```swift
//
//  ViewController.swift
//  RGB USB
//
//  Created by Erik Nordlund on 4/30/19.
//  Copyright © 2019 Erik Nordlund. All rights reserved.
//
//  Arm USB includes the following open-source components:
//      • peertalk-simple: https://github.com/kirankunigiri/peertalk-simple
//      • ORSSerialPort: https://github.com/armadsen/ORSSerialPort

import Cocoa


class ViewController: NSViewController, USBSerialDelegate {

    func messageWasReceived(_ message: String) {
        let printStatement = "message received: " + message
        //let attributedString = NSAttributedString(string: printStatement +
         "\n")
        debugPrint(textView.textColor)

        if var newString = textView.textStorage?.string {
            newString.append(printStatement + "\n")
            textView.textStorage?.mutableString.setString(newString)
            textView.scrollToEndOfDocument(self)

            debugPrint("sending via peertalk: ", message)
            ptManager.sendObject(object: message, type: PTType.string.rawValue)
        }

        print(printStatement)

    }

    @IBOutlet var textView: NSTextView!
    @IBOutlet weak var textField: NSTextField!
    @IBOutlet weak var sendButton: NSButton!

    let ptManager = PTManager.instance

    override func viewDidLoad() {
        super.viewDidLoad()

        // Do any additional setup after loading the view.
        //serialPort!.baudRate = 9600
        //serialPort!.open()
        //print("hi")

        // Setup the PTManager
        ptManager.delegate = self
```

```
    ptManager.connect(portNumber: PORT_NUMBER)

    usbSerial = USBSerial(delegate: self)
    debugPrint("(viewcontroller)")
    debugPrint(usbSerial.serialPort)
    textView.textColor = NSColor.textColor
}

override var representedObject: Any? {
    didSet {
    // Update the view, if already loaded.
    }
}


@IBAction func textFieldReturn(_ sender: Any) {

    do {
        try sendMessage()
    } catch {
        debugPrint(error)
    }
}
@IBAction func sendCommand(_ sender: Any) {

    do {
        try sendMessage()
    } catch {
        debugPrint(error)
    }


}

enum USBMessageError: Error {
    case sendError
}

func sendMessage() throws {


    let message = textField.stringValue

    if message.count > 0 {
        print("string = ", message)

        if var newString = textView.textStorage?.string {
            let printStatement = "message sent: " + message
            newString.append(printStatement + "\n")
            textView.textStorage?.mutableString.setString(newString)
            textView.scrollToEndOfDocument(self)
```

```
            }

            usbSerial.send(message: message)
        } else {
            throw USBMessageError.sendError
        }


    }

    func forwardToUSBSerial(message: String) throws {
        debugPrint("Trying to forward message: ", message)
        if message.count > 0 {
            print("string = ", message)

            if var newString = textView.textStorage?.string {
                let printStatement = "message sent: " + message
                newString.append(printStatement + "\n")
                textView.textStorage?.mutableString.setString(newString)
                textView.scrollToEndOfDocument(self)
            }

            usbSerial.send(message: message)
        } else {
            throw USBMessageError.sendError
        }
    }

    func textViewScrollToBottom() {
        if let textViewString = textView.textStorage?.string {
            let range = NSMakeRange(NSString(string: textViewString).length -
             1, 1)
            textView.scrollRangeToVisible(range)
        }

    }

}


extension ViewController: PTManagerDelegate {

    func peertalk(shouldAcceptDataOfType type: UInt32) -> Bool {
        return true
    }

    func peertalk(didReceiveData data: Data, ofType type: UInt32) {
        if let message = String(data: data, encoding: .utf8) {
            do {
                try forwardToUSBSerial(message: message)
            } catch {
```

```
                debugPrint("ERROR: Failed to forward message")
            }
        } else {
            debugPrint("ERROR: Failed to convert data to String")
        }
        /*
        if type == PTType.number.rawValue {
            let count = data.convert() as! Int
            //self.label.stringValue = "\(count)"
        } else if type == PTType.image.rawValue {
            let image = NSImage(data: data)
            //self.imageView.image = image
        } else if type == PTType.string.rawValue {
            if let string = String(data: data, encoding: .utf8) {
                do {
                    try forward(message: string)
                } catch {
                    debugPrint("ERROR: Failed to forward message")
                }
            } else {
                debugPrint("ERROR: Failed to convert data to String")
            }

        }
        */
    }

    func peertalk(didChangeConnection connected: Bool) {
        print("Connection: \(connected)")
        //self.statusLabel.stringValue = connected ? "Connected" :
         "Disconnected"
        if connected {

        } else {
            let disconnectionMessage = "d!"

            if var newString = textView.textStorage?.string {
                let printStatement = "message sent: " + disconnectionMessage
                newString.append(printStatement + "\n")
                textView.textStorage?.mutableString.setString(newString)
                textView.scrollToEndOfDocument(self)
            }

            usbSerial.send(message: disconnectionMessage)
        }
    }

}
```