interfaces: "<<(int)config->bNumInterfaces<<" ||| ";
    const libusb_interface *inter;
    const libusb_interface_descriptor *interdesc;
    const libusb_endpoint_descriptor *epdesc;
    for(int i=0; i<(int)config->bNumInterfaces; i++) {
        inter = &config->interface[i];
        cout<<"Number of alternate settings: "<<inter->num_altsetting<<" | ";
        for(int j=0; j<inter->num_altsetting; j++) {
            interdesc = &inter->altsetting[j];
            cout<<"Interface Number: "<<(int)interdesc->bInterfaceNumber<<" | ";
            cout<<"Number of endpoints: "<<(int)interdesc->bNumEndpoints<<" | ";
            for(int k=0; k<(int)interdesc->bNumEndpoints; k++) {
                epdesc = &interdesc->endpoint[k];
                cout<<"Descriptor Type: "<<(int)epdesc->bDescriptorType<<" | ";
                cout<<"EP Address: "<<(int)epdesc->bEndpointAddress<<" | ";
            }
        }
    }
    cout<<endl<<endl<<endl;
    libusb_free_config_descriptor(config);
}
```