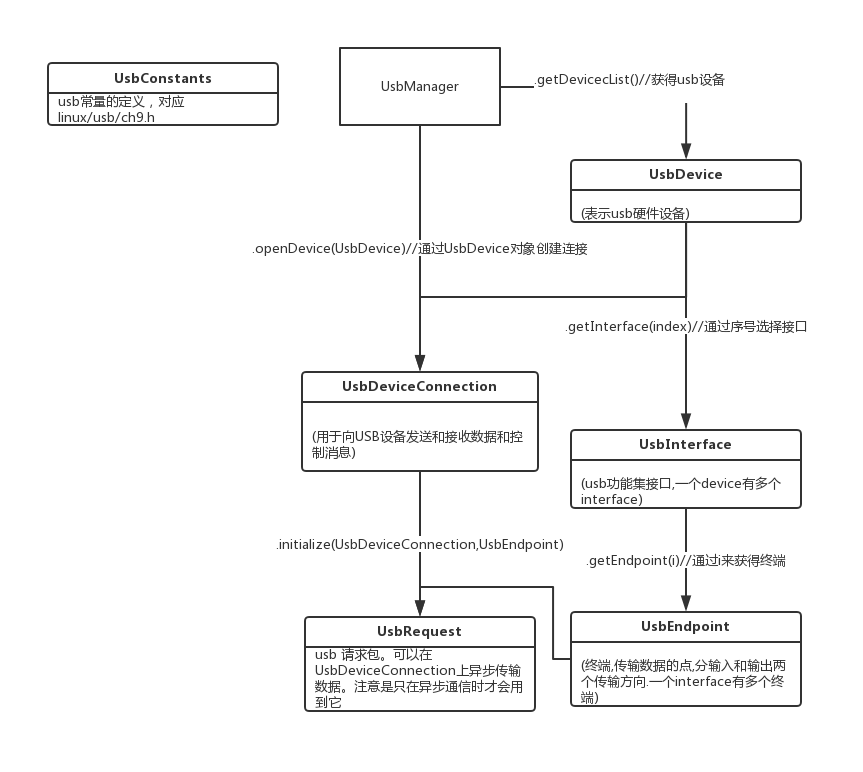
Usb简单关系图



备注:

安卓UsbInterface和UsbEndpoint是什么?

这都来源于usb协议,比如手机的usb,它可用仅充电,也可用作为U盘传输,也可以光连接摄像头,这是3种不同的功能,需要对该usb硬件进行3中不同的配置才能达到,而每种功能,就对应一种interface(接口).

每种功能,需要Endpoint来进行数据传递,要么只能input,要么只能output,但是也有一个特殊的endpoint 0,每个USB设备都有这么一个双工端点,其余单工端点,包含iso、bulk、int等端点来传输大量的数据。

使用步骤

1/监听usb插入广播

USBReceiver registerReceiver = new USBReceiver();  
 public class USBReceiver extends BroadcastReceiver {  
 //自建的授权监听  
 public static final String *ACTION\_USB\_PERMISSION* = "com.android.example.USB\_PERMISSION";  
  
 @RequiresApi(api = Build.VERSION\_CODES.*LOLLIPOP*)  
 @Override  
 public void onReceive(Context context, Intent intent) {  
 String action = intent.getAction();  
 if (*ACTION\_USB\_PERMISSION*.equals(action)) {  
 // 获取权限结果的广播  
 UsbDevice device = (UsbDevice) intent.getParcelableExtra(UsbManager.*EXTRA\_DEVICE*);  
 synchronized (this) {  
 if (device != null) {  
 if (intent.getBooleanExtra(UsbManager.*EXTRA\_PERMISSION\_GRANTED*, false)) {  
 usbDevice = device;//申请usb权限成功  
 Log.*d*("USBReceiver", "获取权限成功：" + device.getDeviceName());  
 } else {  
 usbDevice = null;  
 Log.*d*("USBReceiver", "获取权限失败：" + device.getDeviceName());  
 }  
 }  
 }  
 } else if (UsbManager.*ACTION\_USB\_DEVICE\_ATTACHED*.equals(action)) {  
 // 有新的设备插入了，在这里一般会判断这个设备是不是我们想要的，是的话就去请求权限  
 UsbDevice device = (UsbDevice)intent.getParcelableExtra(UsbManager.*EXTRA\_DEVICE*);  
 Log.*d*("USBReceiver", "\*\*\*\*getDeviceName：" + device.getDeviceName());  
 Log.*d*("USBReceiver", "\*\*\*\*getVendorId：" + device.getVendorId());  
 Log.*d*("USBReceiver", "\*\*\*\*getProductId：" + device.getProductId());  
// if(device==(usbDevice=getUsbDevice(4021,24577));//查找特定usb  
 usbDevice = device;//将新插入的设备赋到全局,等会去申请usb权限  
 requestPermission(usbManager,usbDevice);//申请权限  
 } else if (UsbManager.*ACTION\_USB\_DEVICE\_DETACHED*.equals(action)) {  
 usbDevice = null;  
 // 有设备拔出了  
 }  
 }  
 }

2/获取UsbManager

usbManager = (UsbManager) context.getSystemService(Context.USB\_SERVICE);

3/获取 USB 设备列表

HashMap<String, UsbDevice> deviceList = usbManager.getDeviceList();

4/获取特定厂商的usb

/\*\*

\* mVendorId=1137,mProductId=85 佳博 3150T 标签打印机

\*

\* @param vendorId 厂商ID

\* @param productId 产品ID

\* @return device

\*/

public UsbDevice getUsbDevice(int vendorId, int productId) {

HashMap<String, UsbDevice> deviceList = usbManager.getDeviceList();

Iterator<UsbDevice> deviceIterator = deviceList.values().iterator();

while (deviceIterator.hasNext()) {

UsbDevice device = deviceIterator.next();

if (device.getVendorId() == vendorId && device.getProductId() == productId) {

Log.e("USBUtil", "getDeviceList: " + device.getDeviceName());

return device;

}

}

Toast.makeText(context, "没有对应的设备", Toast.LENGTH\_SHORT).show();

return null;

}

5/申请 USB 设备使用权限

/\*\*

\* 请求获取指定 USB 设备的权限

\*/

public void requestPermission(UsbDevice device) {

if (device != null) {

if (usbManager.hasPermission(device)) {//判断是否已经授权

Toast.makeText(context, "已经获取到权限", Toast.LENGTH\_SHORT).show();

} else {

if (mPermissionIntent != null) {

usbManager.requestPermission(device, mPermissionIntent);

Toast.makeText(context, "请求USB权限", Toast.LENGTH\_SHORT).show();

} else {

Toast.makeText(context, "请注册USB广播", Toast.LENGTH\_LONG).show();

}

}

}

}

6/通信,创建一个新线程来执行所有数据传输，避免阻塞UI线程。

找到了USB设备，并且处理了权限问题，剩下的就是打开USB设备的通信通道，进而可以读取数据。首先我们需要获取设备的UsbInterface，一个USb设备有多个UsbInterface，我们需要的是第一个。一个UsbInterface有多个端口，我们需要的是处理数据的端口，通过遍历可以获取到数据输入端口usbEndpointIn和输出端口usbEndpointOut。这两个端口就是我们进行数据读取的时候用到的。最后usbManager.openDevice(device)打开端口。

读写操作，均使用usbManager.bulkTransfer()方法往USB通道里读写数据，

参数1：输入输出端口。

参数2：发送的字节数组。

参数3：字节数组的长度。

参数4：超时时间。

返回值ret就是发送成功以后返回实际发送的字节数，可以当做发送成功与否的标志。接收的时候定义的字节数组应该比设备一次返回的字节数组大，查阅的资料是最少应该比设备返回的字节数大2个字节，比如设备返回的字节是480，那定义的数组长度应该是>480。接收数据开线程，循环接收，有数据随时处理。

public boolean openPort(UsbDevice device) {

//获取设备接口，一般只有一个，多个的自己研究去

usbInterface = device.getInterface(0);

// 判断是否有权限

if (hasPermission(device)) {

// 打开设备，获取 UsbDeviceConnection 对象，连接设备，用于后面的通讯

usbConnection = usbManager.openDevice(device);

if (usbConnection == null) {

return false;

}

if (usbConnection.claimInterface(usbInterface, true)) {//独占该接口,用于后续数据传输

} else {

usbConnection.close();

return false;

}

} else {

Toast.makeText(Utils.getContext(), "没有 USB 权限", Toast.LENGTH\_SHORT).show();

return false;

}

//获取接口上的两个端点，分别对应 OUT 和 IN,但实际硬件不一定是0和1为输入输出

for (int i = 0; i < usbInterface.getEndpointCount(); ++i) {

UsbEndpoint end = usbInterface.getEndpoint(i);

if (end.getDirection() == UsbConstants.USB\_DIR\_IN) {

usbEndpointIn = end;

} else {

usbEndpointOut = end;

}

}

return true;

}

7/发送数据

usbConnection.bulkTransfer(usbEndpointOut, bytes, bytes.length, 500);

//bytes表数据,500是超时时间,超时将返回负值

踩坑:

1.有次项目上使用usb扫条形码模块symcode,在PC上能正常使用,但安卓平板不行.ttyACM0设备一直未发现,发现是代码中usbConnection.claimInterface(usbInterface, true)中置true导致的,从安卓API上看,置true,代表强制占用设备,这会引起设备自动卸载再重载.所以,这是为什么dev下没有ttyACM设备的原因.但即使能看见ttyACM,依然有坑,一般ttyACM是作为大文件传输使用,默认的驱动是在数据传输完成后会断开连接,这样,usb会经历一个虚拟的拔出和插入的过程.这对扫条形码这样的小数据,可能在设备断开连接后,就无法回传数据了,这得改驱动,让acm长连接