SIoT Documentation

发布 *1.0*

VVlink

2021 年 03 月 25 日

Contents

CHAPTER 1

简介

这一部分主要介绍物联网、MQTT 原理和应用，以及 SIoT 软件。

1.1 物联网简介

物联网（Internet of Things，简称 IoT）是借助互联网、传统电信网等信息承载体，让所有能行使独立功能的普通物体实现互联互通的网络。在物联网上，每个人都可以应用电子标签将真实的物体上网联结，在物联网上都可以查出它们的具体位置。通过物联网可以用中心计算机对机器、设备、人员进行集中管理、控制，也可以对家庭设备、汽车进行遥控，以及搜索位置、防止物品被盗等，类似自动化操控系统，同时通过收集这些小数据，最后聚集成大数据，包含重新设计道路以减少车祸、都市更新、灾害预测与犯罪防治、流行病控制等等社会的重大改变，实现物和物相联。

物联网将现实世界数字化，应用范围十分广泛。物联网拉近分散的信息，统整物与物的数字信息，物联网的应用领域主要包括以下方面：运输和物流领域、工业制造、健康医疗领域范围、智能环境（家庭、办公、工厂）领域、个人和社会领域等，具有十分广阔的市场和应用前景。

1.2 MQTT 简介

常用的物联网应用层协议包括 MQTT、HTTP、XMPP、CoAP 等。

MQTT（Message Queuing Telemetry Transport，消息队列遥测传输协议），是一种基于发布/订阅（Publish/Subscribe）模式的”轻量级”通讯协议，该协议构建于 TCP/IP 协议上，由 IBM 在 1999 年发布。MQTT 最大优点在于，可以以极少的代码和有限的带宽，为连接远程设备提供实时可靠的消息服务。作为一种低开销、低带宽占用的即时通讯协议，使其在物联网、小型设备、移动应用等方面有较广泛的应用。

MQTT 是一个基于客户端-服务器的消息发布/订阅传输协议。MQTT 协议是轻量、简单、开放和易于实现的，这些特点使它适用范围非常广泛。在很多情况下，包括受限的环境中，如：机器与机器（M2M）通信和物联网（IoT）。从当前物联网应用发展趋势来分析，MQTT 协议具有一定的优势。目前国内外主要的云计算服务商，比如阿里云、AWS、百度云、Azure 以及腾讯云都支持 MQTT 协议。

1.2.1 MQTT 协议实现方式

MQTT 系统由与服务器通信的客户端组成，通常称服务器为“代理 Broker”。客户可以是信息发布者 Publish 或订阅者 Subscribe。每个客户端都可以连接到代理。信息按主题层次结构组织。当发布者具有要分发的新数据时，它会将包含数据的控制消息发送到连接的代理（服务器）。然后，代理将信息分发给已订阅该主题的任何客户端。发布者不需要有关于订阅者数量或位置的任何数据，而订阅者又不必配置有关发布者的任何数据。

MQTT 传输的消息分为：Topic 和 payload 两部分：

（1）Topic，可以理解为消息的类型，订阅者订阅（Subscribe）后，就会收到该主题的消息内容（Payload）；

（2）payload，可以理解为消息的内容，是指订阅者具体要使用的内容。

1.2.2 MQTT 协议的核心名词

1. 订阅（Subscription）

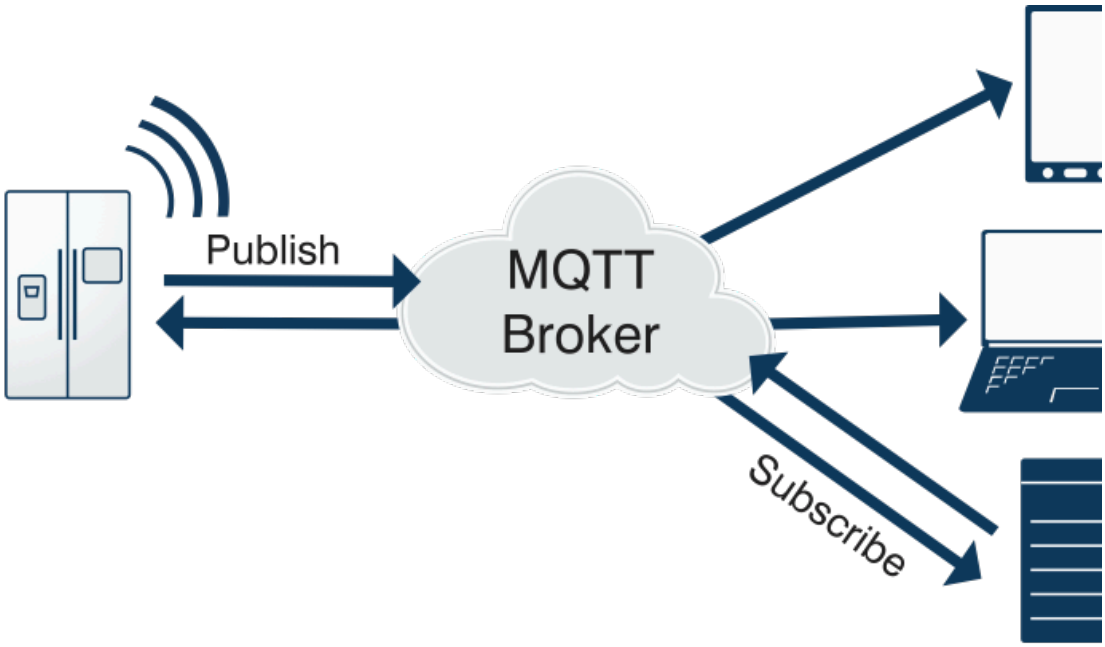
订阅包含主题筛选器（Topic Filter）和最大服务质量（QoS）。订阅会与一个会话（Session）关联。一个会话可以包含多个订阅。每一个会话中的每个订阅都有一个不同的主题筛选器。

2. 会话（Session）

每个客户端与服务器建立连接后就是一个会话，客户端和服务器之间有状态交互。会话存在于一个网络之间，也可能在客户端和服务器之间跨越多个连续的网络连接。

3. 主题名（Topic Name）

连接到一个应用程序消息的标签，该标签与服务器的订阅相匹配。服务器会将消息发送给订阅所匹配标签的每个客户端。



1.2.3 MQTT 协议相关知识

1.QoS 级别

QoS (Quality of Service) 是发送者和接收者之间，对于消息传递的可靠程度的协商。

QoS 的设计是 MQTT 协议里的重点。作为专为物联网场景设计的协议，MQTT 的运行场景不仅仅是 PC，而是更广泛的窄带宽网络和低功耗设备，如果能在协议层解决传输质量的问题，将为物联网应用的开发提供极大便利。

在 MQTT 协议里，定义了三个级别的 QoS，由低到高分别是:

最多一次 (QoS0)

至少一次 (QoS1)

有且仅有一次 (QoS2)

QoS0 是最低级别，基本上等同于 Fire and Forget 模式，发送者发送完数据之后，不关心消息是否已经投递到了接收者那边。

QoS1 是中间级别，保证消息至少送达一次。MQTT 通过简单的 ACK 机制来保证 QoS1。

QoS2 是最高级别，保证到且仅到一次。这通过更加复杂的消息流程保证。

QoS 级别越高，流程越复杂，系统资源消耗越大。应用程序可以根据自己的网络场景和业务需求，选择合适的 QoS 级别:

比如在同一个子网内部的服务间的消息交互往往选用 QoS0；而通过互联网的实时消息通信往往选用 QoS1；QoS2 使用的场景相对少一些，能想到的如国防武器，医疗设备等应用场景。

2. 连接保活心跳机制（Keep Alive Timer）

MQTT 客户端可以设置一个心跳间隔时间（Keep Alive Timer），表示在每个心跳间隔时间内发送一条消息。如果在这个时间周期内，没有业务数据相关的消息，客户端会发一个 PINGREQ 消息，相应的，服务器会返回一个 PINGRESP 消息进行确认。如果服务器在一个半（1.5）心跳间隔时间周期内没有收到来自客户端的消息，就会断开与客户端的连接。心跳间隔时间最大值大约可以设置为 18 个小时，0 值意味着客户端不断开。

3. 遗言标志（Will Flag）

在可变报文头的连接标志位字段（Connect Flags）里有三个 Will 标志位：Will Flag、Will QoS 和 Will Retain Flag，这些 Will 字段用于监控客户端与服务器之间的连接状况。如果设置了 Will Flag，就必须设置 Will QoS 和 Will Retain 标志位，消息主体中也必须有 Will Topic 和 Will Message 字段。

那遗愿消息是怎么回事呢？服务器与客户端通信时，当遇到异常或客户端心跳超时的情况，MQTT 服务器会替客户端发布一个 Will 消息。当然如果服务器收到来自客户端的 DISCONNECT 消息，则不会触发 Will 消息的发送。因此，Will 字段可以应用于设备掉线后需要通知用户的场景。

4. 标注位（RETAIN）

当我们使用 MQTT 客户端发布消息（Publish）时，如果将 RETAIN 标志位设置为 true，那么 MQTT 服务器会将最近收到的一条 RETAIN 标志位为 true 的消息保存在服务器端（内存或文件）。

特别注意：

MQTT 服务器只会为每一个 Topic 保存最近收到的一条 RETAIN 标志位为 true 的消息。也就是说，如果 MQTT 服务器上已经为某个 Topic 保存了一条 Retained 消息，当客户端再次发布一条新的 Retained 消息，那么服务器上原来的那条消息会被覆盖！

每当 MQTT 客户端连接到 MQTT 服务器并订阅了某个 Topic，如果该 Topic 下有 Retained 消息，那么 MQTT 服务器会立即向客户端推送该条 Retained 消息。

1.3 SIoT 简介

SIoT 为一个为教育定制的跨平台的开源 MQTT 服务器程序，S 指科学（Science）、简单（simple）的意思。SIoT 支持 Win10、Win7、Mac、Linux 等操作系统，一键启动，无需用户注册或者系统设置即可使用。

SIoT 为“虚谷物联”项目的核心软件，是为了帮助中小学生理解物联网原理，并且能够基于物联网技术开发各种创意应用。因为其重点关注物联网数据的收集和导出，是采集科学数据的最好选择之一。

SIoT 采用 GO 语言编写，具有如下特点：

- 跨平台。支持 Win10、Win7、Mac、Linux 等操作系统。只要启动这一程序，普通计算机（包括拿铁熊猫、虚谷号和树莓派等微型计算机）就可以成为标准的 MQTT 服务器。
- 一键运行。纯绿色软件，不需要安装，下载后解压就可以使用，对中小学的物联网技术教学尤其适合。
- 使用简单。软件运行后，不需要任何设置就可以使用。利用特定的“Topic”的名称（“项目名称/设备名称”），就能自动在数据库中添加项目和设备的名称，并将消息数据存入数据库。

- 支持数据导出。所有的物联网消息数据都可以在线导出，系统采用 SQLite 数据库，同时支持 Mysql 数据库。
- 支持标准的 MQTT 协议。QoS 级别为 0。
- 支持 WebAPI。系统系统了完善的 WebAPI，方便各种软件以 HTTP 的方式调用，支持 App inventor、Scratch、VB 等默认不支持 MQTT 的中小学生常用编程软件调用。
- 支持插件开发。

SIoT 的资源 GitHub：https://github.com/vvlink/SIoT/ 提供文档、案例、课程

1.3.1 SIoT 软件开发团队

- 核心人员：苏宇、谢作如、夏青
- 技术支持：张路、叶琛、李冬冬
- 系统测试：李亮、林淼焱、张喻

注：SIoT 软件的开发得到温州市科技局 2019 年科技创新项目的资助，为《物联网与科学探究创意实验课程开发》项目的成果之一，软件采用 MIT 协议开源，本文档采用 CC-BY-SA 协议开源。

1.3.2 SIoT 文档开发团队

- 负责人：谢作如
- 参与人员：
  - 谢作如（温州中学）
  - 林淼焱（温州大学）
  - 郑祥（温州四中）
  - 郝晴（天津师大）
  - 张喻（温州大学）
  - 邱奕盛（温州中学）
  - 许靖宇（天津师大）
  - 宋达（天津师大）
  - 毛雁（天津师大）
  - 夏青（上海蘑菇云）
  - 陆雅楠（上海师范大学）
  - ……

1.3.3 SIoT 下载地址

https://github.com/vvlink/SIoT/

最新版本为 1.3。

CHAPTER 2

安装和运行

这一部分主要介绍 SIoT 软件的下载、安装、运行以及软件操作界面。

2.1 文件下载

软件下载地址（项目主页）：

http://mindplus.dfrobot.com.cn/siot

软件下载地址（项目文档）：

https://github.com/vvlink/SIoT/

最新软件版本：SIoT v1.3

直达地址：https://github.com/vvlink/SIoT/tree/master/software

注： 虚谷号上已经预装了 SIoT，插电后连上 Wi-Fi 即可使用。

2.2 安装运行

SIoT 是一个绿色软件，将下载的压缩包解压并打开后，你能看到多个文件，如：

- SIoT1.3_win.exe（Win10、Win7 版本执行文件）
- SIoT1.3_mac（Mac 版本可执行文件）
- SIoT1.3_linux（Linux 版本执行文件）

注：更多操作系统支持的文件在不断增加中。我们正在开发 windows 下的“启动助手”，让软件的使用更加方便。

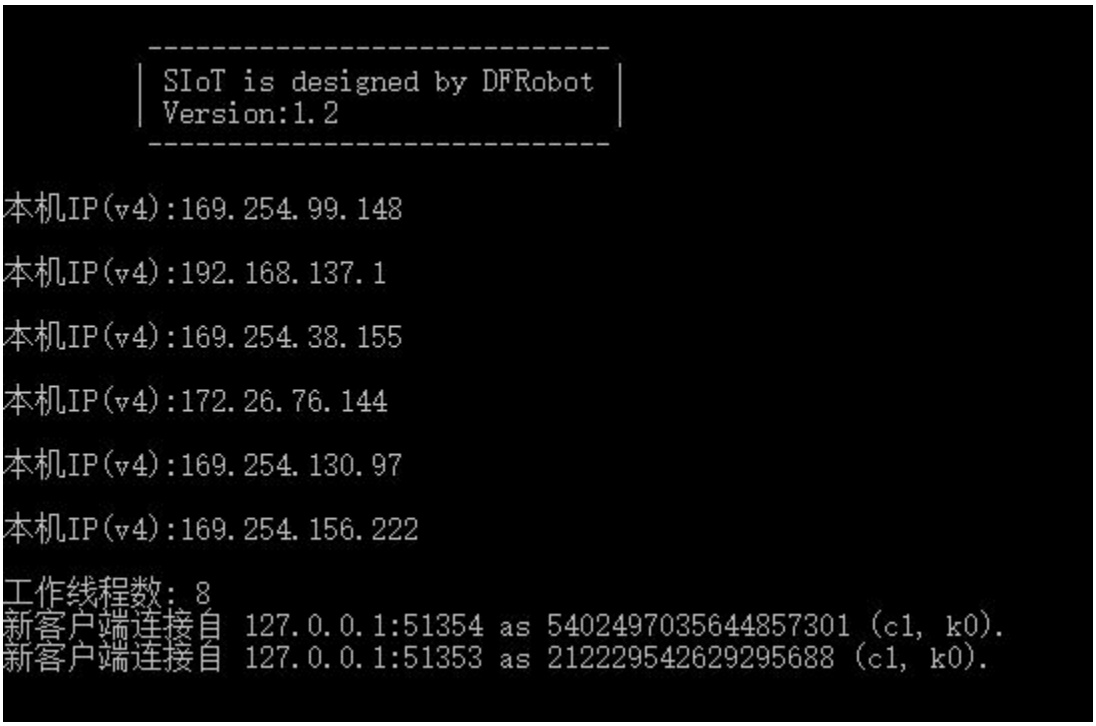
根据自己的操作系统，你可以运行相对应的软件。以 1.2 版为例，简要介绍使用过程。

2.2.1 Window 版本

双击运行 SIoT_windows1.2.exe（根据你的电脑，正确选择 64 位或者 32 位），将看到一个黑色的 CMD 窗口。窗口中显示各种连接信息。

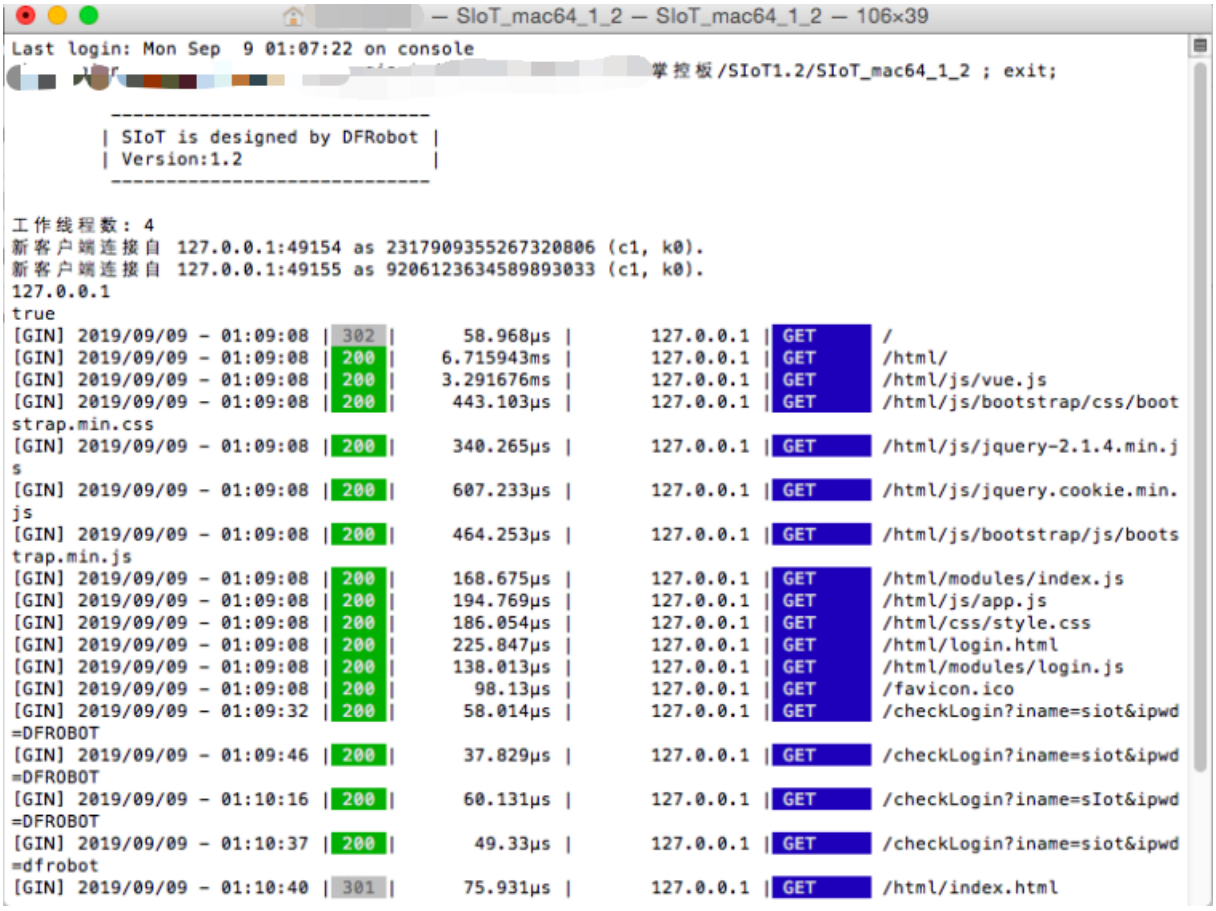
如果你想维持你的计算机作为 MQTT 服务器的话，请不要关闭它。

注意：SIoT 运行后会列出这个电脑可以使用的所有 IP，169 开头的一般是私有 IP 地址，不能作为 MQTT 服务器 IP。



2.2.2 Mac 版本

双击运行，或者打开终端转到相应目录然后执行命令，如“./SIoT_mac64_1_2”。



注意：如果提示没有权限，先添加“执行”权限。

- 增加执行权限的命令：chmod a+x ./SIoT_mac64_1_2

如果担心程序运行后被中止，可以使用 nohup 命令运行。

2.2.3 linux 版本

支持虚谷号、树莓派等开源硬件。

参考命令：nohup ./SIoT_linux &

其中“SIoT_linux”为程序的路径。

注意：需要用“chmod -R 777”命令，将目录的权限设定为最高权限。

2.2.4 服务器信息

SIoT 启动后，你的计算机就成为了一个标准的 MQTT 服务器，使用任何一款 MQTT 客户端程序就可以访问。

- 服务器地址：计算机局域网 IP 地址（在启动 SIoT 环境的时候会显示在黑框中，有时会显示多个 IP 地址，需要逐个尝试，一般为 192 或者 172 开头。）
- MQTT 端口：1883
- 用户名：siot（小写）
- 默认密码：dfrobot（小写）
- 消息主题（Topic）：项目名/设备名（可以自定义，中间的“/”不可缺少。）
- Web 管理地址：http://计算机 IP:8080（如果在本机访问，计算机 IP 可以是 127.0.0.1，也可以是局域网的 IP 地址，或者用“localhost”。）
  - 参考地址 1：http://127.0.0.1:8080
  - 参考地址 2：http://localhost:8080

注意：可以通过 config.json 文件修改用户名、密码和 Web 端口等信息。

2.3 界面简介

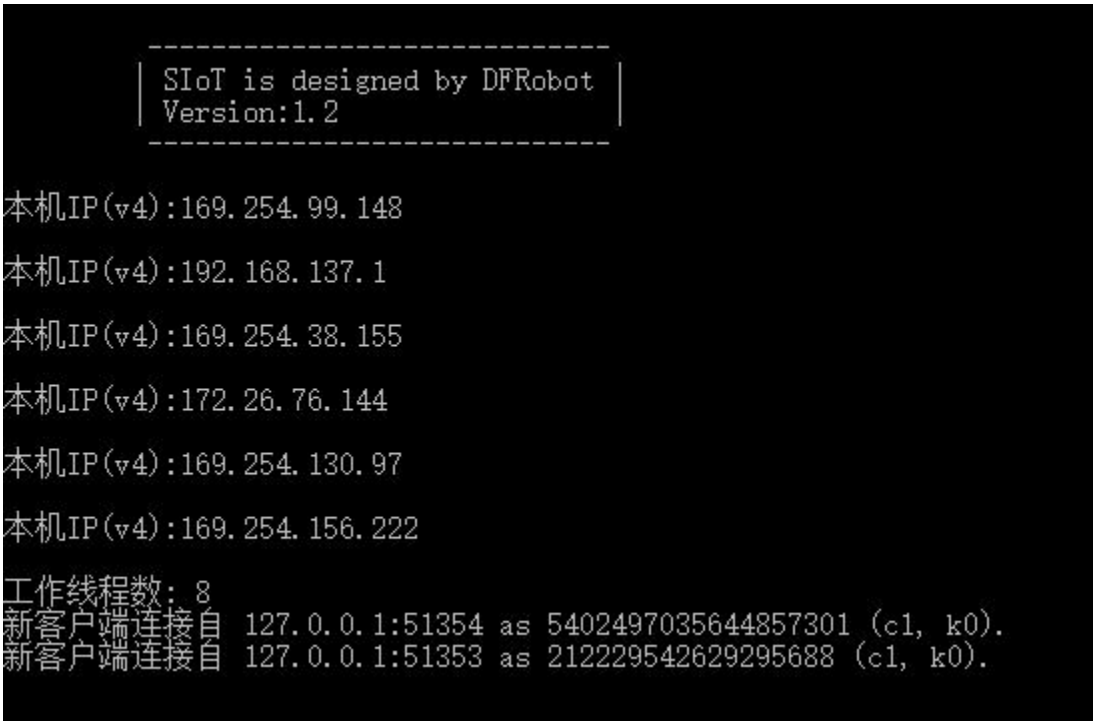
SIoT 既可以作为教师教学物联网课程的教学支持平台，也可以作为学生物联网作品的支持平台。当你下载完成之后就可以运行相应的软件啦。

下载地址：https://github.com/vvlink/SIoT/tree/master/software

或者 http://mindplus.dfrobot.com.cn/siot

2.3.1 登录 Web 页面

如果你是 Windows 用户，请运行 SIot_win.exe。类似的，如果你是 Linux 用户，请运行 SIoT_linux，如果你是 Mac 用户，请运行 SIoT_mac。启动软件后请不要关闭这个黑窗口，它将维持你的计算机（电脑）作为 MQTT 服务器。



打开浏览器，输入：http://localhost:8080 或者 http://127.0.0.1:8080 进行登录

- 使用 v1.2 版本时，在黑框可以看到所有可能的局域网 IP 地址，要想知道哪一个是可行的，可以尝试用这些地址发起连接。具体操作可以参考常见 MQTT 客户端 https://siot.readthedocs.io/zh_CN/latest/demo/01_tool_phone.html

默认用户名（user）为：siot

默认密码（pwd）为：dfrobot

注意：可以通过 config.json 文件修改用户名、密码和 Web 端口等信息。

2.3.2 查看项目

“项目列表”可以修改项目备注和查看设备。

登录成功之后，默认界面就是查看项目界面，你也可以通过点击上方菜单栏的“项目列表”访问。

地址：http://localhost:8080/html/

2.3.3 查看设备

平台提供一个简单的管理界面，能够输入自己的 Topic 和最大消息数量，查询到该 Topic 对应的消息。根据活动时间（最后一次使用时间）进行排序。

通过点击上方菜单栏的“设备列表”访问。

地址：http://localhost:8080/html/devices.html

2.3.4 查看数据

设备的管理功能分为查看消息，清空消息，删除设备，添加备注等。

- 查看消息。可以根据 Topic 查看历史消息，并且可以全部下载。
- 清空消息。清空对应 Topic 的所有消息。

在“设备列表”界面找到需要查看的设备，在操作这一栏中点击“查看消息”访问。

默认可以看到近期 100 条消息（倒序），以及折线图。

- 你可以点击“导出查询结果”按钮，在线生成 Excel 文件，默认名称为“消息数据.xls”。

地址：http://localhost:8080/html/messages.html?topic=PROGRAM_ID/TOPIC_ID

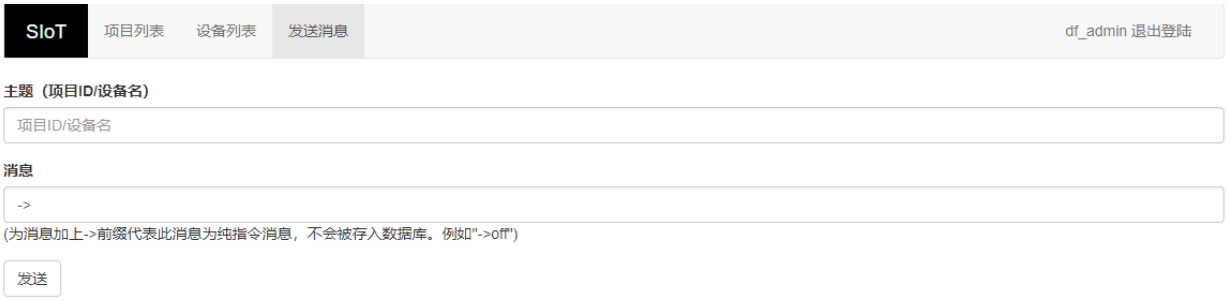
请把 PROGRAM_ID 替换成自己的项目名，把 TOPIC_ID 替换成自己的消息名。

2.3.5 发送消息

可以发送 255 字符内的字符串，包括中文。

通过点击上方菜单栏的“发送消息”访问。

- Topic 的格式为：项目 ID/设备名。例如：Seifer/light1
- 消息发送成功后，系统会自动根据 Topic 建立“项目”和“设备”。如果项目和设备已经存在则在此设备上追加数据。



地址：http://localhost:8080/html/sendMsg.html

注意：当前 Topic（消息）有通信记录（发送或接收均可）后，你也可以通过设备列表-查看消息页面左上角

的“发送消息”窗口发送消息，这样省去了输入 Topic 的麻烦步骤。

地址：http://localhost:8080/html/messages.html?topic=PROGRAM_ID/TOPIC_ID

请把 PROGRAM_ID 替换成自己的项目名，把 TOPIC_ID 替换成自己的消息名。

2.4 快速入门

下面以 win7 系统为例，介绍以 SIoT 为 MQTT 服务器，掌控板板为智能终端，搭建一个最简单的物联网数据采集系统。

2.4.1 运行 SIoT 软件

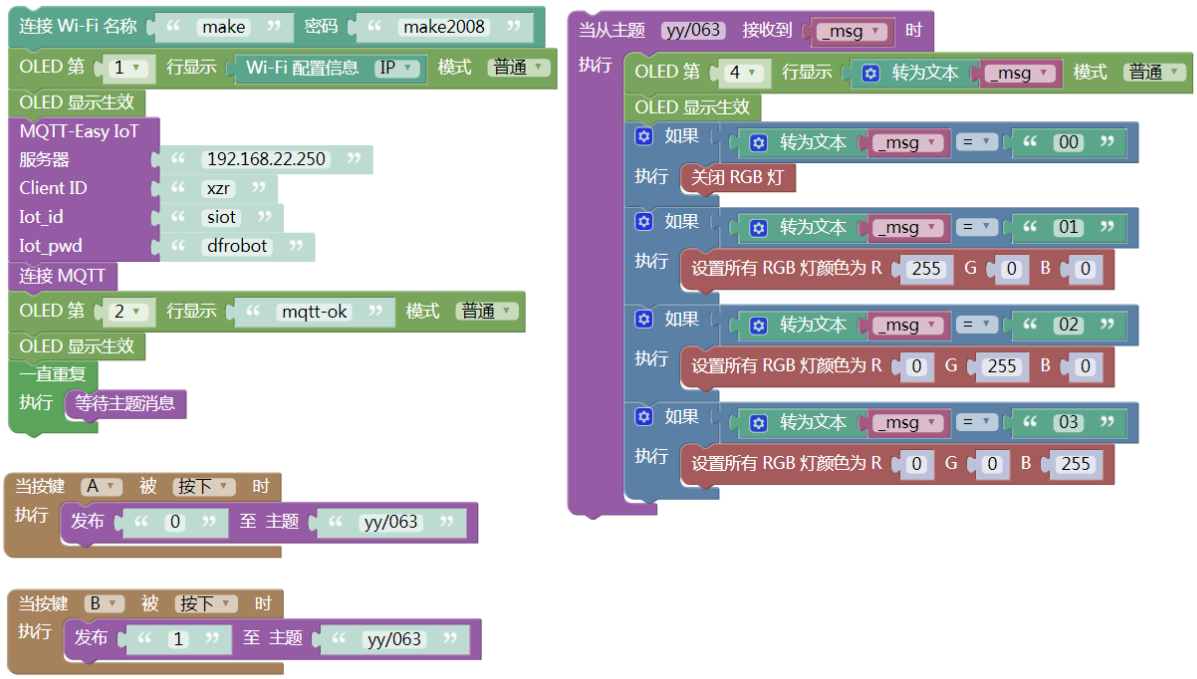
双击运行 SIoT_win.exe，将看到一个黑色的 CMD 窗口，不要关闭它。

```
        ----------------------------
        | SIoT is designed by DFRobot |
        | Version:1.2                 |
        ----------------------------

本机IP(v4):169.254.99.148

本机IP(v4):192.168.137.1

本机IP(v4):169.254.38.155

本机IP(v4):172.26.76.144

本机IP(v4):169.254.130.97

本机IP(v4):169.254.156.222

工作线程数: 8
新客户端连接自 127.0.0.1:51354 as 5402497035644857301 (c1, k0).
新客户端连接自 127.0.0.1:51353 as 212229542629295688 (c1, k0).
```

如图所示，显示的 IP 地址有多个，但是可以在局域网中使用的是“192”开头的那一个。

2.4.2 编写程序（mPythonX）

打开 mPythonX，编写如下代码。



说明:

- 请确保掌控板和电脑所处的局域网可以相互访问，最简单的方式是连接同一个无线路由器。
- “mpythonx/001” 表示项目名称为 “mpythonx”，设备名称为 “001”。
- 服务器地址就是运行 siot 的电脑的 ip 地址。

给掌控板写入程序并且运行。

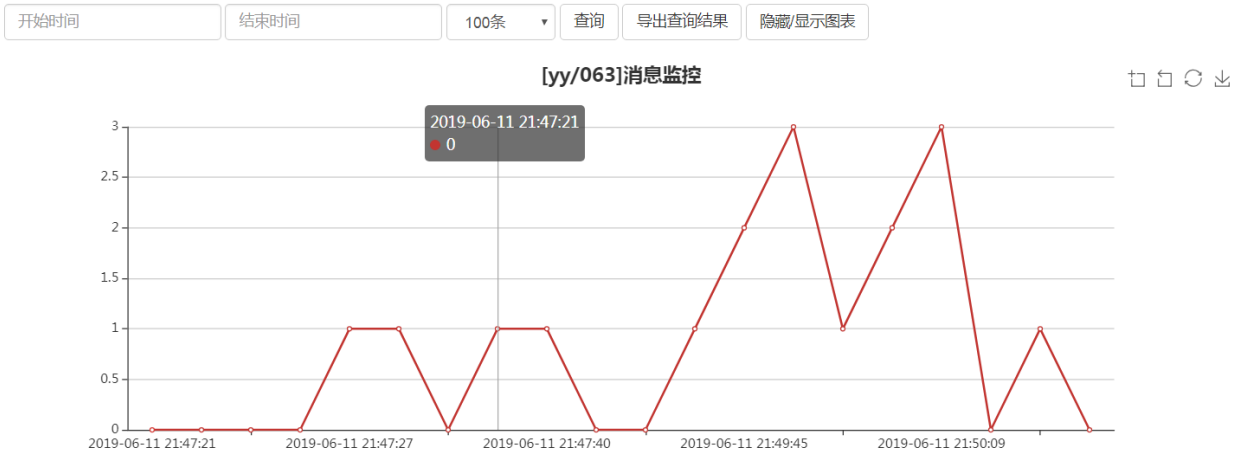
重新启动掌控板，等屏幕显示 IP 地址后，如果出现 “mqtt-ok”，说明 SIoT 服务器连接成功。

2.4.3 Web 管理

打开网址：localhost:8080（或者使用电脑的 IP 地址）。

输入默认的用户名 “siot” 和密码 “dfrobot”（虚谷号自带的 SIoT），就可以看到项目列表中多了” yy”。

在名称为 “yy” 设备消息中，当按下 “A” 键时，在网页端就可以接收到 “0”，按下 “B” 键，可以接收到 “1”。



Topic	消息	时间
yy/063	0	2019-06-13 22:03:44
yy/063	1	2019-06-13 22:03:42
yy/063	00	2019-06-11 21:50:13
yy/063	03	2019-06-11 21:50:09
yy/063	02	2019-06-11 21:50:05
yy/063	01	2019-06-11 21:50:02

在名称为 “yy” 设备消息中，当发送 “01” 时，就可以控制掌控板的红灯亮；发送 “02” 时，就可以控制掌控板亮绿灯；发送 “03” 时，控制掌控板亮蓝灯；发送 “00”，控制掌控板灯熄灭。

2.4.4 故障排查

1、如果接收不到数据，请关闭运行 SIoT 服务器的电脑的各种病毒防火墙或者网络防火墙（安全卫士）。

2、在其他电脑使用 MQTT 客户端测试 SIoT，推荐 MqttTool（一个测试 mqtt 的软件，只有 100k 不到），或者 MQTTX。

- GitHub 地址：https://github.com/vvlink/SIoT/tree/master/MQTT%20tools/Mqtttool
- 下载地址：https://github.com/vvlink/SIoT/tree/master/MQTT%20tools/Mqtttool

3、在手机使用 MQTT 客户端测试 SIoT。

安卓系统推荐使用 MQTT Client。

- 下载地址：http://www.mdpda.com/app/apk7623192.html

iPhone 系统推荐使用 MQTTool，通过 App Store 即可安装 MQTTool。

这些软件的使用，可以参考“客户端连接范例”。

https://siot.readthedocs.io/zh_CN/latest/demo/index.html

4、部分 MQTT 客户端需要设置 QoS 级别，SIoT 的 QoS 级别为 0。

CHAPTER 3

客户端连接范例

这一部分主要介绍各种客户端和 SIoT 软件的连接。SIoT 为标准的 MQTT 服务器，支持绝大多数的客户端程序连接。

3.1 常见 MQTT 客户端（PC 端）

在给掌控板、Arduino 之类的开源硬件写物联网代码之前，建议先使用 PC 端工具（另一台电脑）测试一下，确认 SIoT 的服务是否正常。

注意：如果测试不成功，请检查如下几点。

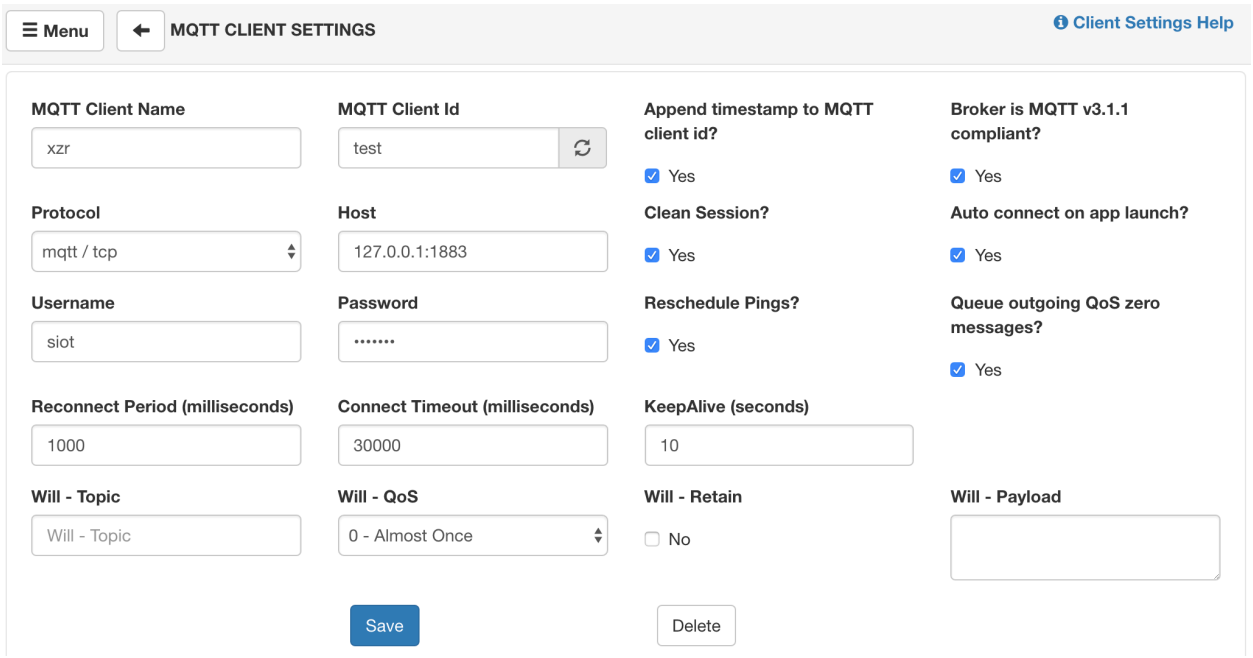
- 运行 SIoT 的电脑和测试电脑，是否在同一个局域网内（其实只要能够相互 ping 通即可）。
- 运行 SIoT 的电脑，是否关闭了病毒或者网络防火墙（如果不熟悉网络配置，建议停用或者卸载）。

3.1.1 MQTTBox

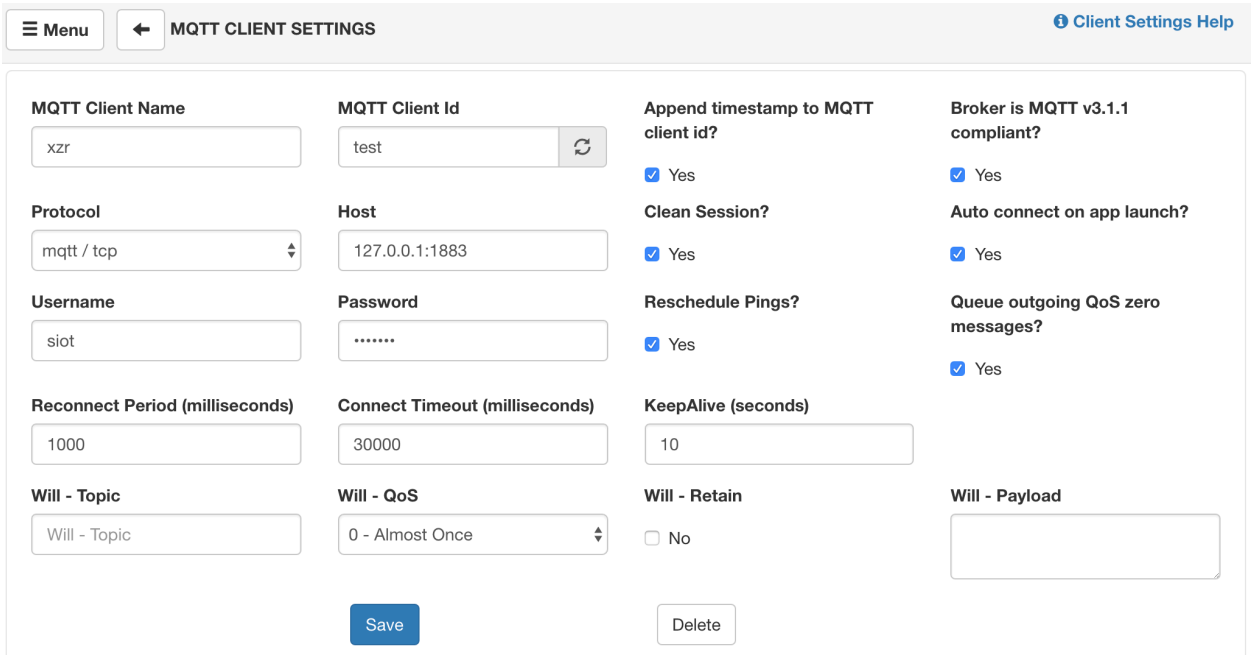
MQTTBox 是一款很好用的 MQTT 客户端调试工具，支持在 Windows、Mac 和 Linux 上面运行。

GitHub 地址：https://github.com/workswithweb/MQTTBox

首先要设置 MQTT 服务器信息，如图所示，除了服务器地址、用户名、密码和 Protocol 外，其他的地方都可以用默认值。



在这个界面中，连接服务器后，就可以给名称为“xzr/003”的 Topic 发送消息和订阅消息了。



在 Web 管理界面中，就能看到自动建立的项目中多了“xzr”，设备中多了“003”，通过网页就可以查看所有的消息记录。

注意

- 客户端发送的消息，所有订阅了这一主题的客户端都能收到，包括自己；
- “xzr/003” 这个 Topic 不需要事先设置，只要发过一次消息，SIoT 会根据这一 Topic 自动建立项目名称和设备名称，方便管理；
- 通过 Web 给 Topic 发送消息，如果不想将这一消息记录在数据库中，可以在前面加上 “->” 的标志。

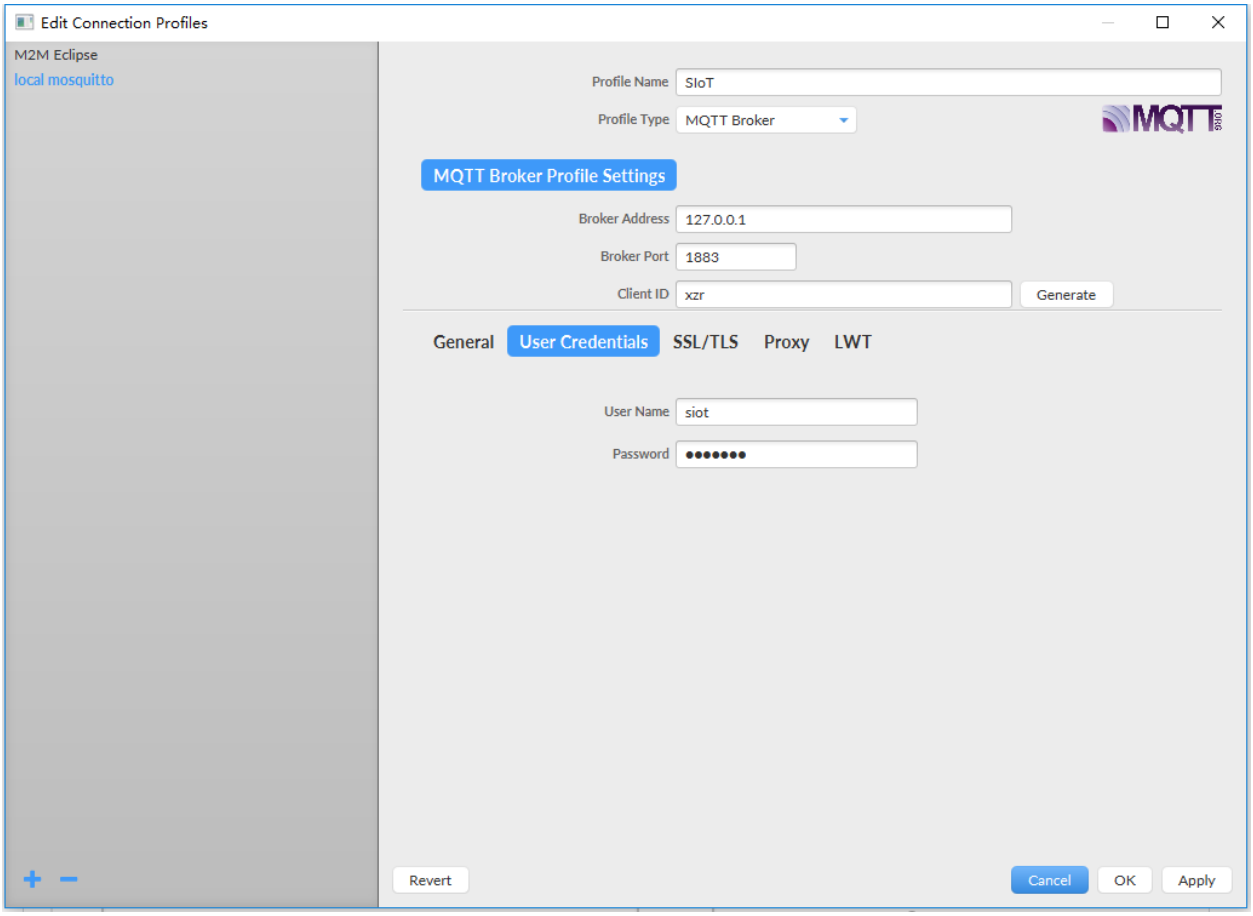
3.1.2 MQTTfx

MQTTfx 是一款很好用的 MQTT 客户端调试工具，支持在 Windows、Mac 和 Linux 上面运行。设备将当前所处的状态作为 MQTT 主题发送给 IoT Hub，每个 MQTT 主题 topic 具有不同等级的名称，如 “建筑/楼层/温度。” MQTT 代理服务器将接收到的主题 topic 发送给给所有订阅的客户端。

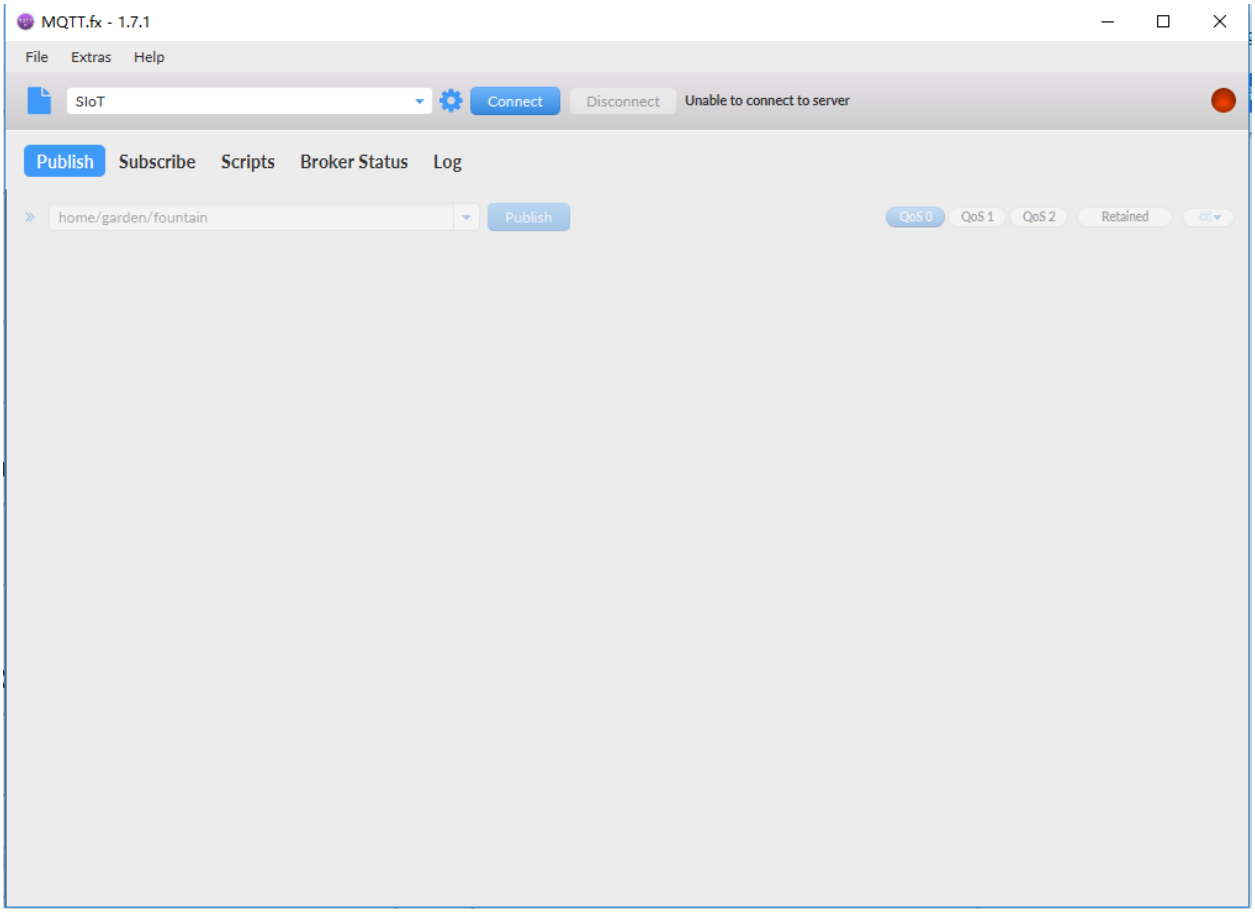
下载地址（支持 windows，linux，mac): http://mqttfx.jensd.de/index.php/download

安装完成后，运行时会提示有更新，最好别点击 yes，会报错。

主界面点击左上方的 Extras，进入 Edit connection Profiles，设置 MQTT 代理。这里上半部分如图填写（ip 地址有时会有不同），下半部分选择 User Credentials，填写用户名和密码。



接着回到主界面，点击 connect 连接到 MQTT 代理服务器上，就可以进行订阅和发布消息测试了。

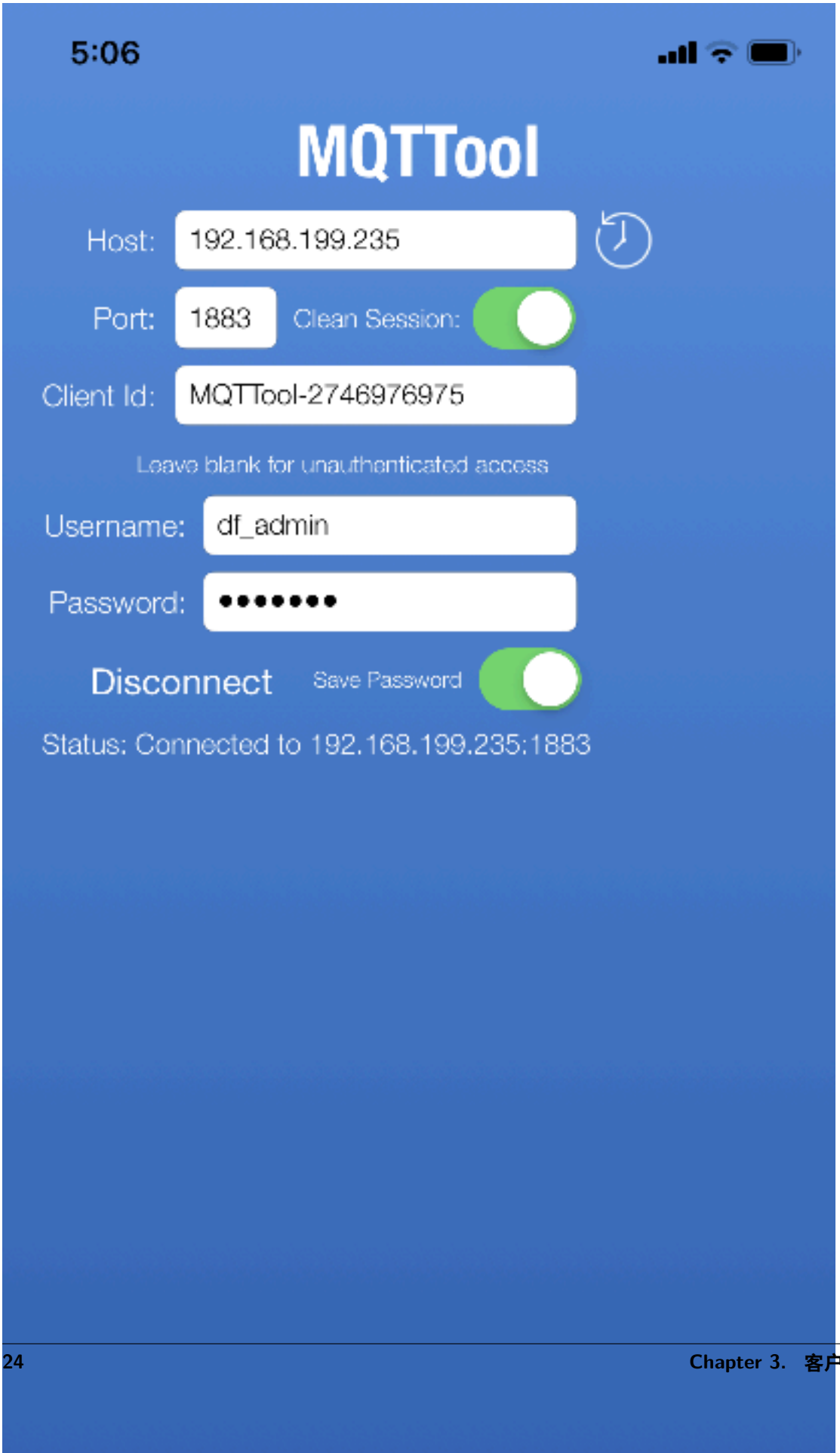


3.2 常见 MQTT 客户端（手机端）

手机俨然成为人们最熟悉的智能终端了。借助一些 MQTT 客户端 App，手机可以连接 SIoT，成为一个物联网终端。本文分别介绍运行于 iPhone 和安卓系统的两款 App。有 App Inventor2 基础的用户，可以自行开发一个。

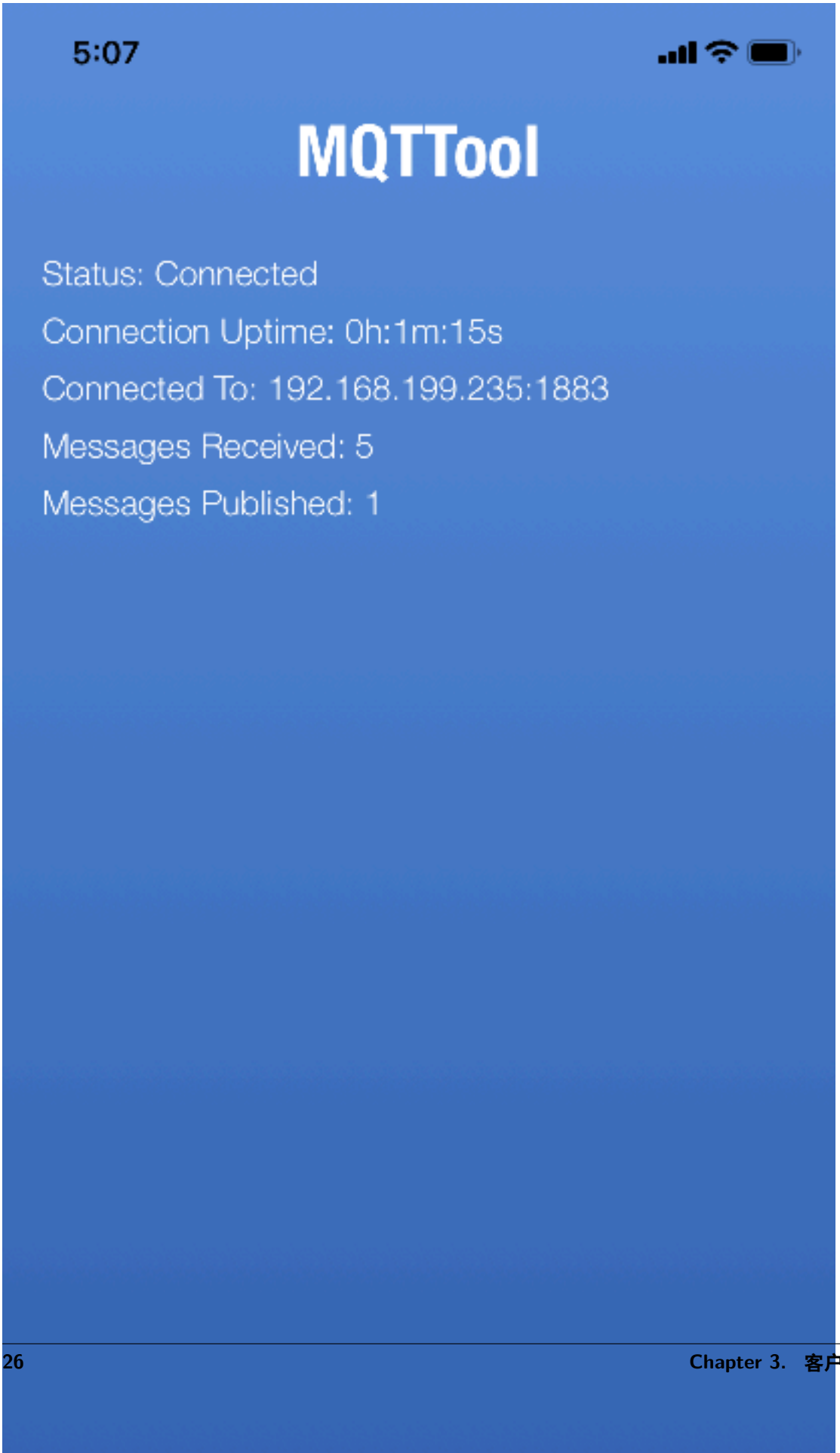
3.2.1 MQTTool（iPhone 手机）

MQTTool 是运行在 iPhone 上的 MQTT 客户端软件，也是一个 MQTT 测试工具。通过 App Store 即可安装 MQTTool。

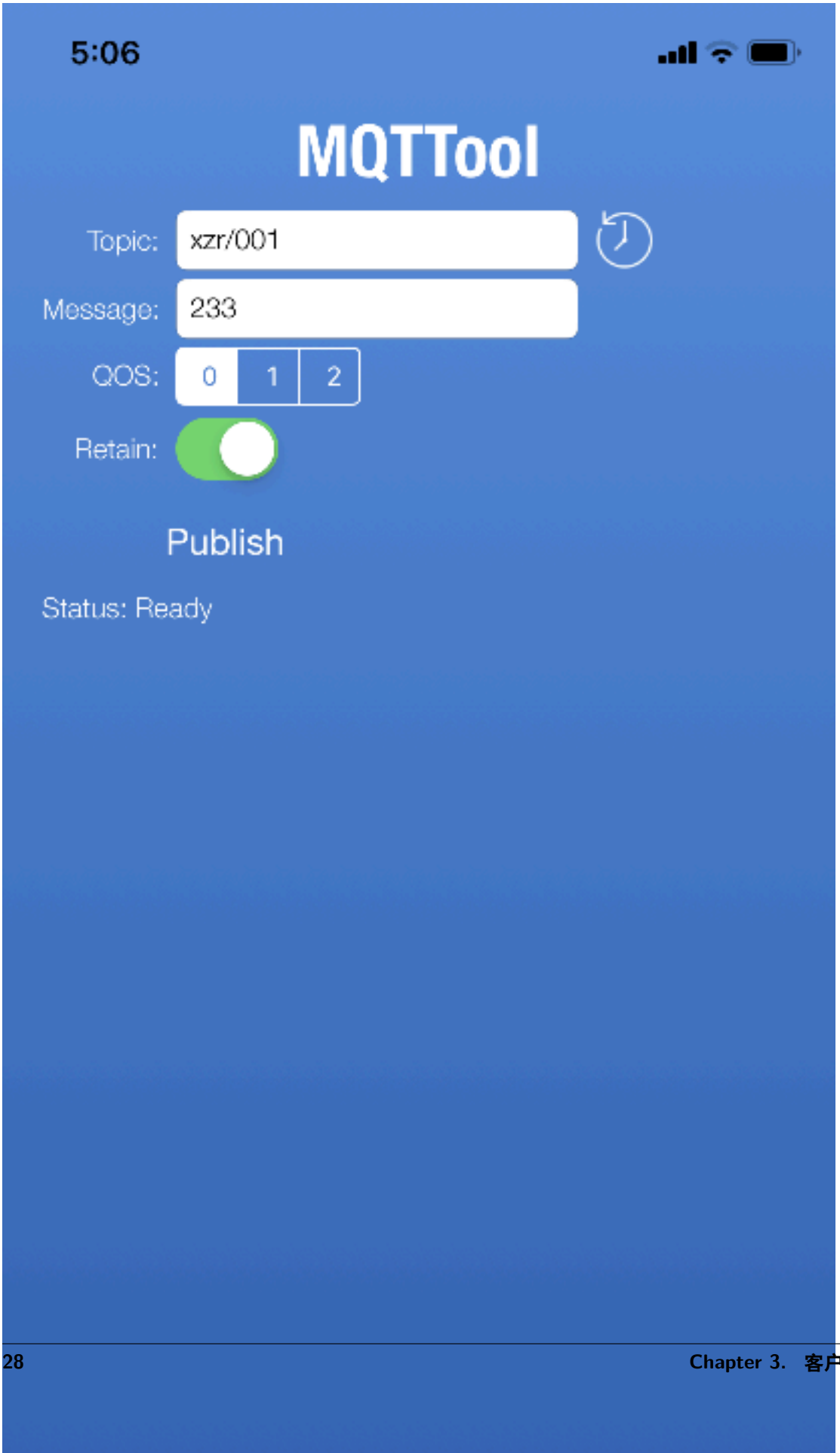

5:06
MQTTool
Host:
192.168.199.235
Port:
1883
Clean Session:
Client Id:
MQTTool-2746976975
Leave blank for unauthenticated access
Username:
df_admin
Password:
Disconnect
Save Password
Status: Connected to 192.168.199.235:1883

首先要确认计算机和手机连接的是同一个无线路由器，或者在同一个局域网中，相互之间可以访问。输入计算机的 IP 地址和用户名、密码，点击“Connenct”即可连接。

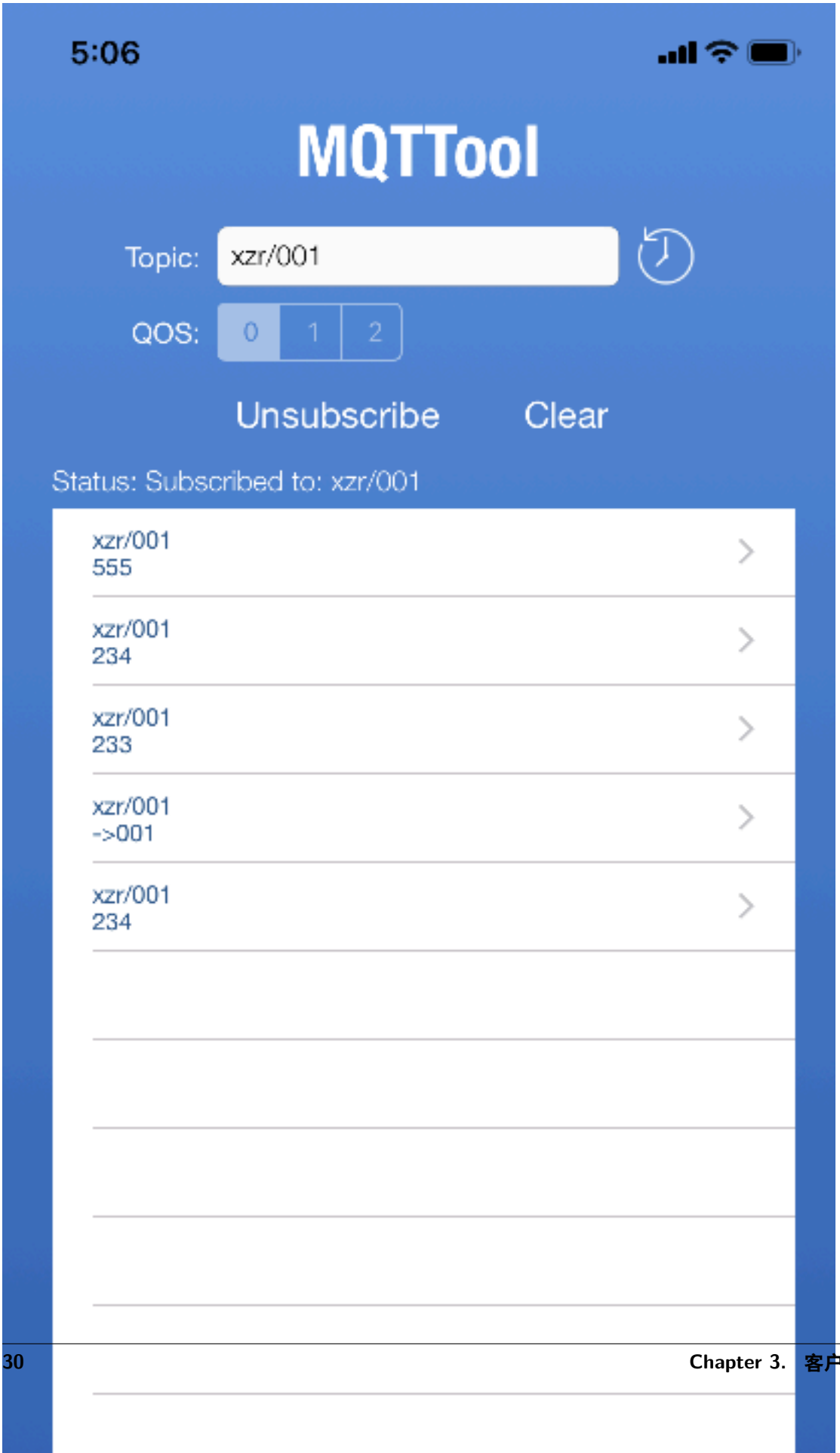
- 图中的 username 应填写 siot


5:07
MQTTool
Status: Connected
Connection Uptime: 0h:1m:15s
Connected To: 192.168.199.235:1883
Messages Received: 5
Messages Published: 1

点击“Stats”，可以查看服务器的连接状态。


5:06
MQTTool
Topic:
xzr/001
Message:
233
QOS:
0
1
2
Retain:
Publish
Status: Ready

在“Topic”（主题）中输入“xzr/001”（表示项目 id 为 xzr，设备 id 为 001），点击“Publish”（发送）即可发送消息。在 SIoT 的 Web 页面可以看到这一消息。

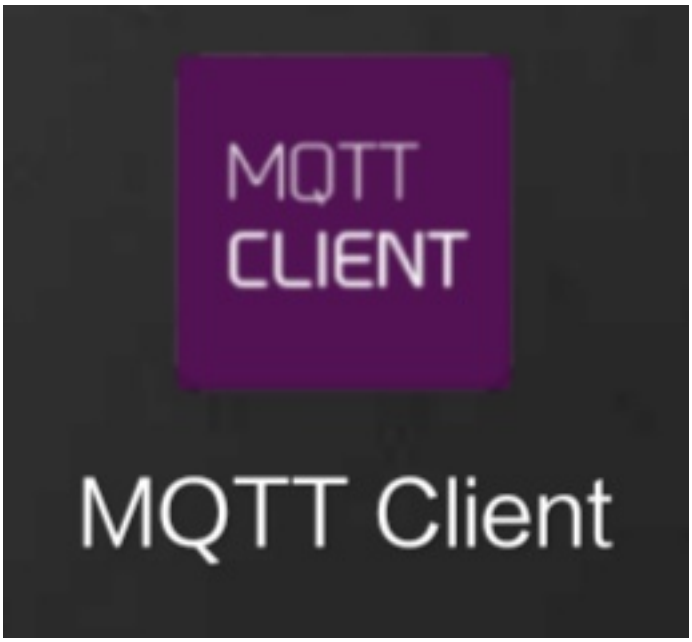

5:06
MQTTool
Topic:
xzr/001
QOS:
0
1
2
Unsubscribe
Clear
Status: Subscribed to: xzr/001
xzr/001
555
xzr/001
234
xzr/001
233
xzr/001
->001
xzr/001
234

同样，在 Web 端给 Topic “xzr/001” 发送消息，手机端即可收到信息，在 “Subscribe”（订阅）。

多个手机同时连接这个 MQTT 服务器，只要 Topic 相同，相互之间都能收到。

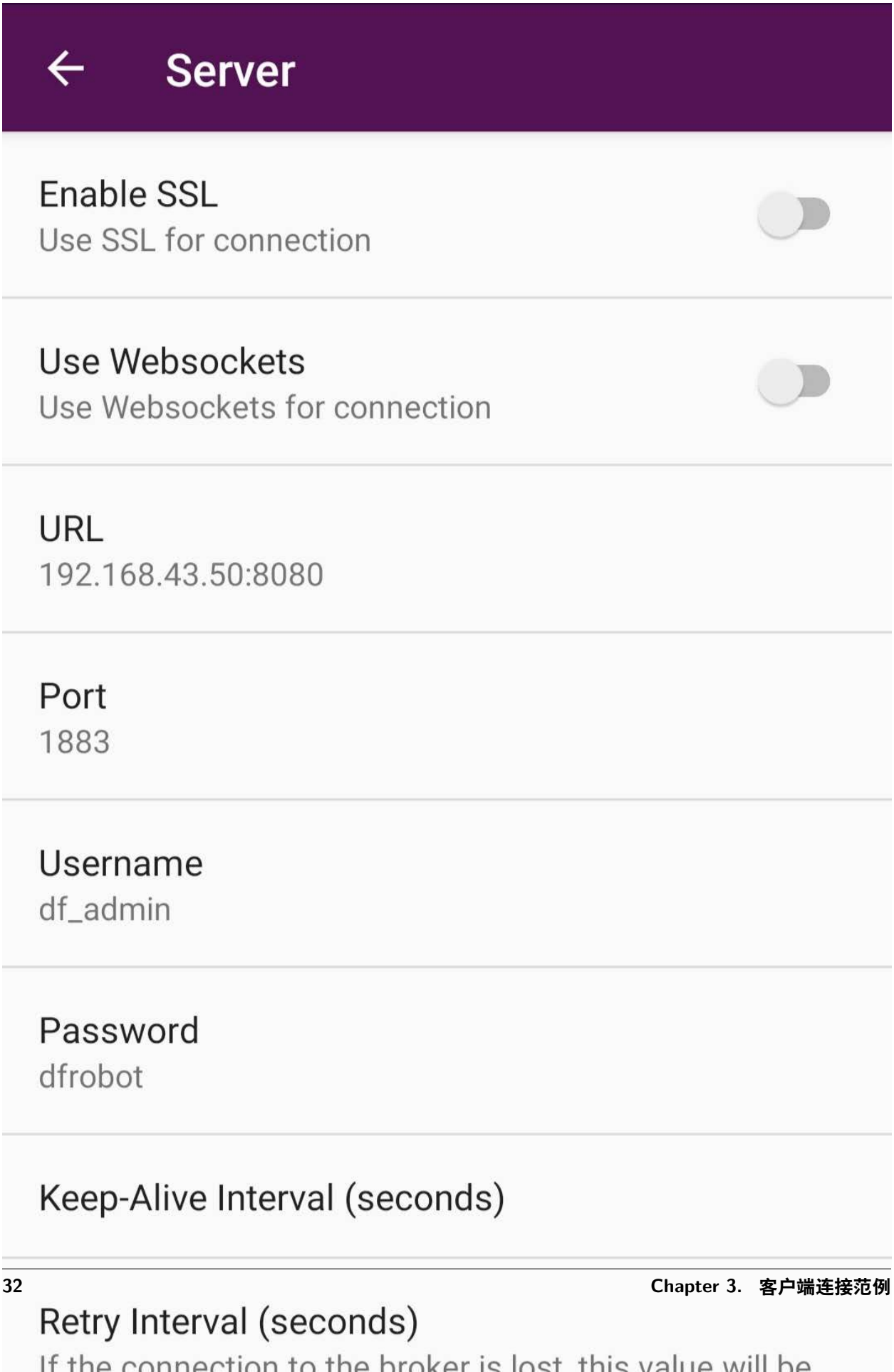
3.2.2 MQTT Client（安卓手机）

MQTT Client 是一款安卓环境的 MQTT 客户端软件，也是一个 MQTT 测试工具。可以在 http://www.mdpda.com/app/apk7623192.html 下载。



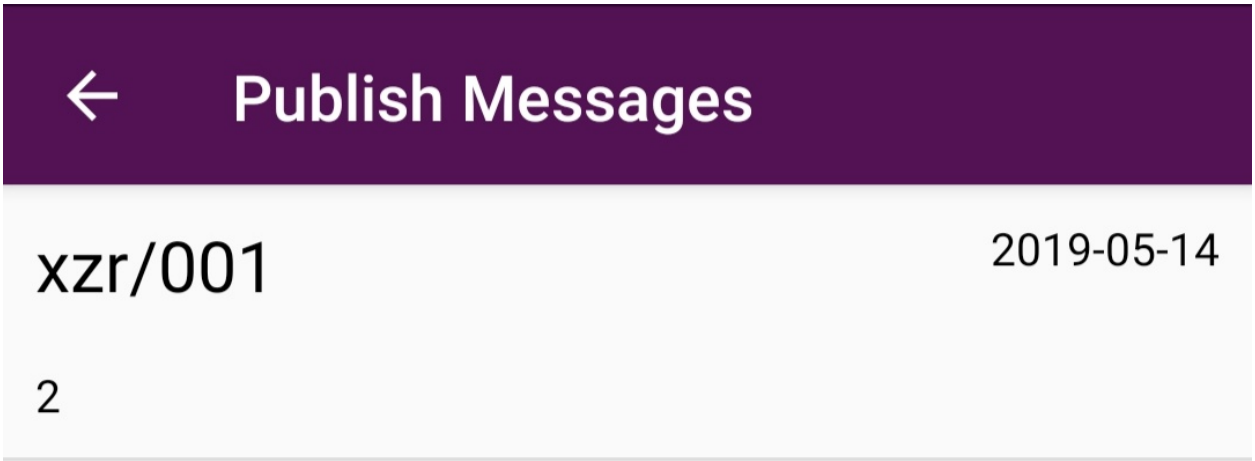
首先要确认计算机和手机连接的是同一个无线路由器，或者在同一个局域网中，相互之间可以访问。

打开软件，点击右上角的 Settings，点击 Server。

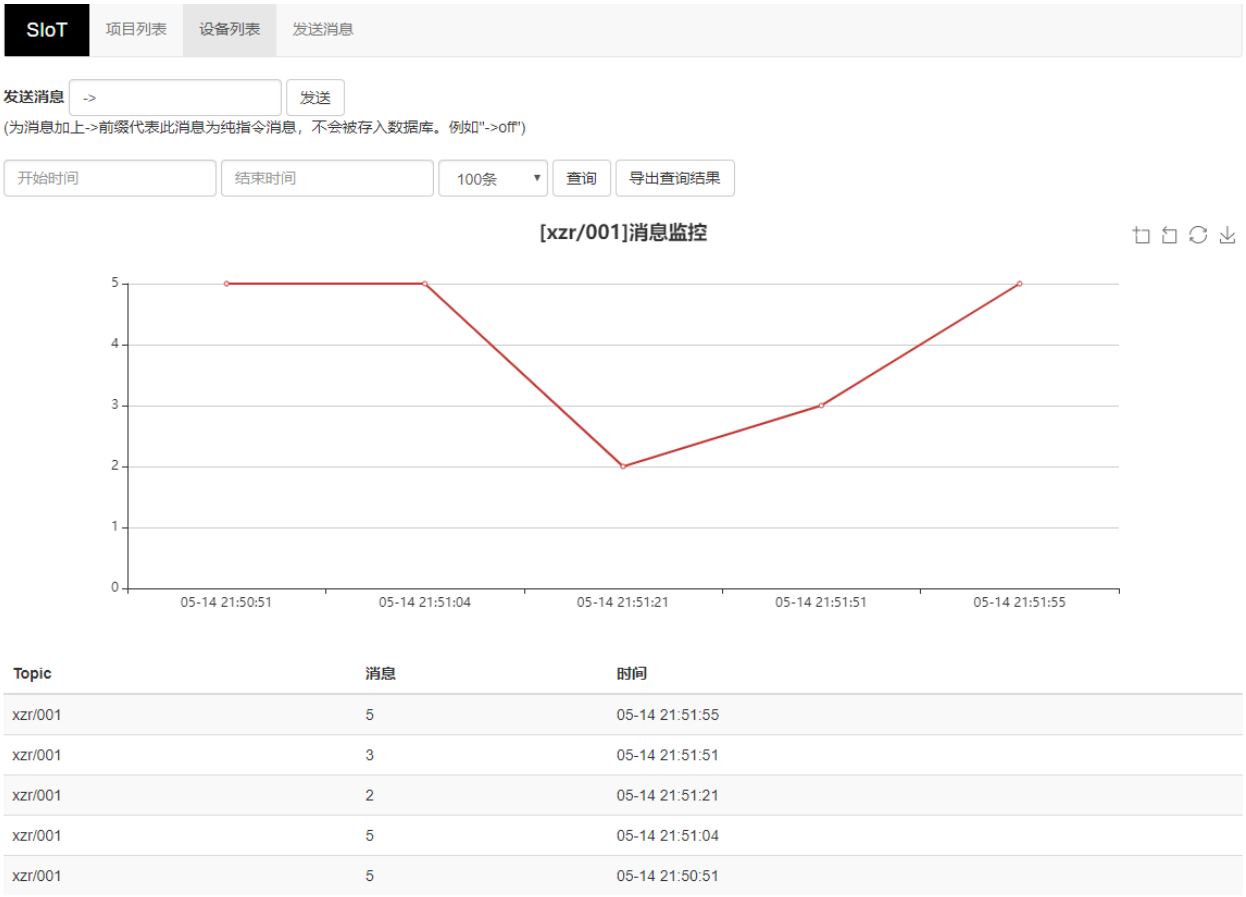

Server
Enable SSL
Use SSL for connection
Use Websockets
Use Websockets for connection
URL
192.168.43.50:8080
Port
1883
Username
df_admin
Password
dfrobot
Keep-Alive Interval (seconds)
Retry Interval (seconds)

- 在 URL 处输入计算机的 IP 地址，后面跟上 “: 8080”
- Port 填写 1883
- Username 填写用户名（siot）
- Password 填写密码（dfrobot）
- 其他内容保持默认就可以了

发送消息需要在主页面点击右下角的紫色按钮，进入发送消息界面。

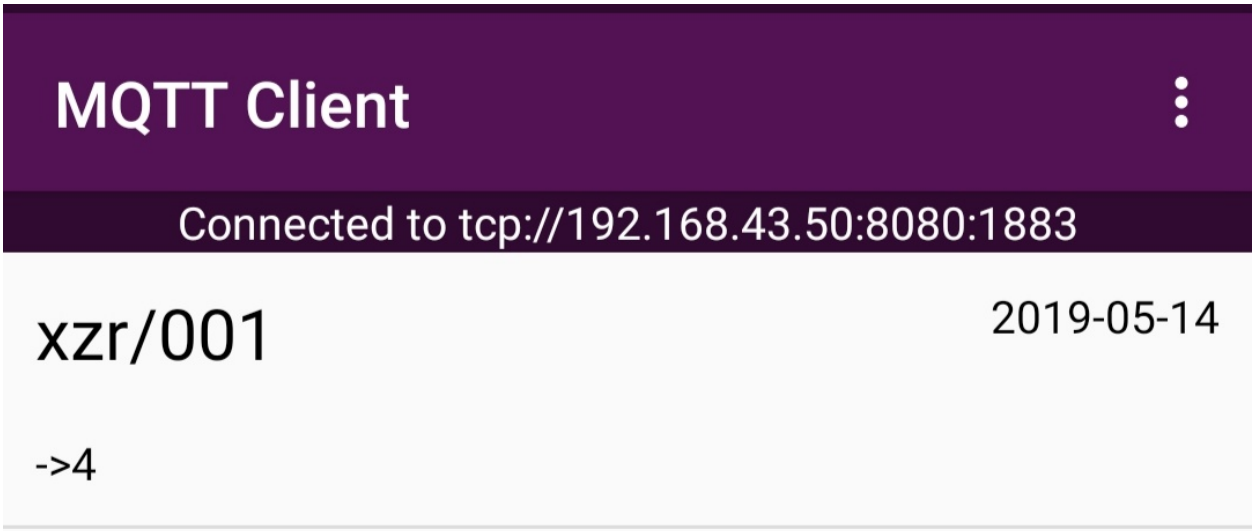

Publish Messages
xzr/001
2019-05-14
2

在“Topic”（主题）中输入“xzr/001”（表示项目 id 为 xzr，设备 id 为 001），在“Message”（消息）中输入想要发送的消息内容，点击“PUBLISH”（发送）即可发送消息。在 SIoT 的 Web 页面可以看到这一消息。



同样，在 Web 端给 Topic“xzr/001”发送消息，手机端即可收到信息，这一消息在主页面可以订阅。

在主页面下方输入想要订阅的主题“xzr/001”，就可以获取实时消息了。


MQTT Client
Connected to tcp://192.168.43.50:8080:1883
xzr/001
2019-05-14
->4

想要看具体的消息收发详情，可以点击相应的栏目查看，例如点击上图中的 “xzr/001” 就可以出现如下画面。

->4

xzr/001
Not Retained 2019-05-14 13:54:32

->1

xzr/001
Not Retained 2019-05-14 13:54:27

->3

xzr/001
Not Retained 2019-05-14 13:54:12

5

xzr/001
Retained 2019-05-14 13:54:06

多个手机同时连接这个 MQTT 服务器，只要 Topic 相同，相互之间都能收到。

3.3 Node-RED

Node-RED 是 IBM 开发的一个开源项目，本来是为了满足工程师快速连接硬件和设备到 Web 服务和其他软件的需求。因为编程简单，流程清晰得到好评，并很快发展成为一种通用的物联网编程工具。Node-RED 与 Scratch 的编程思想相近，通过节点块（Node）完成基础代码的编写，而节点间数据的传递则通过连线来创建数据流（Flows）。Node-RED 提供了一系列支持服务器及物联网的接口，能在传感器、服务器、路由器等设备间建模大量应用程序功能，简化了整体项目的开发。只需要简单修改节点中的参数，就能够让学生搭建出一个小有规模的客户端。

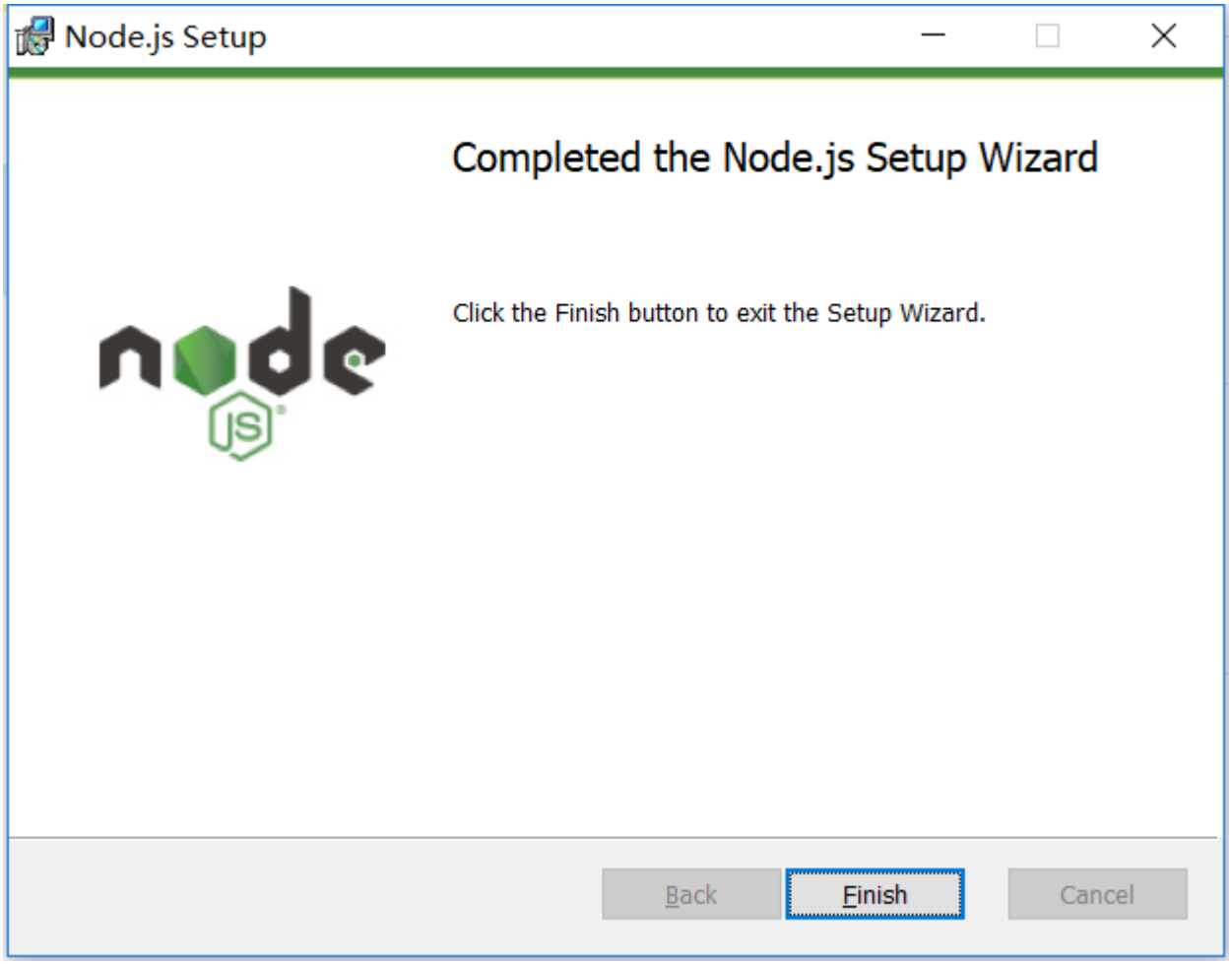
3.3.1 Node-RED 的安装

进入 Node.js 官网，跳转至下载界面。

http://nodejs.cn/download/

在该界面中选择相应的操作系统，下载.msi 文件，运行后就能完成 Node.js 的安装，Node.js 是用于运行 JavaScript 的网页编辑运行器，而 Node-RED 则是在其基础之上建立的。

在完成了 Node.js 的安装后，输入命令继续安装 node-red。



打开 cmd 窗口，输入以下命令完成 node-red 的安装

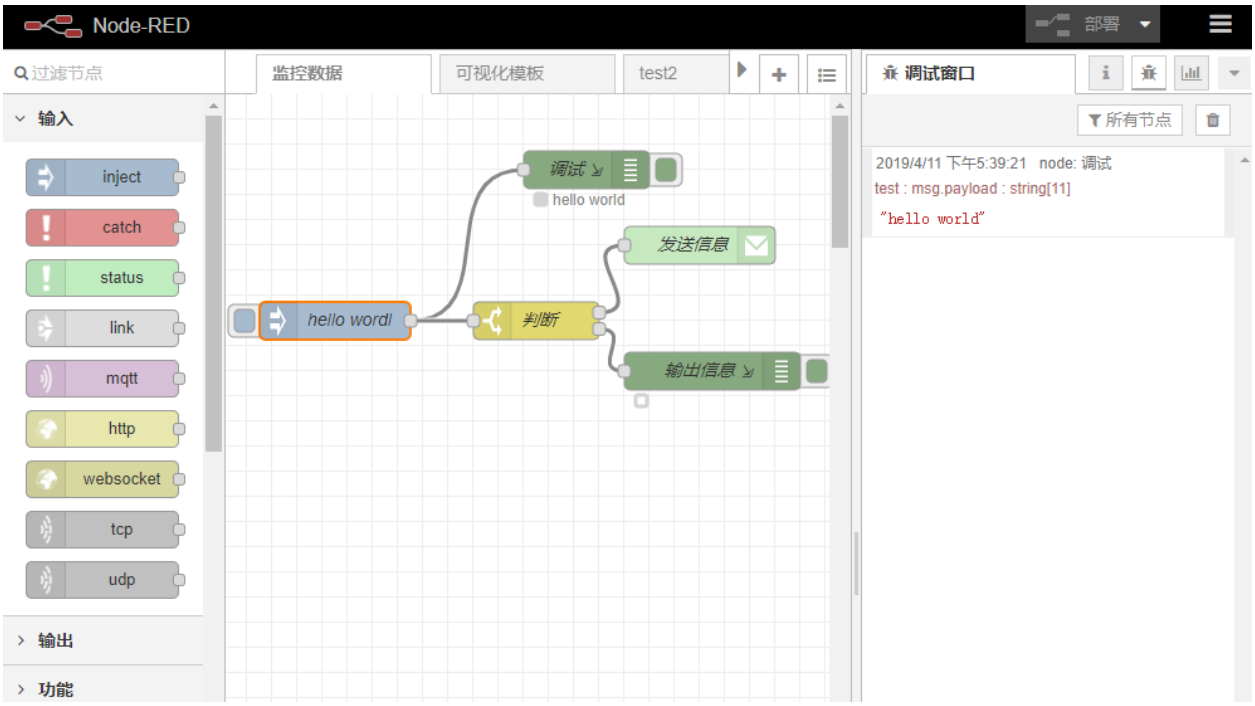
npm install -g –unsafe-perm node-red

安装成功后在 cmd 窗口中输入 node-red 即可启动服务器。

```
node-red
Microsoft Windows [版本 10.0.17134.706]
(c) 2018 Microsoft Corporation。保留所有权利。

C:\Users\13736>node-red
25 Apr 09:38:38 - [info]

Welcome to Node-RED
===================

25 Apr 09:38:38 - [info] Node-RED version: v0.20.3
25 Apr 09:38:38 - [info] Node.js  version: v10.15.3
25 Apr 09:38:38 - [info] Windows_NT 10.0.17134 x64 LE
25 Apr 09:38:38 - [info] Loading palette nodes
25 Apr 09:38:46 - [warn] rpi-gpio : Raspberry Pi specific node set inactive
25 Apr 09:38:47 - [info] Dashboard version 2.14.0 started at /ui
25 Apr 09:38:48 - [info] Settings file  : \Users\13736\.node-red\settings.js
25 Apr 09:38:48 - [info] Context store  : 'default' [module=memory]
25 Apr 09:38:48 - [info] User directory : \Users\13736\.node-red
25 Apr 09:38:48 - [warn] Projects disabled : editorTheme.projects.enabled=false
25 Apr 09:38:48 - [info] Flows file     : \Users\13736\.node-red\flows_LinmysComputer.json
25 Apr 09:38:48 - [info] Server now running at http://127.0.0.1:1880/
```

服务器需要在终端中持续运行。在浏览器中输入 http://127.0.0.1:1880/

进入编程环境，我们可以在终端看到数据流的开始与终止，一旦关闭终端，浏览器会提示丢失连接。同一局域网内的终端可以在浏览器中输入" IP 地址:1880"，进入 node-red 界面。比如本机 IP 地址为 192.168.102.xx，在本机 cmd 终端输入 node-red 成功建立服务器后，只需要输入 192.168.102.xx:1880 便可以在同一局域网内编辑该服务器上的节点。



更加详细的安装过程可以参考网上其他教程。

3.3.2 连接步骤

Node-RED 的主界面共有三个部分，从左到右分别为：拥有各种功能的节点栏，放置各种编程节点的流程栏，用于提供节点帮助和调试信息的信息栏。

在 Node-RED 中简单地输出一串字符，需要用到左侧的 inject 节点和 debug 节点，按住鼠标左键将节点拖至流程图中，发现节点的名字发生了改变，这是由于节点被实例化，代表某个具体的数值。我们可以通过修改节点的名称属性来改变其在流程图中的名字，并不影响整个流程中的其他数据。

在 Node-RED 中，MQTT 的节点有“接收”和“发送”两种，看图标即可区分。我们只需要将 MQTT 输入节点拖出，双击修改其中参数，就可以接收到从服务器传来的数据。

完成节点的拖动与信息的修改后需要点击右上方的“部署”，就可以在右侧的调试窗口中看到信息了。需要注意的是，如果不点击“部署”对当前流程进行保存，进行的操作与保存的数据都将会基于未保存之前的节点。

参考代码

在右方的节点中选择 MQTT，并对其中的参数进行修改，我们需要的参数有：

随机生成的主题

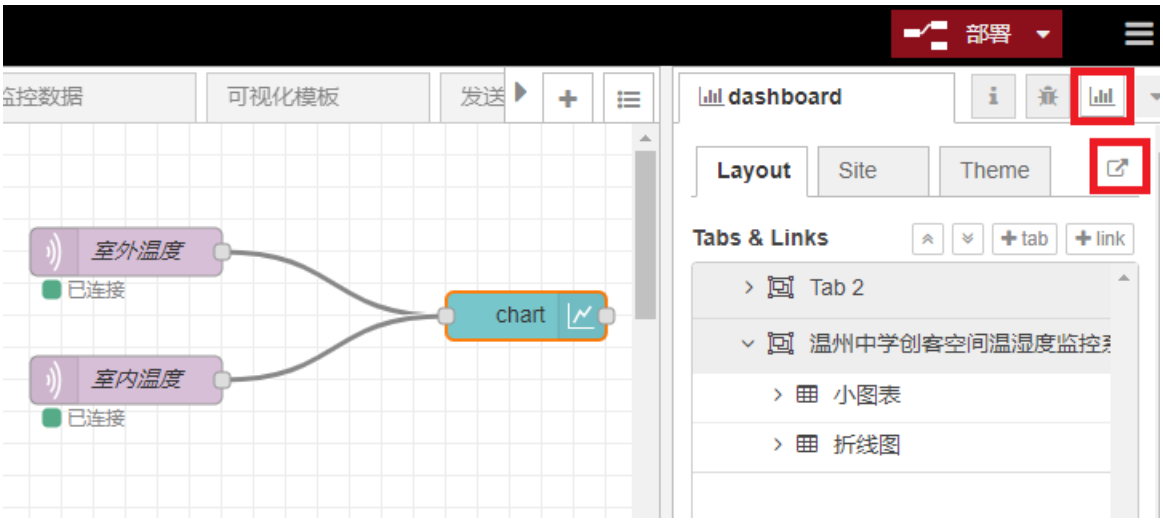
服务器 ip 及端口

随机生成的用户名及密码

拓展应用：数据可视化

在 Node-RED 中，有许多的第三方控件可以供我们使用。如果需要图表功能，则要使用 Dashboard 控件。

Dashboard 模块的安装：在右上角设置菜单中，选择节点管理，输入 Dashboard 进行安装，成功安装后会看到左侧的列表中出现了新的可用节点，我们可以从中选择不同的图表以及各种数据表现形式。



将已经设置好参数的 Mqtt 输入与折线图控件连接起来，点击右上角的部署，保存当前节点，点击图中右上角框选部分

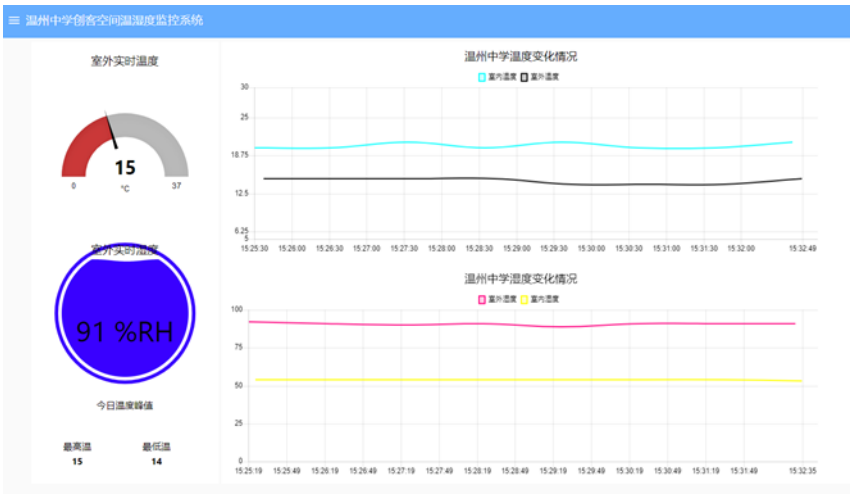
或在浏览器中输入地址 http://localhost:1880/u

就能得到一张简单图表。

其他拓展应用

用 Dashboard 呈现物联网数据，仅仅是 Node-RED 的众多功能之一。作为物联网的一种粘合剂，Node-RED 能够完成很多工作。比如，我们可以利用 Node-RED 监控物联网数据的传输情况，当传感器出现故障或者某个传感器数据达到阈值后，发送邮件提醒用户；可以简单的代码，按照既定的条件筛选比较数据，收集每日的温湿度峰值等；也可以根据数据的阈值，实时给某个 Topic 发送信息，实现物联网控制的功能。

完成效果图



3.4 Mind plus

Mind plus（Mind+）是著名开源硬件（创客教育）企业 DFRobot 推出的 IDE 工具。其支持两种编程模式，一是实时交互模式，二是离线下载模式。实时交互模式类似普通的 Scratch 程序，离线下载模式则是指给 Arduino、掌控板写程序。

3.4.1 Mind plus 的安装

这里介绍的是“”实时交互模式”下的 MQTT 范例。Mind+ 可以直接访问 MQTT 服务器，和其他接入 MQTT 服务器的智能硬件进行交互。

注意： Mind+ 在 1.55 版后才支持 MQTT 功能，请下载最新版本。

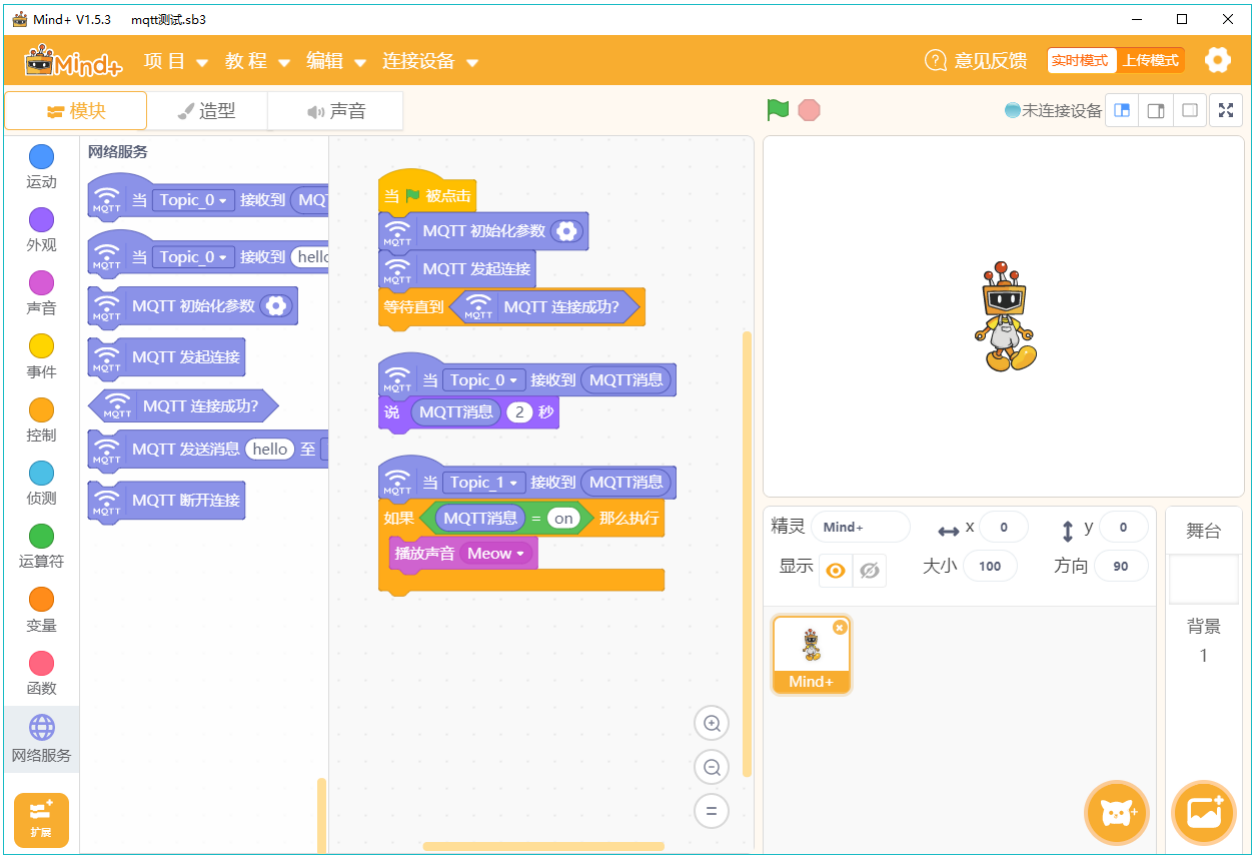
http://mindplus.cc/download.html

编程步骤

实时交互模式

1）在“实时模式”下选择“扩展”，找到网络服务，选择 MQTT。

2）在 MQTT 中，找到相应的语句开始编程。



代码功能简介

> 当 Topic_0（在 MQTT 服务器中设置）收到一条消息，小机器人就开始说话，将消息显示在对话气泡中。
>
> 当 Topic_1 收到一条内容为“on”的消息，小机器人发出猫叫声。

离线下载模式

1）在“上传模式”下选择“扩展”，找到对应的主控板，再到网络服务中选择 MQTT 和 WIFI。

2）示例代码如下，需要修改 WIFI 和 MQTT 的对应参数

代码功能简介

> 发送端：当与服务器成功连接后，不断发送信息到服务器

参考代码

1）可以在软件自带的示例代码中寻找。

2）更多代码下载地址：

https://github.com/vvlink/SIoT/tree/master/examples/Mind+

3.5 掌控板（mPython）

掌控板由虚谷计划组委会推出，是国内第一款专为编程教育而设计的开源硬件！为普及创客教育而生，反应一线 Python 编程教学需求，迎接普通高中新课改。掌控板委托创客教育知名品牌 Labplus 盛思设计、制造与发行，历经十几轮次研究讨论，三次升级改版，是国内第一款专为编程教育而设计的开源硬件。

3.5.1 掌控板功能介绍

掌控板上集成了 OLED 显示屏、RGB 灯、加速度计、麦克风、光线传感器、蜂鸣器、按键开关、触摸开关、金手指外部拓展接口，支持图形化及 python 代码编程，可实现智能机器人、创客智造作品等智能控制类应用。利用掌控板上丰富的传感器，结合它小尺寸的特点还可以做很多智能穿戴、电子饰品等各种 DIY 作品应用。

因为掌控板使用著名物联网芯片 ESP32，所以掌控板是“虚谷物联”项目的最佳选择。

掌控板地址：https://mpython.cn/mPython/index

3.5.2 掌控板的编程工具介绍

掌控板的编程工具很多，常用的有：

- mPython X
- Mind +
- Mixly
- BXY

3.5.3 掌控板的 MQTT 代码（基于 Mind+）

Mind+ 是一款基于 Scratch3.0 开发的青少年编程软件，让大家轻松体验创造的乐趣。

网站：http://mindplus.cc

代码说明：

这段代码可以提供消息的发送和订阅功能，MQTT 服务器既可以用 EasyIot 物联网，也可以用 SIoT。要实现功能，我们只需修改“发送消息”截图中的红框区域，设定相应的数据即可。

如果在左侧的图形化代码中没有看到蓝色的 MQTT 模块，请点击左下角的拓展，在主控板中选择掌控版，随后在网络服务中选择 wifi 和 mqtt 模块。

代码下载链接：**mind+ 中自带范例**

发送消息


ESP32 主程序
屏幕显示文字 掌控板 & 物联网 在坐标 X: 0 Y: 16 预览
WIFI 连接到 热点: dfrobotYanfa 密码: hidfrobot
等待直到 WIFI 连接成功?
屏幕显示为 全黑
屏幕显示文字 WIFI连接成功 在坐标 X: 0 Y: 22 预览
等待 1 秒
MQTT 初始化参数
MQTT 发起连接
屏幕显示为 全黑
屏幕显示文字 MQTT连接成功 在坐标 X: 0 Y: 22 预览
循环执行
如果 按钮 A 被 按下 ? 那么执行
MQTT 发送消息 Message received 至 Topic_0
如果 按钮 B 被 按下 ? 那么执行
MQTT 发送消息 Call back later 至 Topic_0

订阅消息

3.5.4 掌控板的 MQTT 代码（基于 Mixly）

Mixly 是在北京师范大学教育学部创客教育实验室负责人傅骞老师的带领下，由其团队开发的一款国内自主研发，且免费开源的图形化编程工具。

Mixly 支持包括 arduino、ESP32、ESP8266、micro:bit 在内的绝大多数创客类硬件。

软件下载及准备工作：https://mixly.readthedocs.io/zh_CN/latest/basic/01.Installation-for-Mixly.html

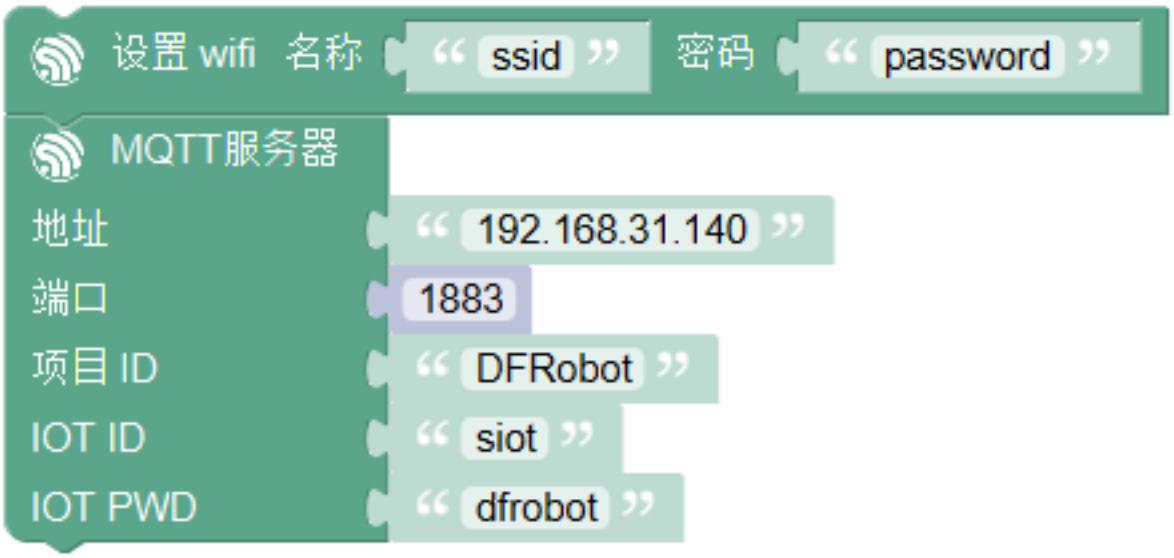
编程准备

下载并解压好 Mixly 之后，在软件右下角板卡选择中选择 Arduino HandBit ，该板卡即为掌控板。

右上角选择 高级试图。

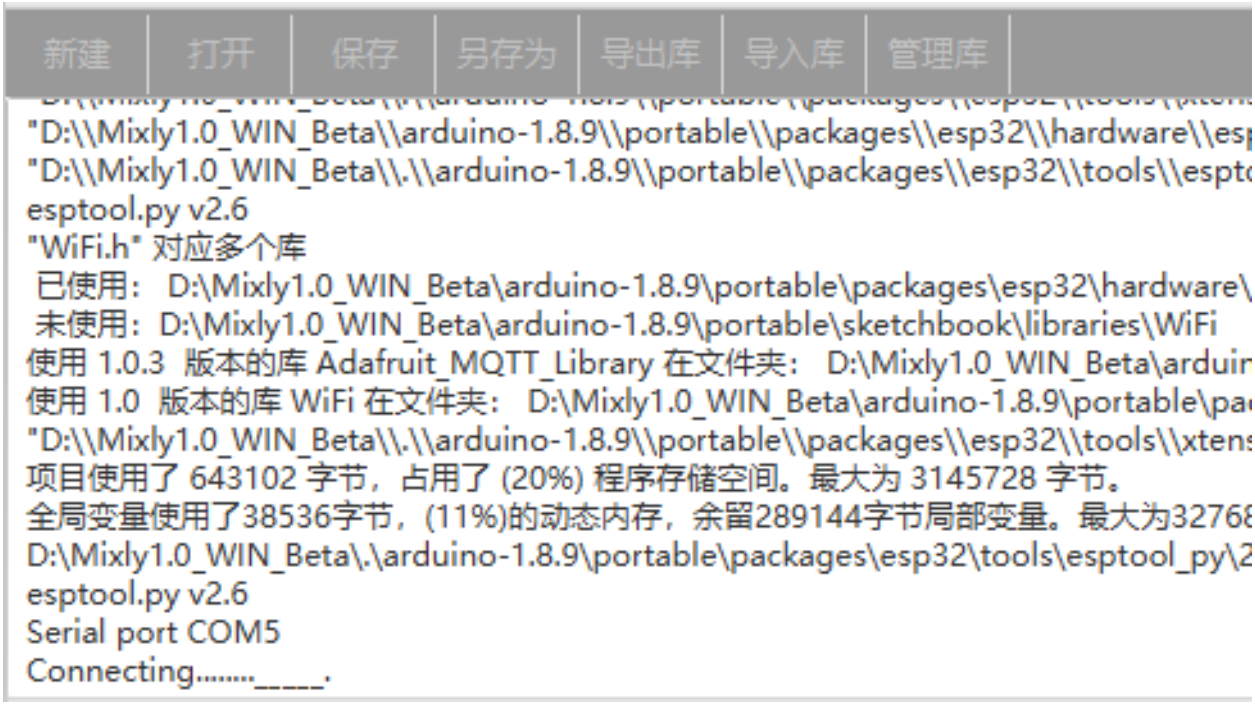
连接到 MQTT 服务器

修改 WiFi 信息，填写 MQTT 服务器信息，程序如下图所示：

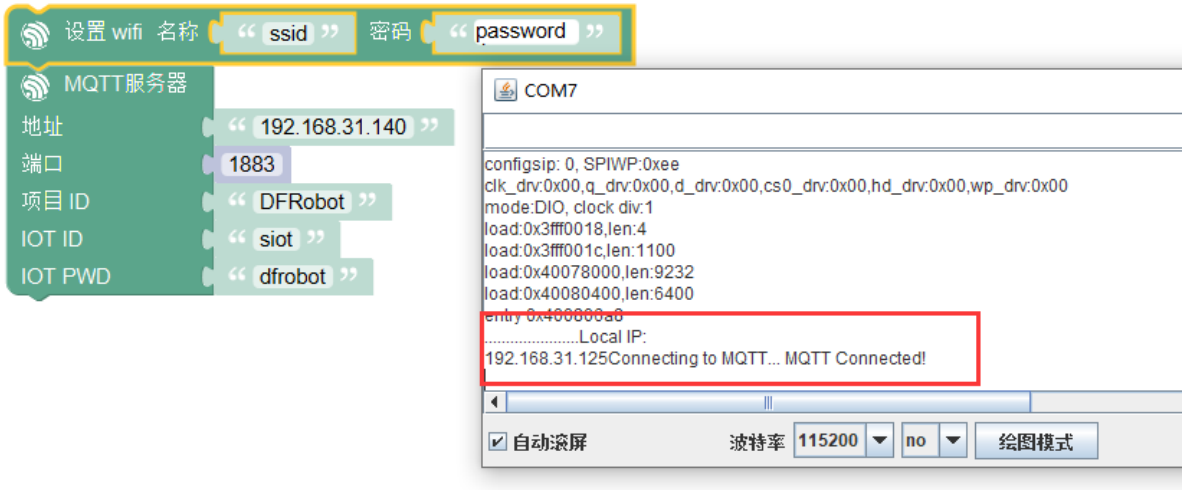


点击软件右下角上传，上传程序到掌控板。

在上传过程中，如果在编译信息框中看到如下信息，则需要按住掌控板的 A 键才能上传。（老版本掌控板有此问题，新版本可以直接上传）

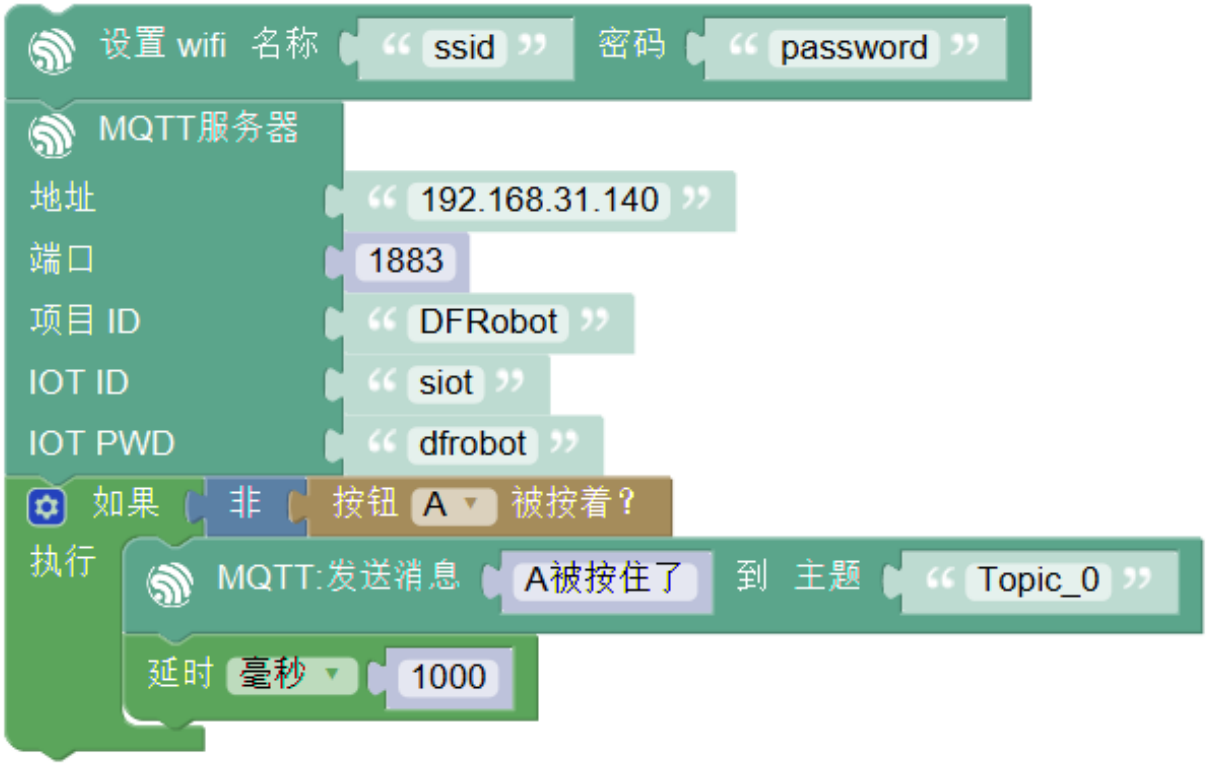


上传成功后，打开串口监视器，可以看到如下信息，说明掌控板连接网络正常，首先是连接 wifi 正常，掌控板得到 IP 地址，然后是连接 MQTT 服务器正常。



MQTT 发送消息

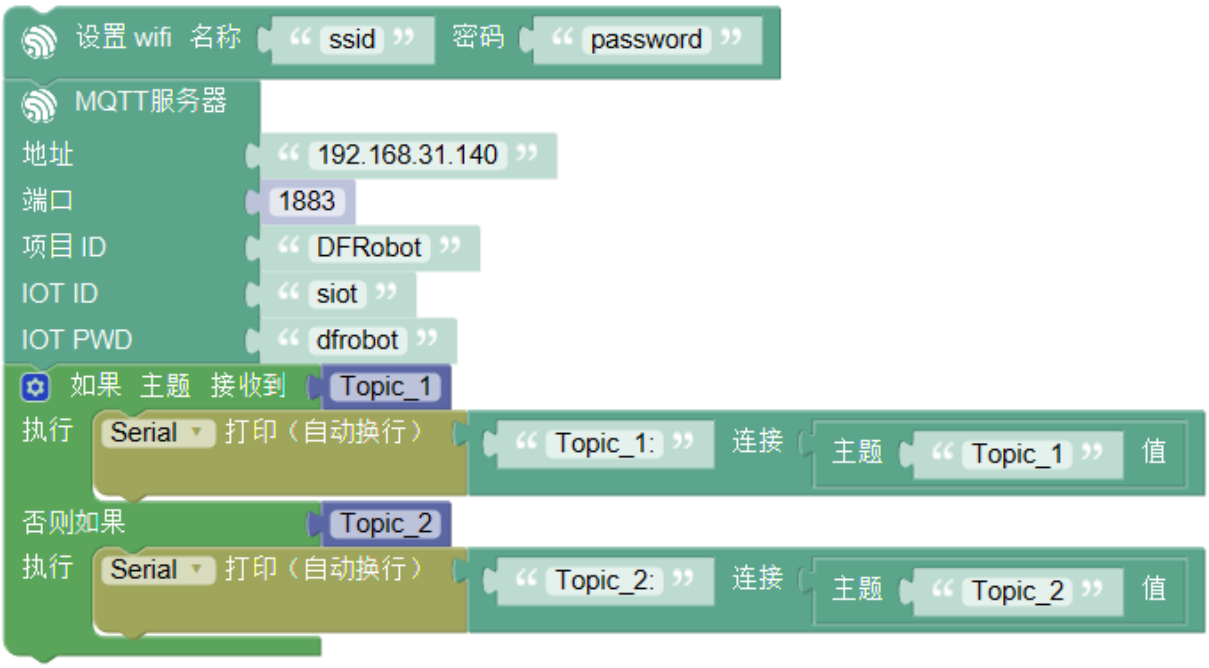
当某事件发生时，发送消息到某个主题。



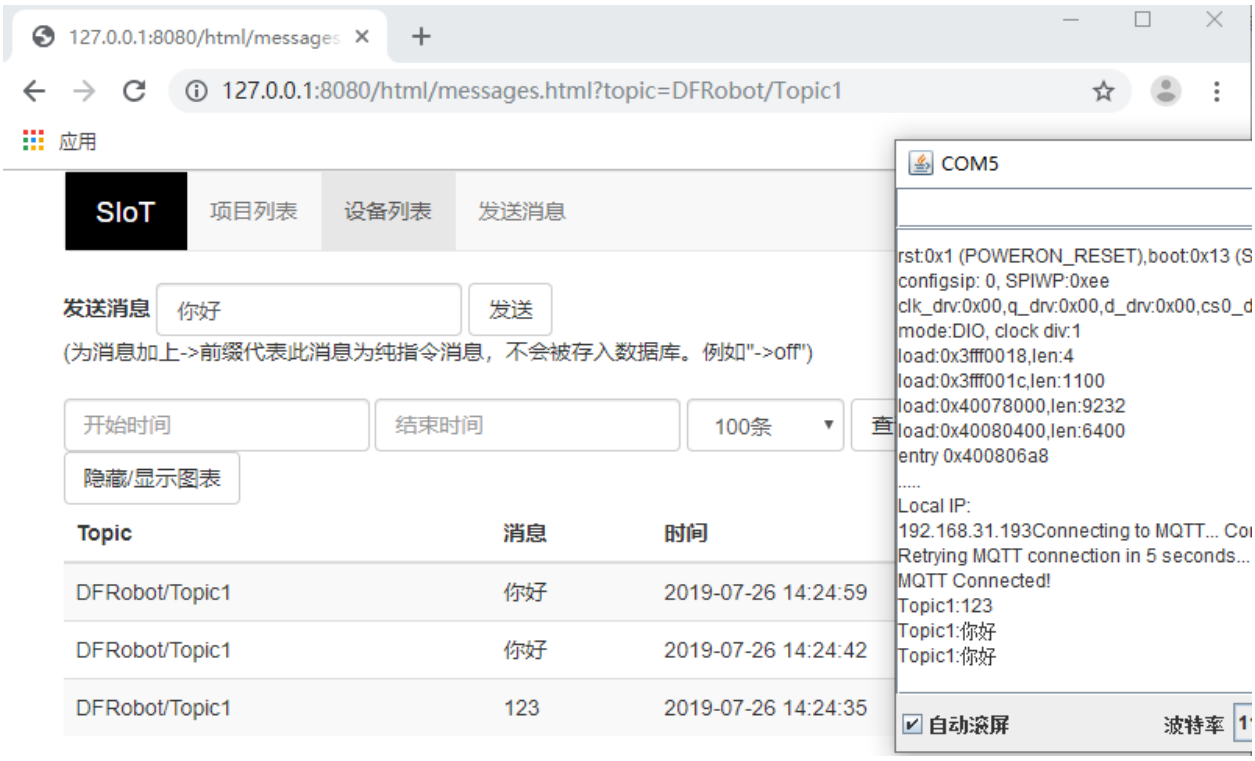
程序上传之后，按下掌控板上的 A 键，在网页端可以收到消息，如下图所示。

MQTT 订阅消息

当从某个主题接收到消息时，打印该消息。



通过 siot 网页发送消息，可以在 mixly 的串口监视器中查看到接收到的值。



3.5.5 掌控板的 MQTT 代码（基于 BXY）

BXY 是一款运行于 Windows 平台的 MicroPython 编程 IDE，界面简洁，操作便利。BXY 是浙教版普通高中信息技术教材的为众多 micro:bit 和掌控板爱好者提供了一个简洁实用的平台。

下载地址：http://docs.dfrobot.com.cn/bxy/

代码说明：

这个实验需要 2 个掌控板，一个发布光线数据一个订阅光线数据，MQTT 服务器既可以用 EasyIot 物联网，也可以用 SIoT。

注意：使用 BXY 下载下面的代码，还需要另外添加一个库文件 Iot.py。本代码已经整合在 BXY 的范例中，将 BXY 升级到最新即可看到。

代码下载链接：https://github.com/vvlink/SIoT/tree/master/examples/%E6%8E%8C%E6%8E%A7%E6%9D%BF%E4%BB%A3%E7%A0%81/Bxy

发送消息

```
# 功能：发布光线数据
from mpython import light
from Iot import Iot
from umqtt.simple import MQTTClient
from machine import Timer
import machine
import time
import json
import network

WIFI_SSID = 'yourSSID'# 替换成你的 WIFI 热点名称
WIFI_PASSWORD = 'yourPASSWD'# 替换成你的 WIFI 热点密码

IOT_SERVER = "server address" #EASYIOT 的服务器为 iot.dfrobot.com.cn；Siot 地址为用户搭建的服务器的 ip 地址，例如：192.168.0.100
IOT_PORT = 1883
IOT_ClientID = "your ClientID"# 替换成你的 ClientID，可为空
IOT_UserName = "your UserName"# 替换成你的 UserName
IOT_PassWord = "your PassWord"# 替换成你的 PassWord
IOT_pubTopic = 'your PubTopic' # 如果是 siot，自定义的 topic 中需要添加"/"，例如:"abc/abc"

myIot = Iot(IOT_SERVER, IOT_UserName, IOT_ClientID, IOT_PassWord)
client = MQTTClient(myIot.client_id, myIot.mqttserver, port = IOT_PORT, user = myIot.
↪username, password = myIot.password)
```

(下页继续)

(续上页)

```
tim1 = Timer(1)

def connectWIFI():
  station = network.WLAN(network.STA_IF)
  station.active(True)
  station.connect(WIFI_SSID,WIFI_PASSWORD)
  while station.isconnected() == False:
    pass
  print('Connection successful')
  print(station.ifconfig())

def restart():
  time.sleep(10)
  machine.reset()

def check(_):
  try:
    msg = {}
    client.check_msg()
    msg["light"] = light.read()
    print(json.dumps(msg))
    client.publish(IOT_pubTopic,json.dumps(msg))
    lastTime = time.time()
  except OSError as e:
    tim1.deinit()
    restart()

connectWIFI()
client.connect()

tim1.init(period=5000, mode=Timer.PERIODIC,callback=check)
while True:
  pass
```

订阅消息

```
# 功能：订阅光线数据
from mpython import *
from Iot import Iot
from umqtt.simple import MQTTClient
```

(下页继续)

(续上页)

```
from machine import Timer
from machine import Pin
import machine
import time
import json
import network

WIFI_SSID = 'yourSSID'# 替换成你的 WIFI 热点名称
WIFI_PASSWORD = 'yourPASSWD'# 替换成你的 WIFI 热点密码

IOT_SERVER = "server address" #EASYIOT 的服务器为 iot.dfrobot.com.cn; Siot 地址为用户搭建
的服务器的 ip 地址, 例如: 192.168.0.100
IOT_PORT = 1883
IOT_ClientID = "your ClientID"# 替换成你的 ClientID, 可为空
IOT_UserName = "your UserName"# 替换成你的 UserName
IOT_PassWord = "your PassWord"# 替换成你的 PassWord
IOT_subTopic = 'your SubTopic' # 如果是 siot, 自定义的 topic 中需要添加"/", 例如:"abc/abc"

myIot = Iot(IOT_SERVER, IOT_UserName, IOT_ClientID, IOT_PassWord)
client = MQTTClient(myIot.client_id, myIot.mqttserver, port = IOT_PORT, user = myIot.
↪username, password = myIot.password)

tim1 = Timer(1)

def connectWIFI():
  station = network.WLAN(network.STA_IF)
  station.active(True)
  station.connect(WIFI_SSID,WIFI_PASSWORD)
  while station.isconnected() == False:
    pass
  print('Connection successful')
  print(station.ifconfig())

def sub_cb(topic,msg):
  print((topic,msg))
  if topic == b'light':
    try:
      print(type(msg))
      print("msg=%s"%str(msg))
      light= json.loads(msg)["light"]
```

(下页继续)

(续上页)

```
      oled.DispChar("接收到对方光强度",0,0)
      oled.DispChar("%s"%str(light),64,16)
      oled.show()
      oled.fill(0)
      v=light//16
      rgb[0] = (v,v,v)
      rgb[1] = (v,v,v)
      rgb[2] = (v,v,v)
      rgb.write()
    except:
      print("error msg:%s"%msg)
  else:
    print("other topic=%s msg=%s"%(topic,msg))

def restart():
  time.sleep(10)
  machine.reset()

def check(_):
  try:
    client.check_msg()
  except OSError as e:
    tim1.deinit()
    restart()

oled.DispChar("正在连接网络...",0,0)
oled.show()
oled.fill(0)
connectWIFI()

client.set_callback(sub_cb)
client.connect()
client.subscribe(IOT_subTopic)

tim1.init(period=1000, mode=Timer.PERIODIC,callback=check)

while True:
  pass
```

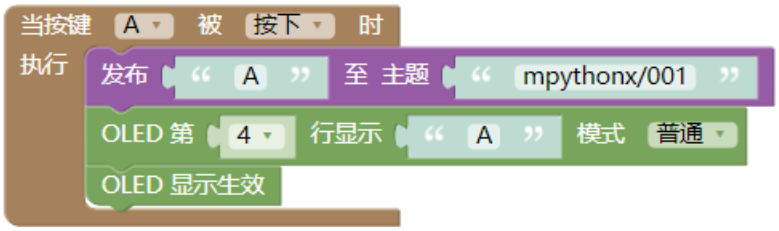
3.5.6 掌控板的 MQTT 代码（基于 mPythonX）

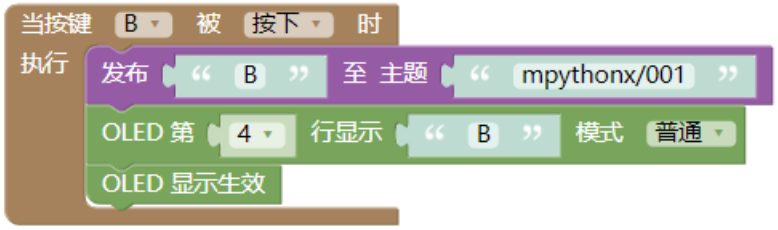
图形化代码拖入后无法正常显示，请先检查左边控件栏中，物联网箭头下是否有 MQTT 模块，如无该模块请检查软件版本。

注意：因为 MicroPython 的 MQTT 库（simple.py）的缺陷，我们发现如果 SIoT 运行在 Windows 系统上，在 mPythonX 的发送消息语句后加上延时，将会导致一定时间后消息发送不成功！在 MQTT 库未升级之前，请使用定时器来发送消息。

代码下载地址：https://github.com/vvlink/SIoT/tree/master/examples/%E6%8E%8C%E6%8E%A7%E6%9D%BF%E4%BB%A3%E7%A0%81/mPythonX/%E5%8F%91%E9%80%81%E6%B6%88%E6%81%AF

发送消息





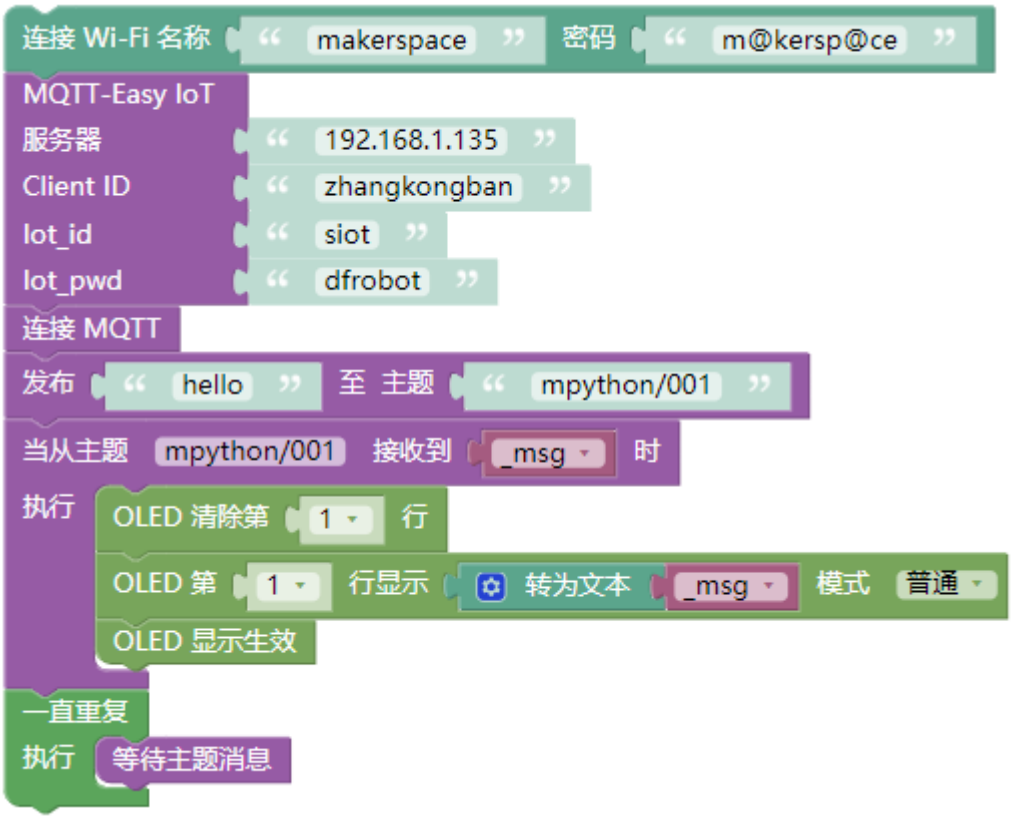

```
from mpython import *
import network
```

(下页继续)

(续上页)

```
from umqtt.simple import MQTTClient

my_wifi = wifi()
my_wifi.connectWiFi("makerspace", "m@kersp@ce")

mqtt = MQTTClient("zhangkong", "192.168.1.135", 1883, "siot", "dfrobot", keepalive=30)

try:
    mqtt.connect()
    print('Connected')
except:
    print('Disconnected')

def on_button_a_down(_):
    mqtt.publish("mpythonx/001", "A")
    oled.DispChar("A", 0, 48, 1)
    oled.show()

def on_button_b_down(_):
    mqtt.publish("mpythonx/001", "B")
    oled.DispChar("B", 0, 48, 1)
    oled.show()

button_a.irq(trigger=Pin.IRQ_FALLING, handler=on_button_a_down)

button_b.irq(trigger=Pin.IRQ_FALLING, handler=on_button_b_down)


oled.DispChar(my_wifi.sta.ifconfig()[0], 0, 0, 1)
oled.show()
```

订阅消息




```
from mpython import *
import network
from umqtt.simple import MQTTClient
from machine import Timer
import ubinascii

my_wifi = wifi()
my_wifi.connectWiFi("makerspace", "m@kersp@ce")

mqtt = MQTTClient("zhangkongban", "192.168.1.135", 1883, "siot", "dfrobot", keepalive=30)

try:
    mqtt.connect()
    print('Connected')
except:
    print('Disconnected')

def mqtt_topic_6d707974686f6e2f303031(_msg):
    oled.fill_rect(0, 0, 128, 16, 0)
```

(下页继续)

(续上页)

```
    oled.DispChar((str(_msg)), 0, 0, 1)
    oled.show()

def mqtt_callback(topic, msg):
    try:
        topic = topic.decode('utf-8', 'ignore')
        _msg = msg.decode('utf-8', 'ignore')
        eval('mqtt_topic_' + bytes.decode(ubinascii.hexlify(topic)) + '("' + _msg + '")')
    except: print((topic, msg))

mqtt.set_callback(mqtt_callback)

mqtt.subscribe("mpython/001")

def timer14_tick(_):
    mqtt.ping()

tim14 = Timer(14)
tim14.init(period=20000, mode=Timer.PERIODIC, callback=timer14_tick)


mqtt.publish("mpython/001", "hello")
while True:
    mqtt.wait_msg()
```

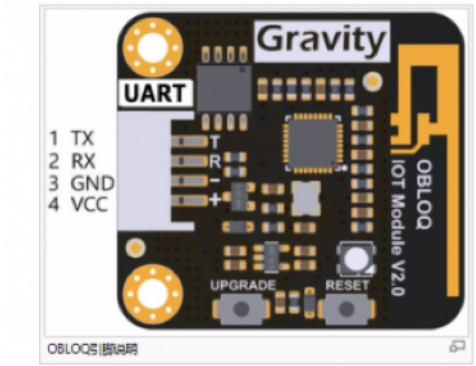
3.6 Arduino

Arduino 是一款便捷灵活、方便上手的开源电子原型平台。在创客眼里，Arduino 是一个造物的神器，基于 Arduino 设计的创客作品不计其数。

3.6.1 OBLOQ 模块介绍

OBLOQ 是一款基于 ESP8266 设计的串口转 WIFI 物联网模块，用以接收和发送物联网信息。接口简单，即插即用，适用于 3.3V~5V 的控制系统。OBLOQ 物联网模块当没有连接 wifi 的时候，OBLOQ 指示灯显示红色，正在连接 wifi 时显示蓝色，连接到 wifi 后，OBLOQ 指示灯显示绿色。

OBLOQ 物联网模块引脚说明：



引脚定义

标号	名称	功能描述
1	TX	串口发送端
2	RX	串口接收端
3	GND	地
4	VCC	电源

关于 OBLOQ 物联网模块的介绍：http://www.dfrobot.com.cn/goods-1577.html

3.6.2 典型案例

本案例将 Uno 板应用在家居物联网中，将温度传感器与 Uno 板连接，用于采集室内的温湿度数据，并将数据上传到 SIoT 中，结果将以数值与折线图的形式展现，便于主人监测室内温湿度的变化情况。

所需材料：Uno 板，温度传感器，OBLOQ 物联网模块

使用软件：Mind+（所有代码可以转成）

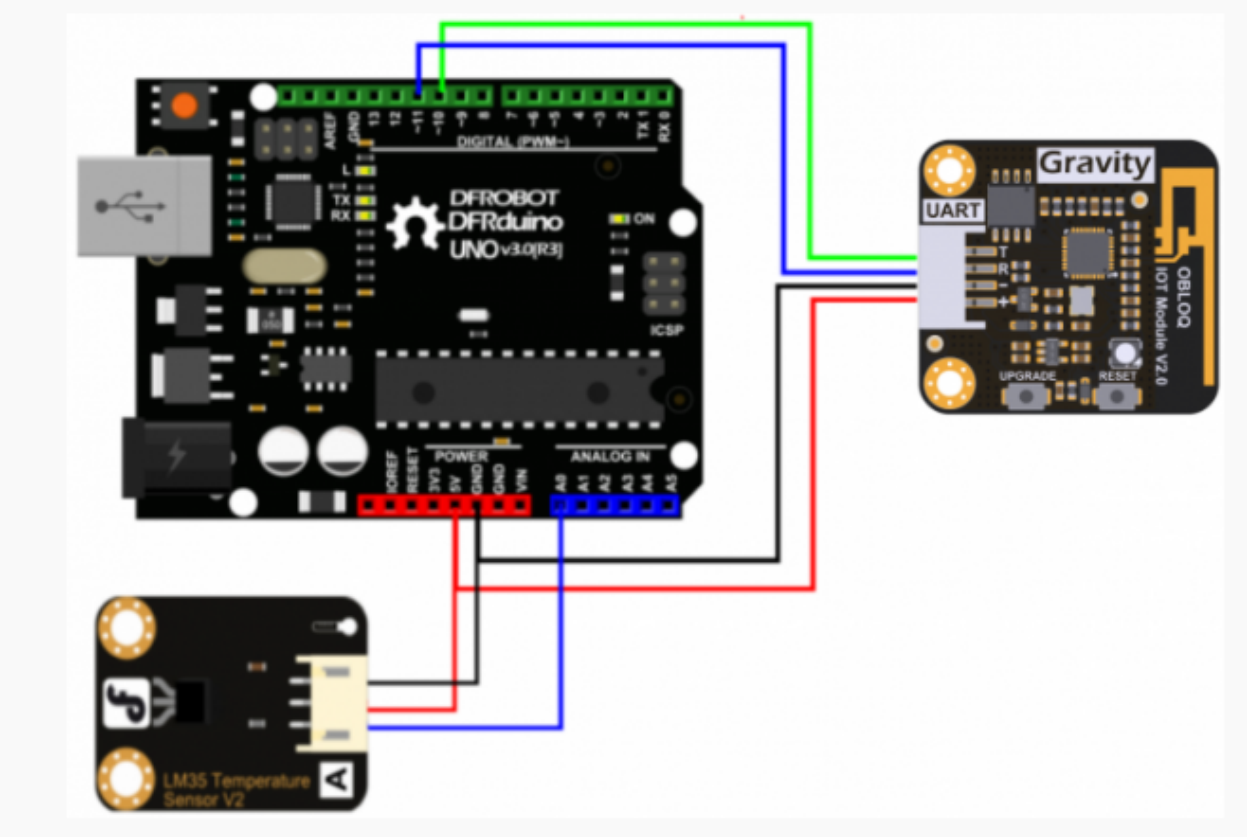
STEP1 Mind+ 软件设置

1、打开 Mind+ 软件（1.5.5 及以上版本），选择“上传模式”：

STEP2 硬件连线图

OBLOQ 模块：TX、RX、GND 和 VIN 引脚分别连接到 D10、D11、GND 和 VCC 引脚。

温度传感器 LM35：A0 引脚。



可通过修改程序来自定义相关引脚。

STEP3 编写程序

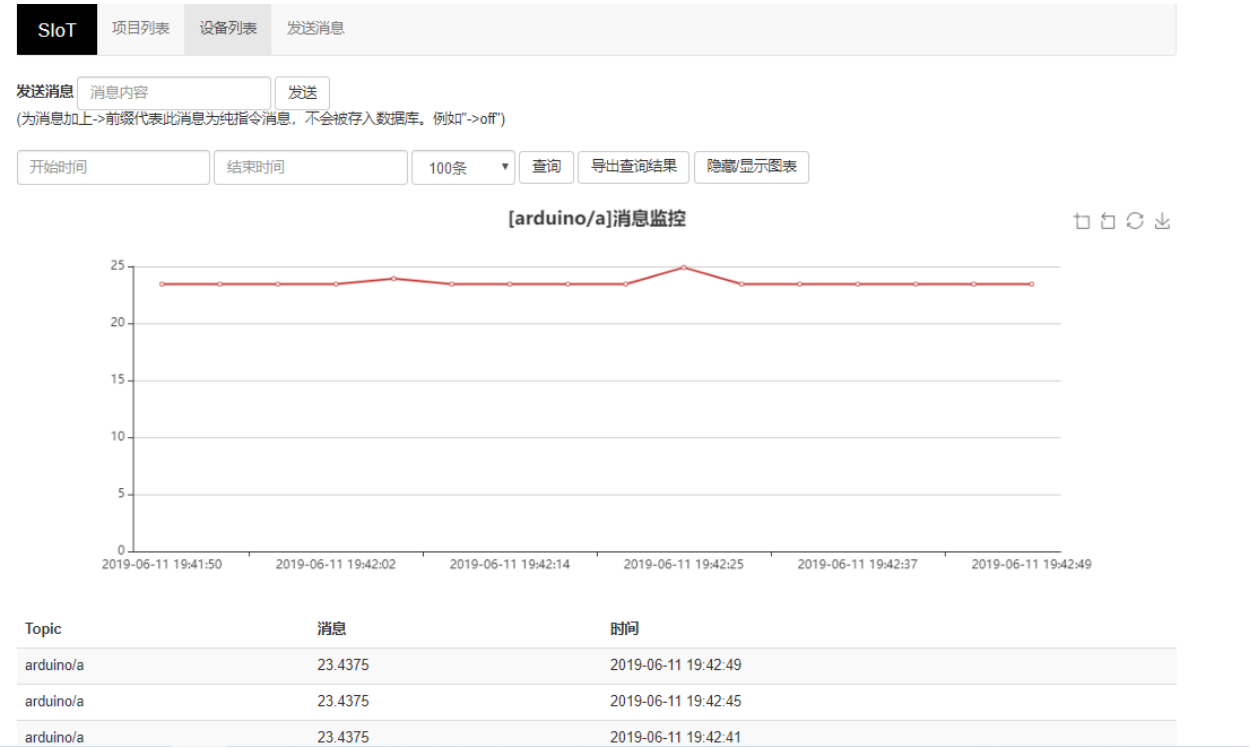
STEP4 功能实现

Uno 板在下载完程序后，OBLOQ 物联网模块指示灯显示为绿色，表示成功与 Uno 板连接成功正常工作。

程序运行时，若无法连接 OBLOQ 物联网模块（指示灯不为绿色），先检查参数有没有填错，例如 ip 错误、Topic 中有没有斜杠，依旧无法连接的话，可尝试关闭电脑防火墙，重新上传程序；若依旧不成功需要查看 OBLOQ 模块的接线对应的管脚是否连接正确。

打开 SIoT 网页端，可以在“设备列表”下看到对应的 Topic 信息。

1、点击上图中“a”后的“查看消息”，可以看到温度传感器实时收集的温度值。



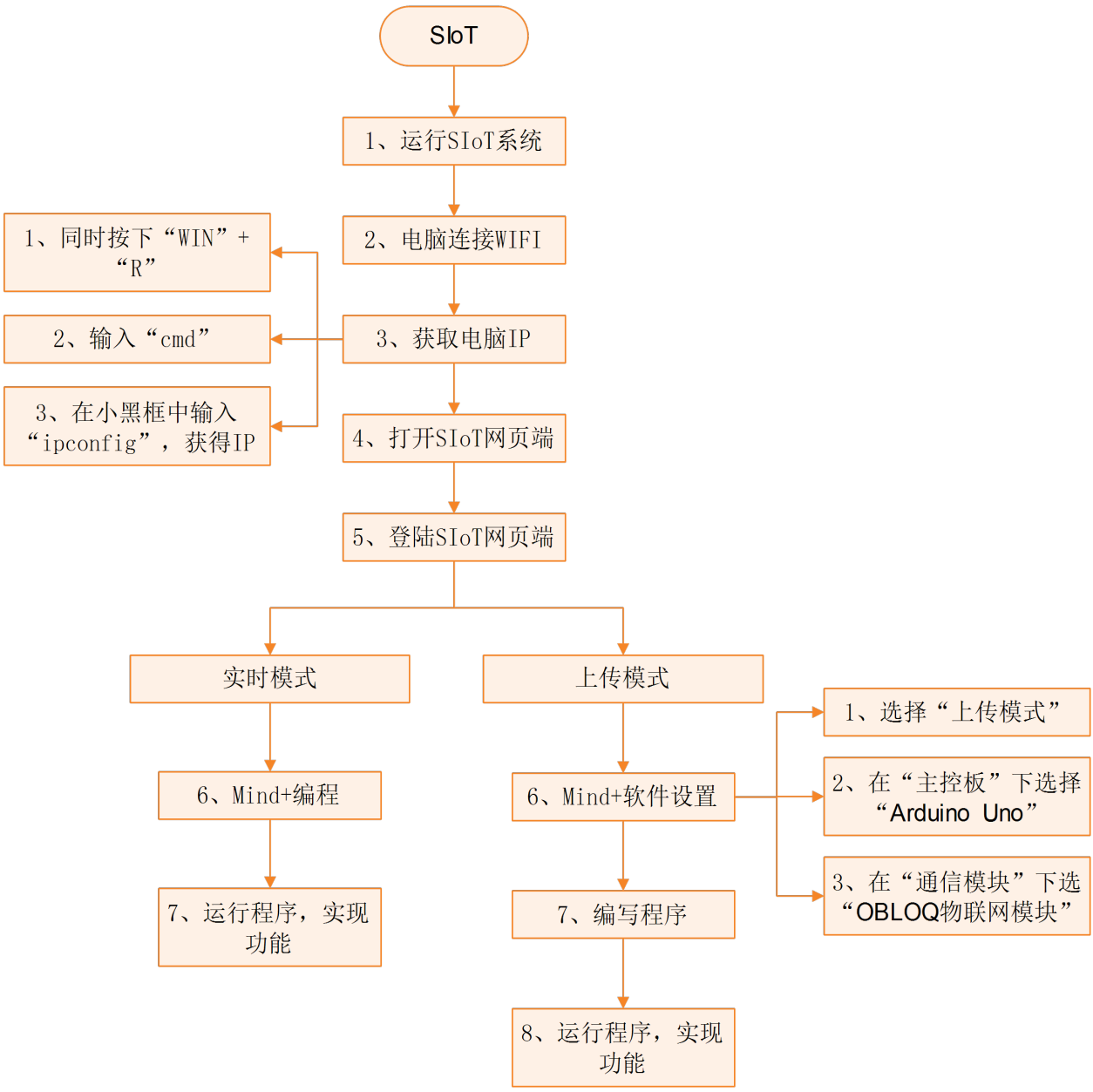
2、点击“b”后的查看消息

在弹出窗口中发送消息“ON”，可以看到 Uno 板子的 LED 被点亮。

同理，发送消息“OFF”，可以看到 Uno 板子的 LED 被点亮。

3.6.3 操作流程归纳

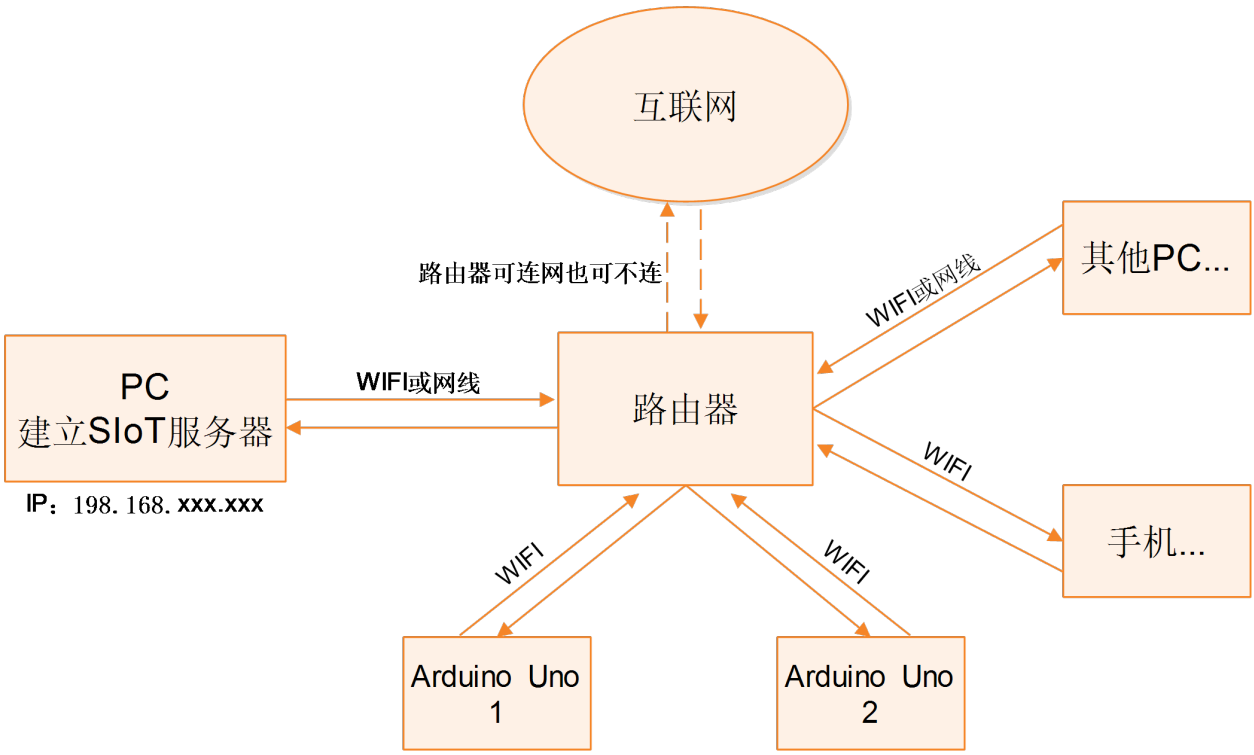
Arduino 和 SIoT 互联的一般过程图示。



3.6.4 物联网框图

以路由器建立无线局域网为例，通过下面这张图可以说明 SIoT 的作用原理。

在一台电脑上建立 SIoT 服务器后，其他设备在知道路由器分配给这台电脑的 IP 地址后，可以利用 WIFI 访问 SIoT 服务器。这些设备可以是电脑、手机、Uno 板等。



3.6.5 OBLOQ 模块的常见问题

1) OBLOQ 指示灯一直显示蓝色：

表示 OBLOQ 正在连接 wifi，需要一定时间，如果超过一分钟依然显示蓝灯，则可能为 wifi 账号密码设置错误，请检查程序。

2) OBLOQ 指示灯一直显示紫色：

表示 OBLOQ 的 wifi 连接成功但是 mqtt 异常断开，尝试检查所在 wifi 是否断网，也有可能 easyiot 服务器问题，等待一会儿再连接或联系论坛管理员。

3) OBLOQ 指示灯一直显示红色：

表示 OBLOQ 的 wifi 连接不成功，尝试检查是否 tx 和 rx 接反了（调换一下 tx 和 rx 接线顺序），或者是 wifi 有问题（使用手机开热点，不要用中文 WIFI 名称），然后就是参数有没有填错（物联网网站里面的参数）。

需要了解更多细节请参考：http://mc.dfrobot.com.cn/thread-281129-1-1.html

3.7 micro:bit

Micro:bit 是由英国 BBC 公司所推出的面向青少年编程教育的微型计算机。Micro:bit 电路板上集成了 LED 灯、两个可编程按钮、加速度传感器、磁力传感器以及蓝牙等常用设备，采用 micro USB 口供电，可外接电池盒，底部有多个环孔连接器，可用于控制外接设备。

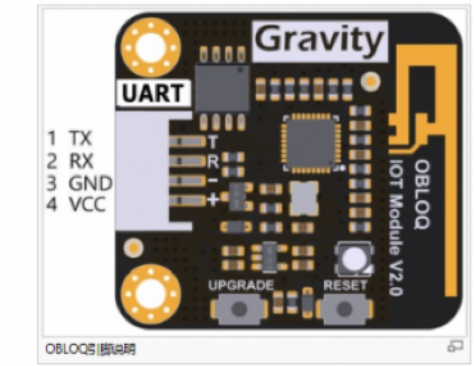
此外，Micro:bit 拥有在线的编程网站（https://makecode.microbit.org/），可通过图形化的编程界面，以及支持 python 或 Javascript 等多种编程语言的软件进行编程，例如：BXY、Mind+ 等。

micro:bit 购买网址：http://www.dfrobot.com.cn/goods-1395.html

3.7.1 OBLOQ 模块介绍

OBLOQ 是一款基于 ESP8266 设计的串口转 WIFI 物联网模块，用以接收和发送物联网信息。接口简单，即插即用，适用于 3.3V~5V 的控制系统。OBLOQ 物联网模块当没有连接 wifi 的时候，OBLOQ 指示灯显示红色，正在连接 wifi 时显示蓝色，连接到 wifi 后，OBLOQ 指示灯显示绿色。

OBLOQ 物联网模块引脚说明：



引脚定义

标号	名称	功能描述
1	TX	串口发送端
2	RX	串口接收端
3	GND	地
4	VCC	电源

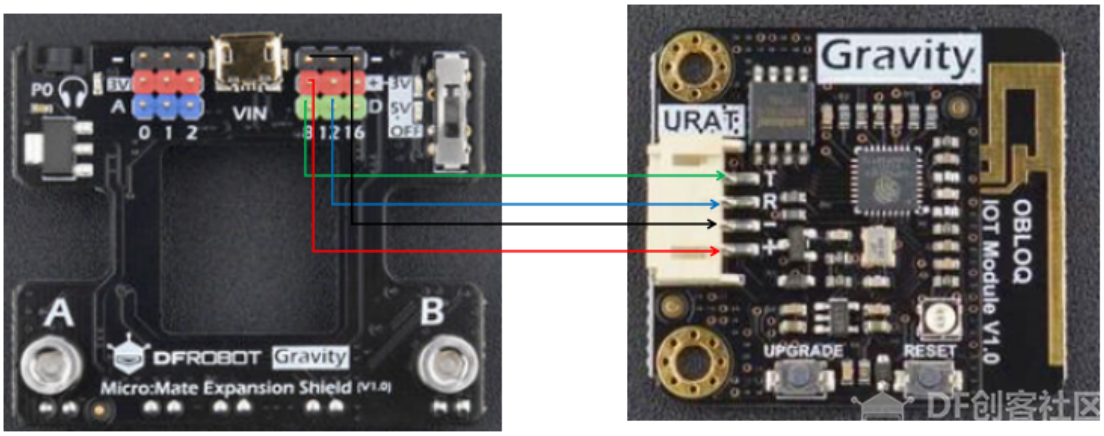
关于 OBLOQ 物联网模块的介绍：http://www.dfrobot.com.cn/goods-1577.html

3.7.2 典型案例

将 micro:bit 板应用在家庭物联网中：使用 LED 灯模块，并将其与 micro:bit 板进行连接，通过 SIoT 控制灯的亮灭以及闪烁，实现 SIoT 控制台灯的效果。所需材料：micro:bit 板、Micro:Mate（micro:bit 扩展板介绍详见 4.3.2）、LED 灯模块、OBLOQ 物联网模块

STEP1 OBLOQ 模块与 micro:bit 连线图

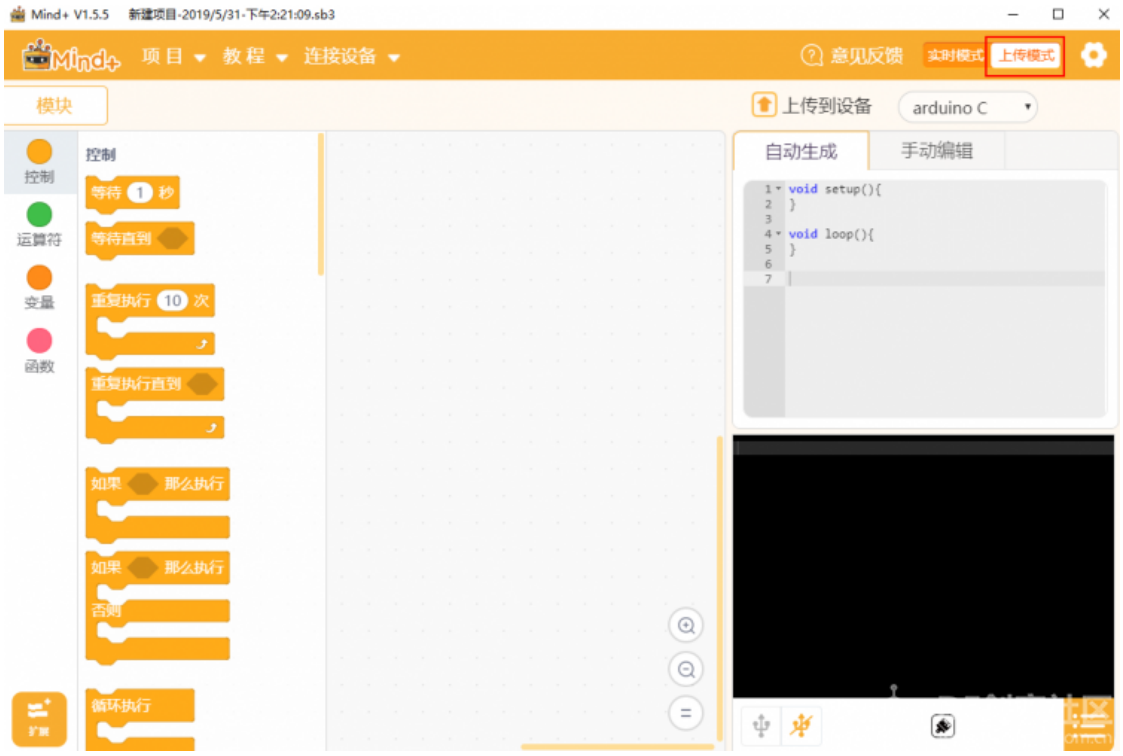
OBLOQ 模块：TX、RX、GND 和 VIN 引脚分别连接到 D8、D12、GND 和 VCC 引脚。LED 模块：D16 引脚 (对应 mind+ 中 P16)。



可通过修改程序来自定义相关引脚。

STEP2 Mind+ 软件设置

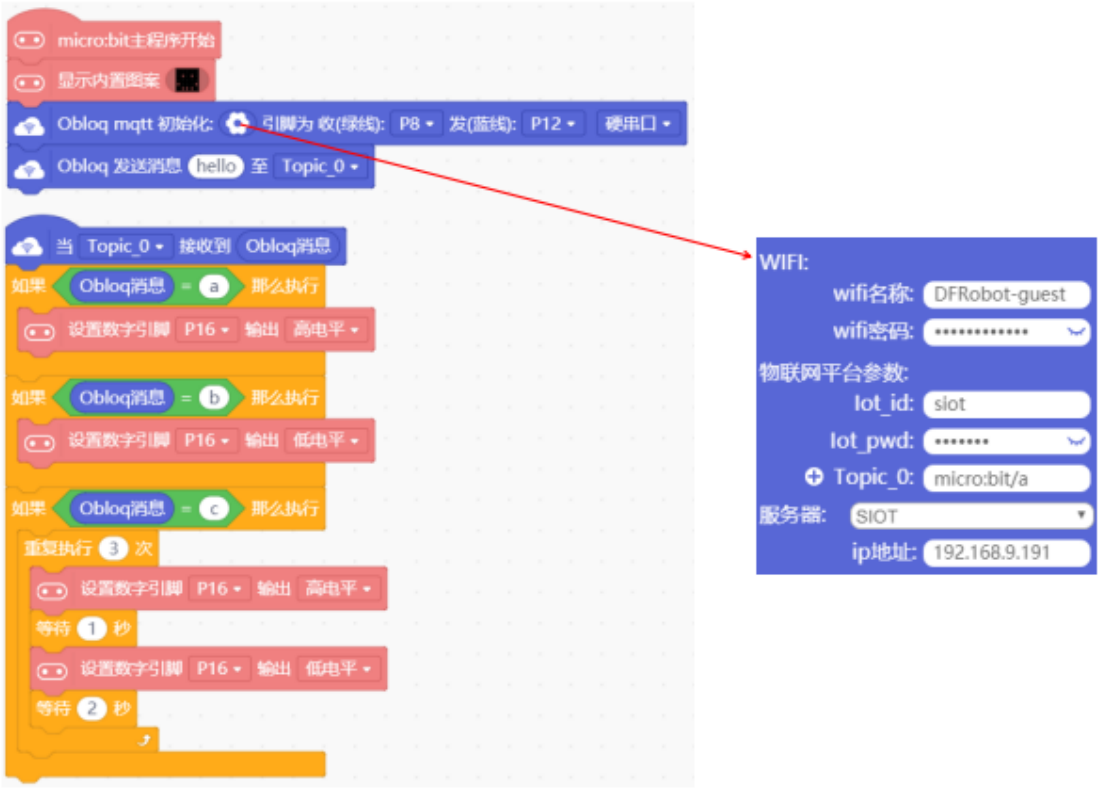
1、打开 Mind+ 软件（1.5.5 及以上版本），选择“上传模式”：



2、点击“扩展”，在“主控板”下选择“micro:bit 板”，在通信模块下选“OBLOQ 物联网模块”：

STEP3 编写程序

请参考如下代码。



注意：需要正确填写 Wi-Fi 的名称和密码，还有 SIoT 的 IP，用户名和密码。

代码说明：通过网页端发送“a”、“b”和“c”到 Topic_0（“micro:bit/a”），点亮 LED、关闭 LED 以及 LED 灯进行闪烁。

点击“上传到设备”，将程序下载到 micro:bit 板中。

STEP4 功能实现

micro:bit 板在下载完程序后，OBLOQ 物联网模块指示灯显示为绿色，表示成功与 micro:bit 板连接成功正常工作。

- 程序运行时，若无法连接 OBLOQ 物联网模块（指示灯不为绿色），先检查参数有没有填错，例如 ip 错误、Topic 中有没有斜杠，依旧无法连接的话，可尝试关闭电脑防火墙，重新上传程序；若依旧不成功需要查看 OBLOQ 模块的接线对应的管脚是否连接正确。

打开 SIoT 网页端，可以在“设备列表”下看到对应的 Topic 信息（“micro:bit”）。

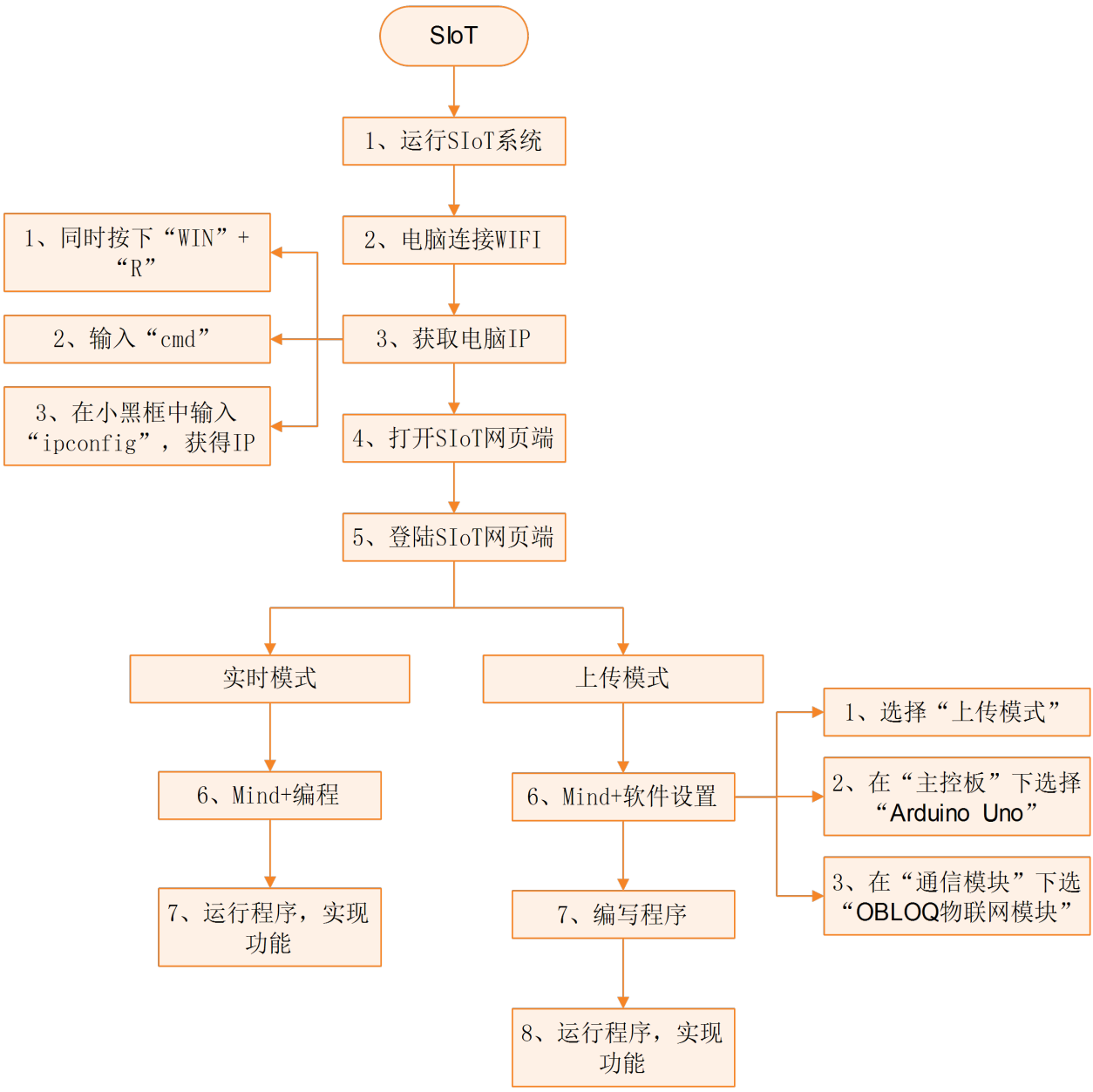
1、点击上图中“a”后的“查看消息”，在弹出窗口中发送消息“a”，可以看到 LED 灯模块中的 LED 被点亮。

2、发送消息“b”，可以看到 LED 灯模块中的 LED 被关闭。

3、同理，发送消息“c”，可以看到 LED 灯模块中的 LED 闪烁。

3.7.3 操作流程归纳

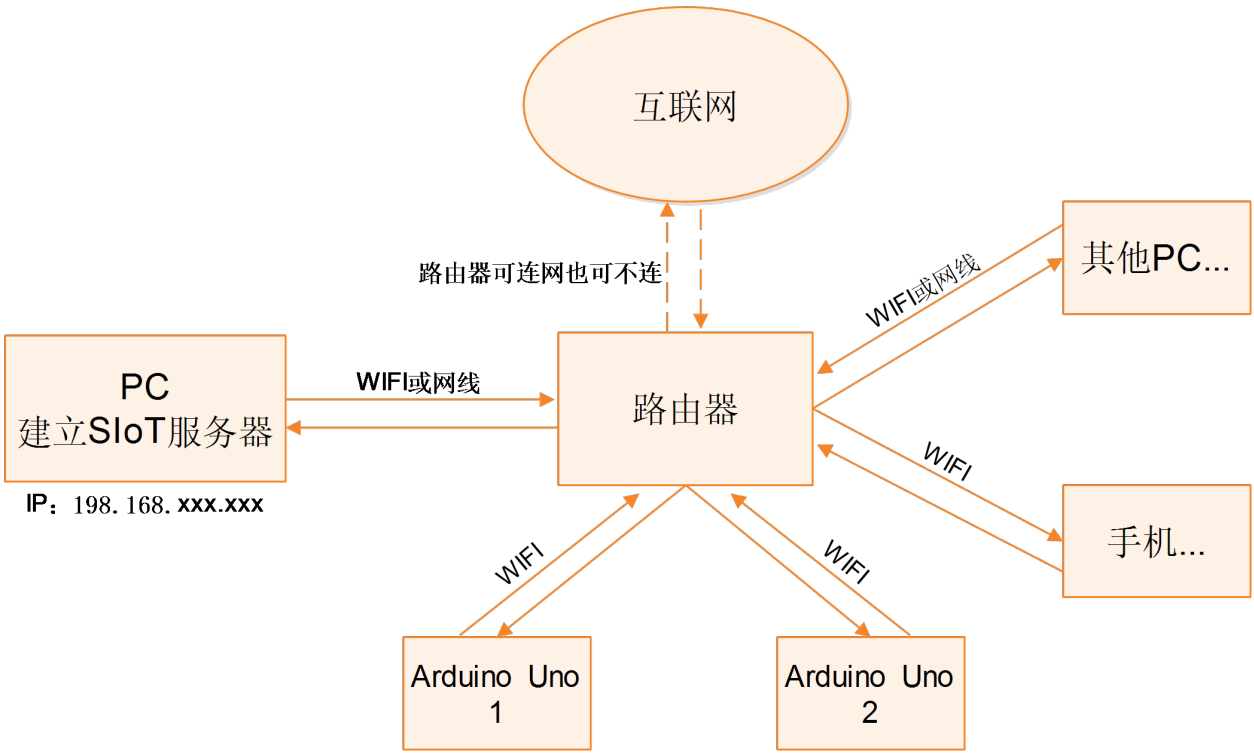
Arduino 和 SIoT 互联的一般过程图示。



3.7.4 物联网框图

以路由器建立无线局域网为例，通过下面这张图可以说明 SIoT 的作用原理。

在一台电脑上建立 SIoT 服务器后，其他设备在知道路由器分配给这台电脑的 IP 地址后，可以利用 WIFI 访问 SIoT 服务器。这些设备可以是电脑、手机、Uno 板等。



3.7.5 Micro:Mate（micro:bit 扩展板）介绍

Micro:Mate 是一款为 micro:bit 设计的微型多功能 IO 传感器扩展板，其彩色 3Pin 接口支持 DFRobot 百款 Gravity 系列电子模块，可直插传感器或电子元件，省去了繁琐的鳄鱼夹接插步骤，为 micro:bit 添加新的玩法和扩展可能性。扩展板体积小巧，完成后仅增加 micro:bit 板 5mm 的厚度，最大限度地集成了 micro:bit 的常用功能，板载 3.5mm 耳机插口，支持 3 路模拟输入和 3 路数字输入输出，支持 3V 与 5V 两种驱动电压，使 micro:bit 兼容 5V 数字输入输出元件。扩展板 USB 口可为大功率设备提供外接供电，保证舵机、电机等设备能够正常运行。

Micro:Mate 引脚说明：

3.7.6 OBLOQ 模块的常见问题

1) OBLOQ 指示灯一直显示蓝色：

表示 OBLOQ 正在连接 wifi，需要一定时间，如果超过一分钟依然显示蓝灯，则可能为 wifi 账号密码设置错误，请检查程序。

2) OBLOQ 指示灯一直显示紫色：

表示 OBLOQ 的 wifi 连接成功但是 mqtt 异常断开，尝试检查所在 wifi 是否断网，也有可能 easyiot 服务器问题，等待一会儿再连接或联系论坛管理员。

3) OBLOQ 指示灯一直显示红色：

表示 OBLOQ 的 wifi 连接不成功，尝试检查是否 tx 和 rx 接反了（调换一下 tx 和 rx 接线顺序），或者是 wifi 有问题（使用手机开热点，不要用中文 WIFI 名称），然后就是参数有没有填错（物联网网站里面的参数）。

需要了解更多细节请参考：http://mc.dfrobot.com.cn/thread-281150-1-1.html

3.8 App Inventor2

App Inventor 2 是一个基于云端的，以图形化形式编程的手机应用程序开发环境。它将枯燥的代码编程方式转变为积木式的图形化编程，同时不同功能代码的积木颜色也不同，这使手机应用程序的开发变得简单而有趣。即使不懂得编程语言的人，也可以开发出属于自己的手机应用程序。

广州市教育信息中心的 App Inventor 2 服务器地址：http://app.gzjkw.net。

我们可以通过 App Inventor2 来实现手机和 SIoT 之间的通信，从而实现以 MQTT 的方式和智能终端互联。

3.8.1 准备工作

1. 本案例以 SIoT 本地物联网平台为例进行示范 (SIoT 下载地址：https://github.com/vvlink/SIoT/tree/master/software/SIoT1.1)。

朋友们也可以注册一个物联网平台账号进行更远距离的控制，在线版物联网平台推荐：DFrobot 公司搭建的 EasyIot 平台（网址：http://iot.dfrobot.com.cn）。

2. 在广州市教育技术中心的 App Inventor2 服务器上注册一个账号，服务器网址为：http://app.gzjkw.net。

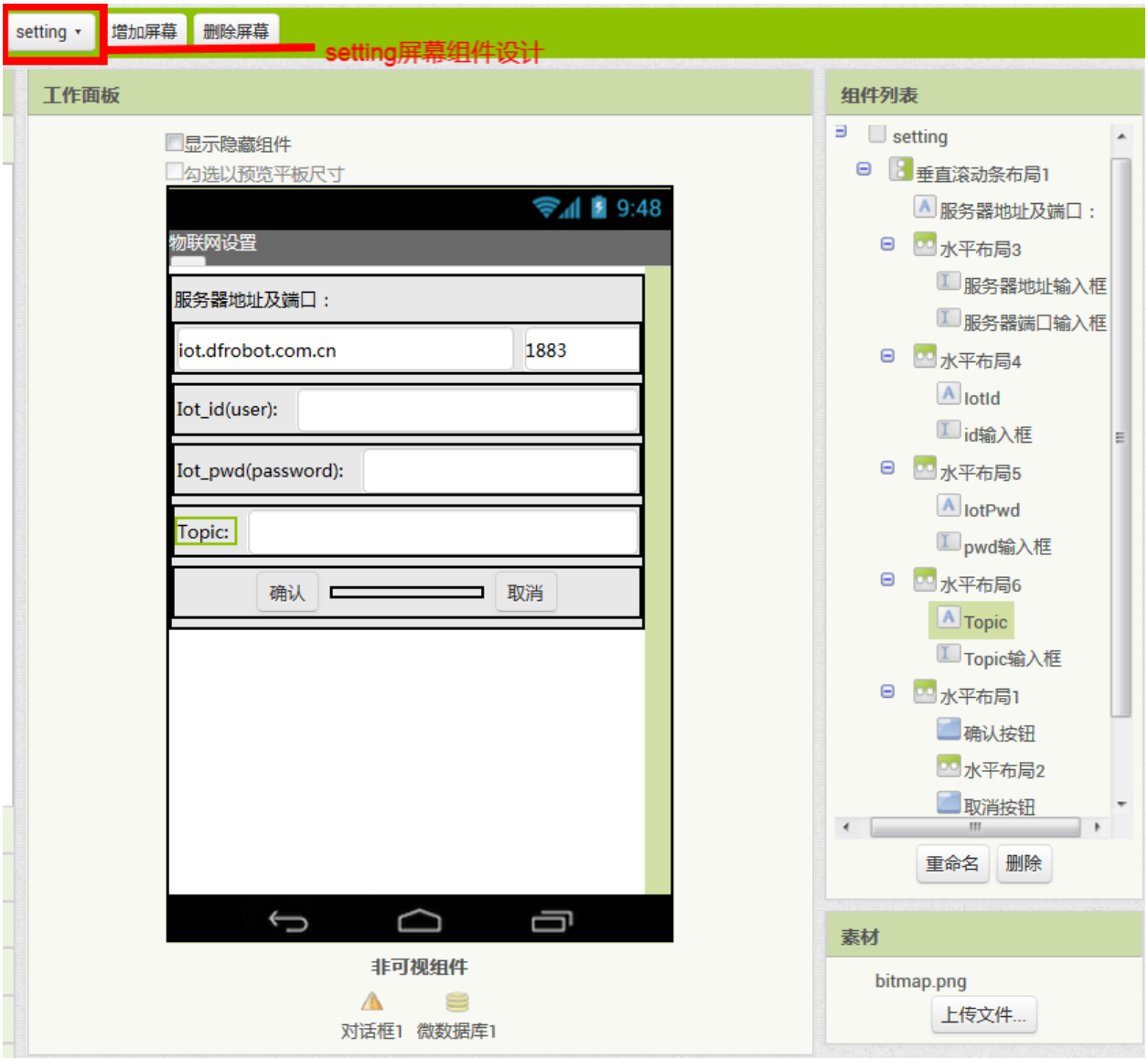
3. 下载安装 mind+1.5.5 版本的编程软件或 mpythonX 0.3.1 以上的版本。

4. 准备一块掌控板、一根 micro 口数据线、一台电脑。

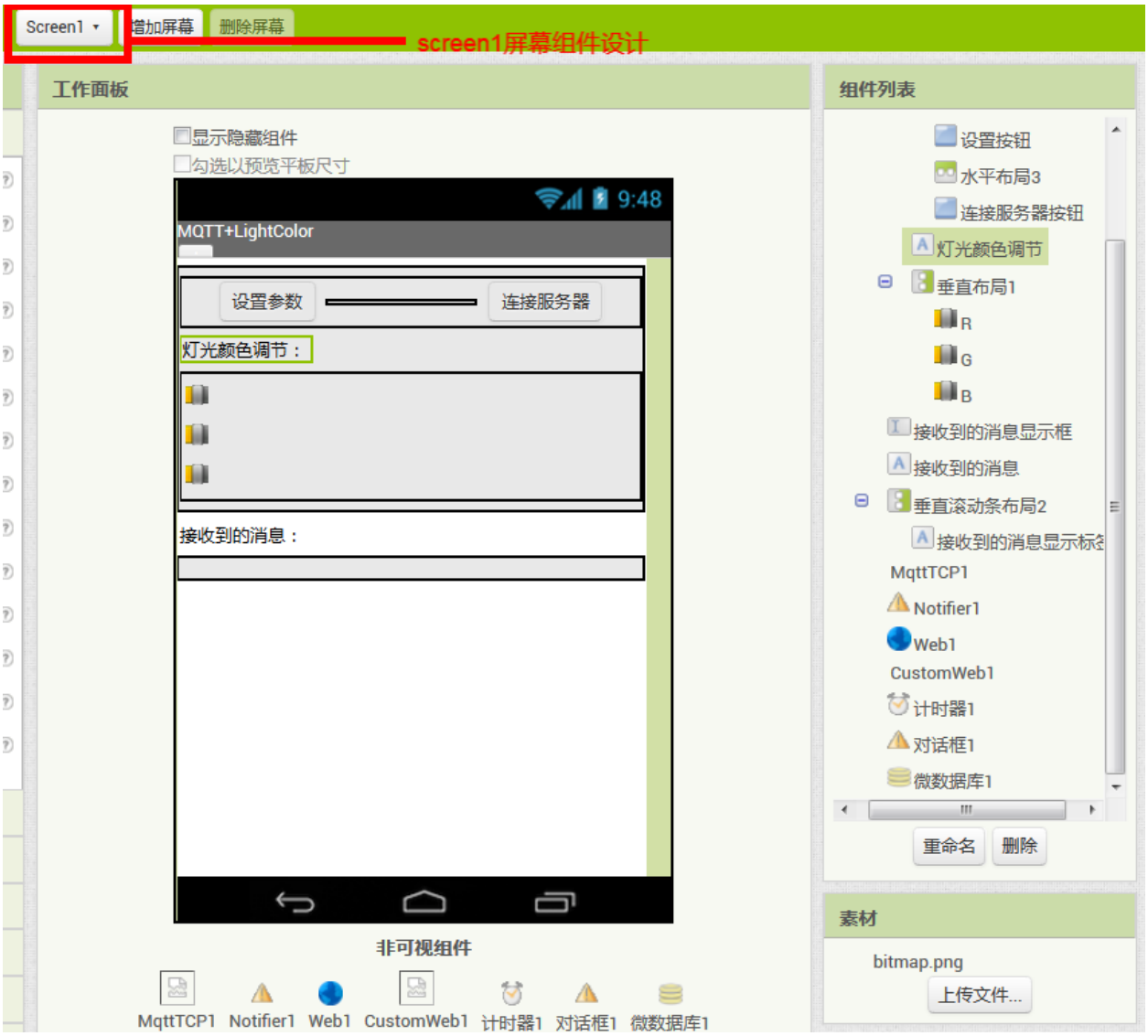
3.8.2 案例：手机控制掌控板全彩 LED 灯

1. 手机 APP 端编程 (工具平台：AppInventor2)：

组件设计-setting 屏幕组件设计图

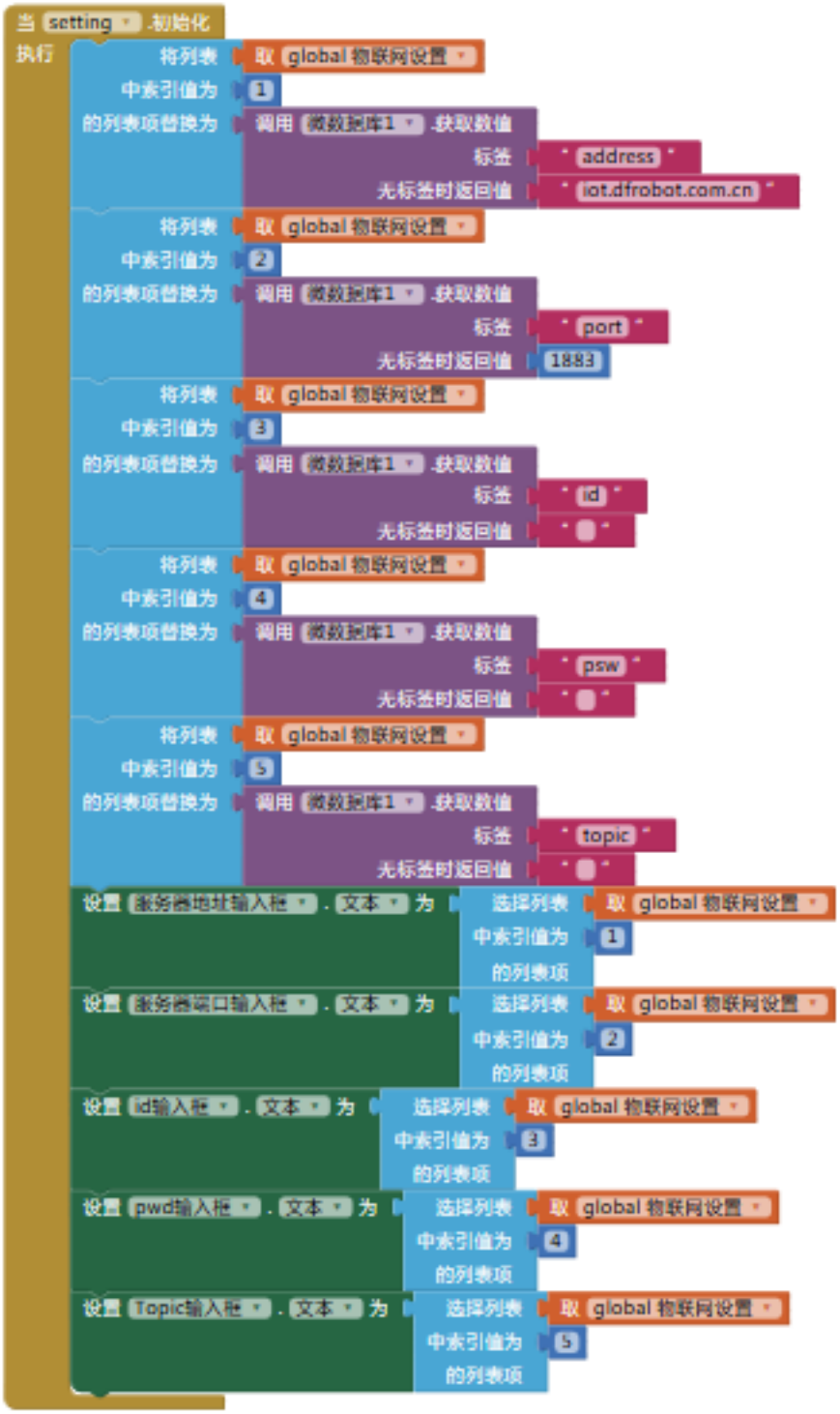


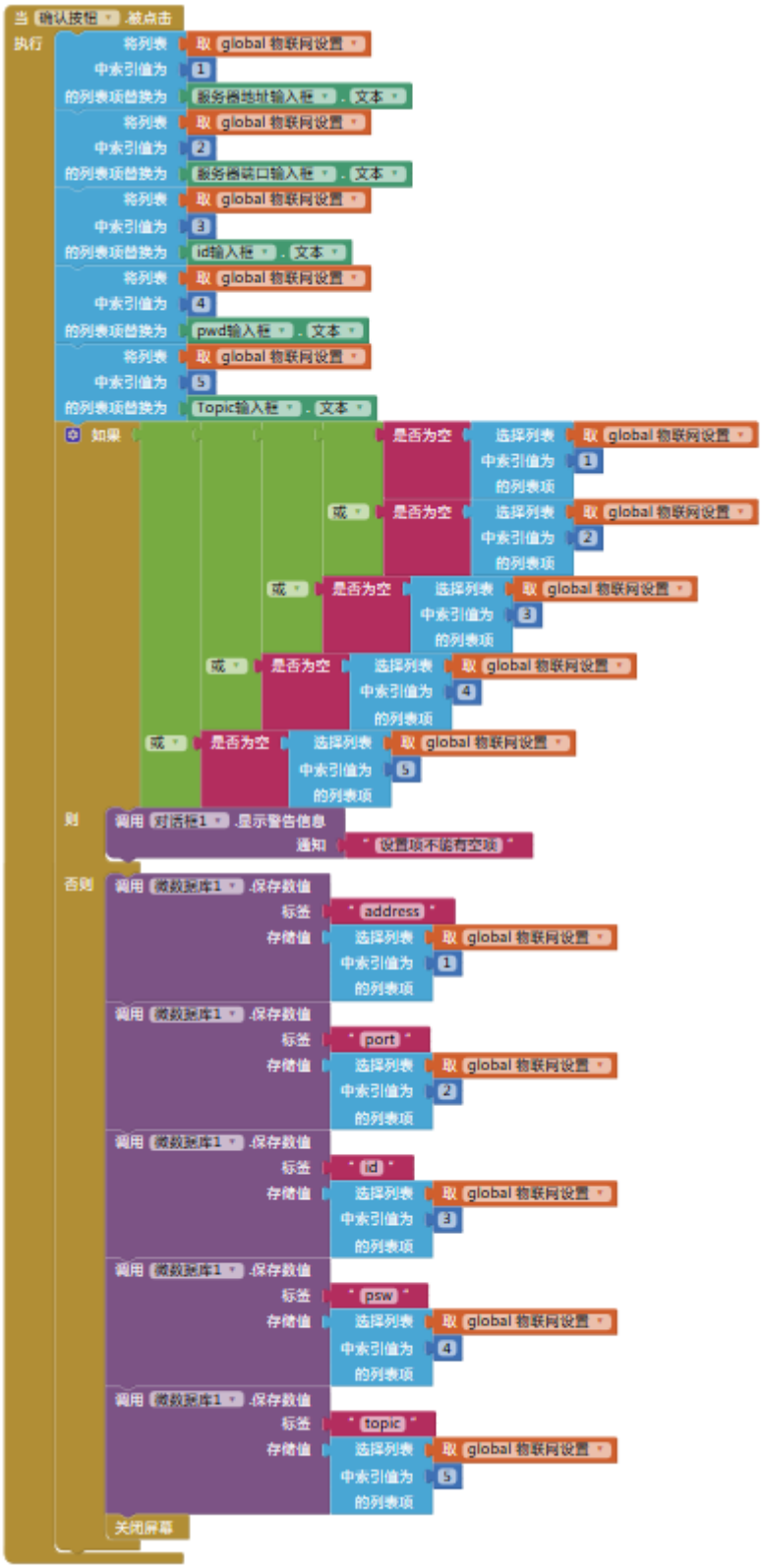
组件设计-screen1 屏幕组件设计图



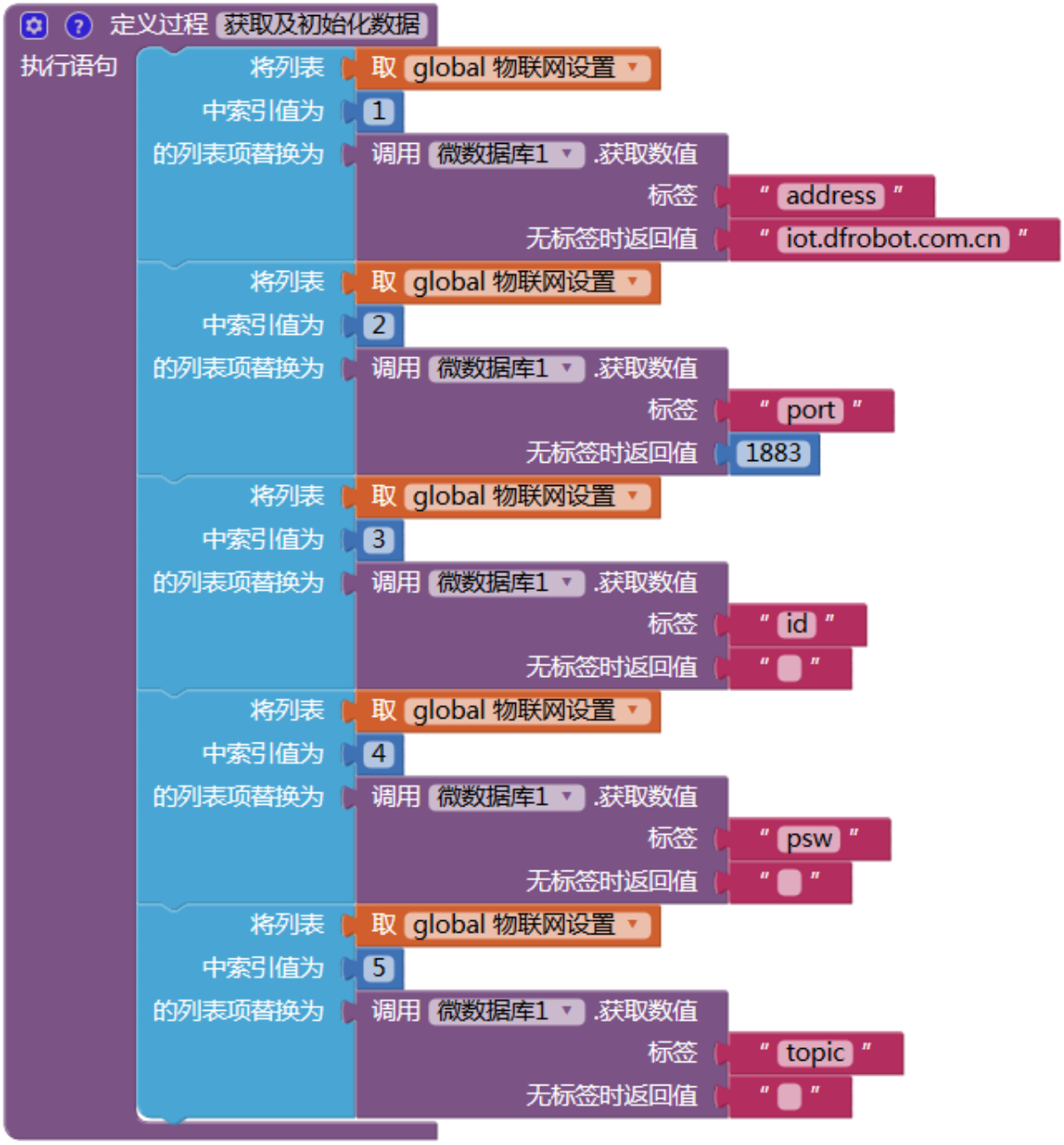
- 这里的非可视组件从左到右依次为：文件管理器、对话框、web 客户端、文件管理器、计时器、对话框、微数据库。

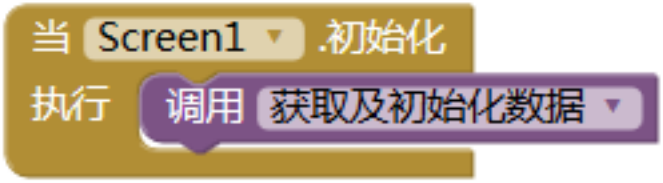
逻辑设计-setting 屏幕逻辑设计图

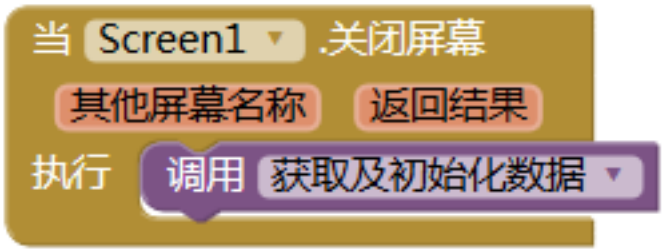

当 setting .初始化
执行
将列表 取 global 物联网设置
中索引值为 1
的列表项替换为 调用 微数据库1 .获取数值
标签 " address "
无标签时返回值 " iot.dfrobot.com.cn "
将列表 取 global 物联网设置
中索引值为 2
的列表项替换为 调用 微数据库1 .获取数值
标签 " port "
无标签时返回值 1883
将列表 取 global 物联网设置
中索引值为 3
的列表项替换为 调用 微数据库1 .获取数值
标签 " id "
无标签时返回值 " "
将列表 取 global 物联网设置
中索引值为 4
的列表项替换为 调用 微数据库1 .获取数值
标签 " psw "
无标签时返回值 " "
将列表 取 global 物联网设置
中索引值为 5
的列表项替换为 调用 微数据库1 .获取数值
标签 " topic "
无标签时返回值 " "
设置 服务器地址输入框 . 文本 为 选择列表 取 global 物联网设置 中索引值为 1 的列表项
设置 服务器端口输入框 . 文本 为 选择列表 取 global 物联网设置 中索引值为 2 的列表项
设置 id输入框 . 文本 为 选择列表 取 global 物联网设置 中索引值为 3 的列表项
设置 pwd输入框 . 文本 为 选择列表 取 global 物联网设置 中索引值为 4 的列表项
设置 Topic输入框 . 文本 为 选择列表 取 global 物联网设置 中索引值为 5 的列表项

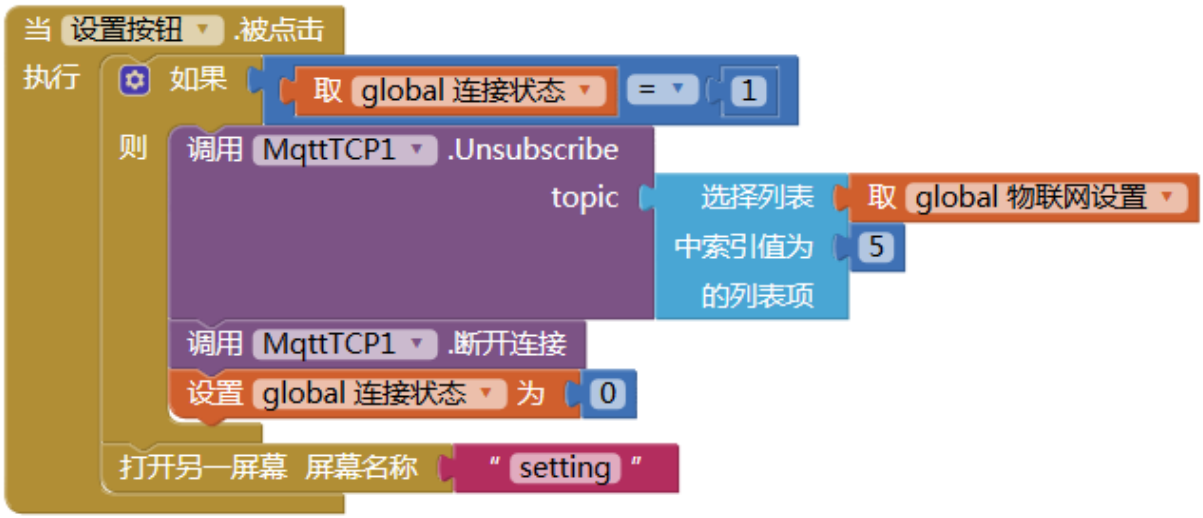

当 确认按钮 .被点击
执行
将列表 取 global 物联网设置
中索引值为 1
的列表项替换为 服务器地址输入框 . 文本
将列表 取 global 物联网设置
中索引值为 2
的列表项替换为 服务器端口输入框 . 文本
将列表 取 global 物联网设置
中索引值为 3
的列表项替换为 id输入框 . 文本
将列表 取 global 物联网设置
中索引值为 4
的列表项替换为 pwd输入框 . 文本
将列表 取 global 物联网设置
中索引值为 5
的列表项替换为 Topic输入框 . 文本
如果 是否为空 选择列表 取 global 物联网设置 中索引值为 1 的列表项
或 是否为空 选择列表 取 global 物联网设置 中索引值为 2 的列表项
或 是否为空 选择列表 取 global 物联网设置 中索引值为 3 的列表项
或 是否为空 选择列表 取 global 物联网设置 中索引值为 4 的列表项
或 是否为空 选择列表 取 global 物联网设置 中索引值为 5 的列表项
则 调用 对话框1 .显示警告信息 通知 " 设置项不能有空项 "
否则 调用 微数据库1 .保存数值 标签 " address " 存储值 选择列表 取 global 物联网设置 中索引值为 1 的列表项
调用 微数据库1 .保存数值 标签 " port " 存储值 选择列表 取 global 物联网设置 中索引值为 2 的列表项
调用 微数据库1 .保存数值 标签 " id " 存储值 选择列表 取 global 物联网设置 中索引值为 3 的列表项
调用 微数据库1 .保存数值 标签 " psw " 存储值 选择列表 取 global 物联网设置 中索引值为 4 的列表项
调用 微数据库1 .保存数值 标签 " topic " 存储值 选择列表 取 global 物联网设置 中索引值为 5 的列表项
关闭屏幕

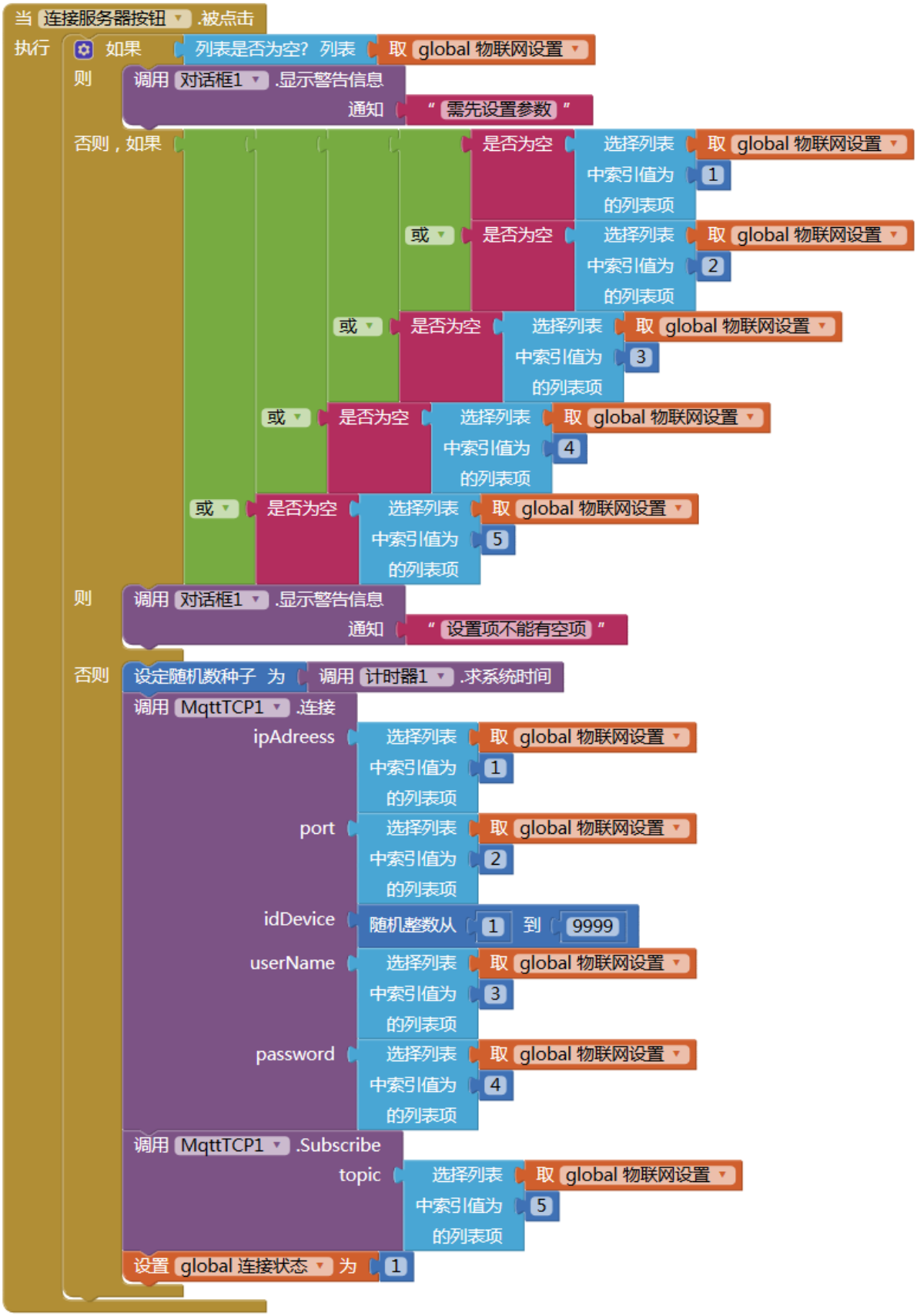
逻辑设计-screen1 屏幕逻辑设计图

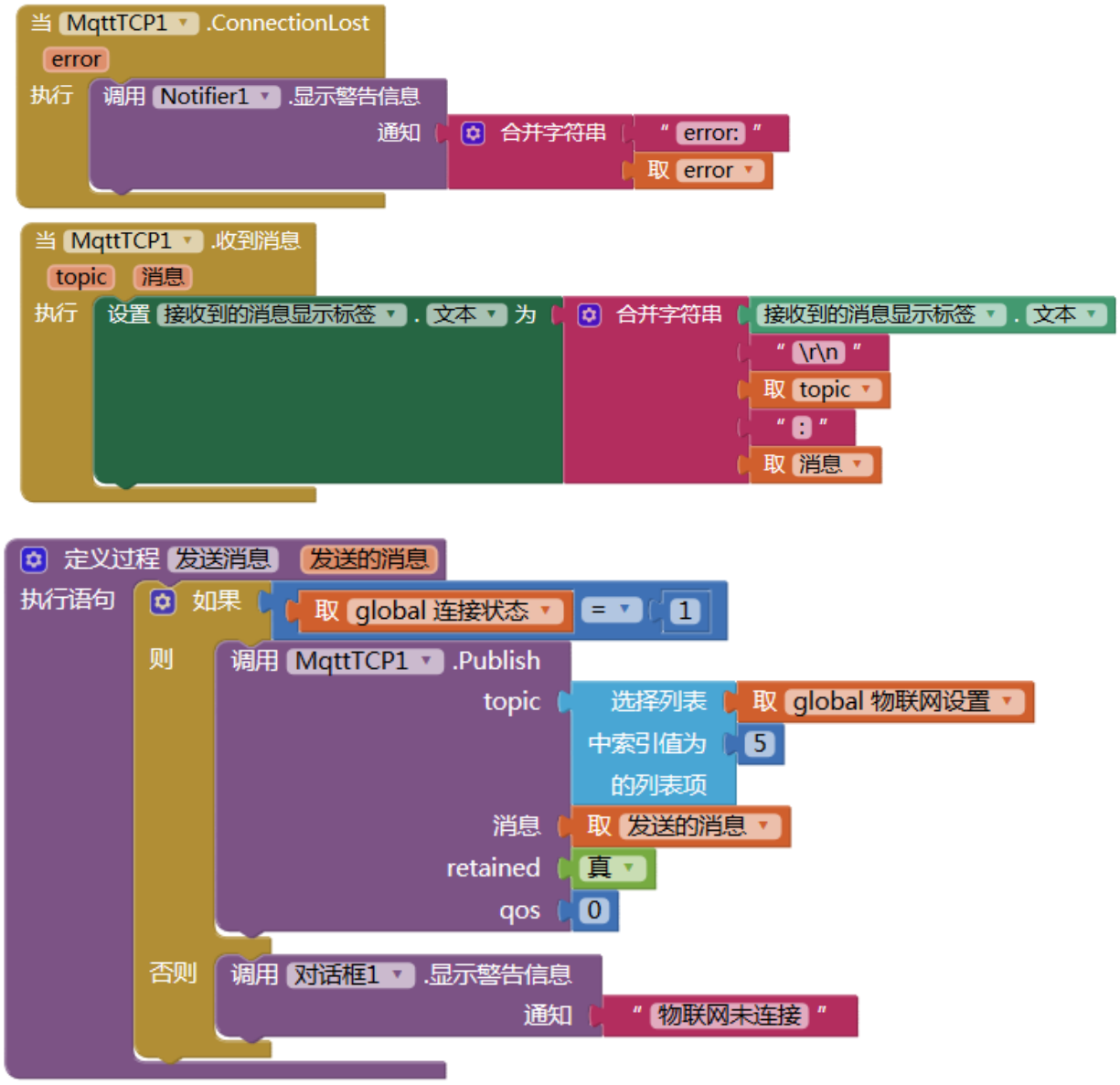

定义过程 获取及初始化数据
执行语句
将列表 取 global 物联网设置
中索引值为 1
的列表项替换为 调用 微数据库1 .获取数值
标签 " address "
无标签时返回值 " iot.dfrobot.com.cn "
将列表 取 global 物联网设置
中索引值为 2
的列表项替换为 调用 微数据库1 .获取数值
标签 " port "
无标签时返回值 1883
将列表 取 global 物联网设置
中索引值为 3
的列表项替换为 调用 微数据库1 .获取数值
标签 " id "
无标签时返回值 " "
将列表 取 global 物联网设置
中索引值为 4
的列表项替换为 调用 微数据库1 .获取数值
标签 " psw "
无标签时返回值 " "
将列表 取 global 物联网设置
中索引值为 5
的列表项替换为 调用 微数据库1 .获取数值
标签 " topic "
无标签时返回值 " "

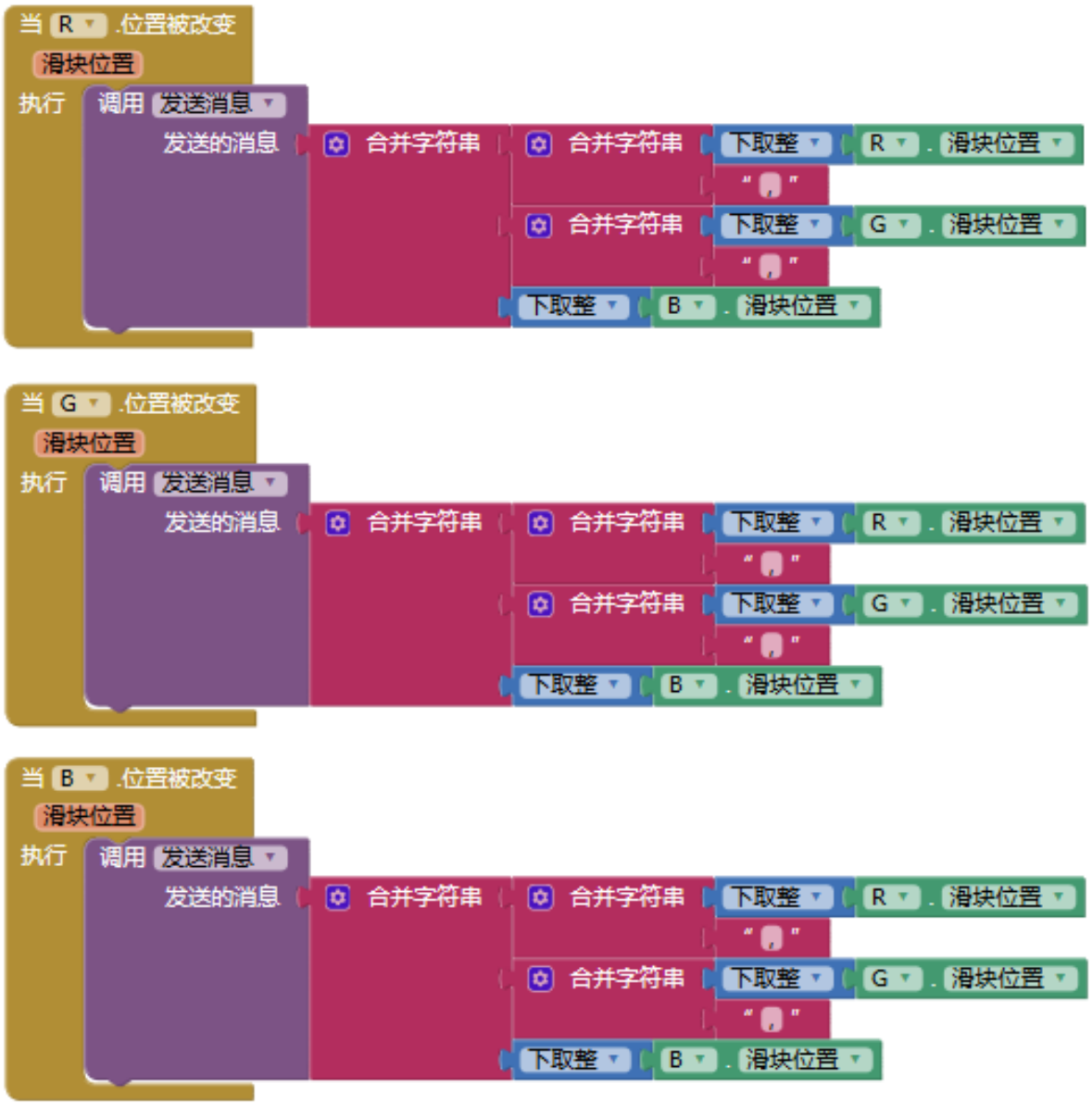

当 Screen1 .初始化
执行 调用 获取及初始化数据


当 Screen1 .关闭屏幕
其他屏幕名称 返回结果
执行 调用 获取及初始化数据


当 设置按钮 .被点击
执行 如果 取 global 连接状态 = 1
则 调用 MqttTCP1 .Unsubscribe
topic 选择列表 取 global 物联网设置
中索引值为 5
的列表项
调用 MqttTCP1 .断开连接
设置 global 连接状态 为 0
打开另一屏幕 屏幕名称 " setting "

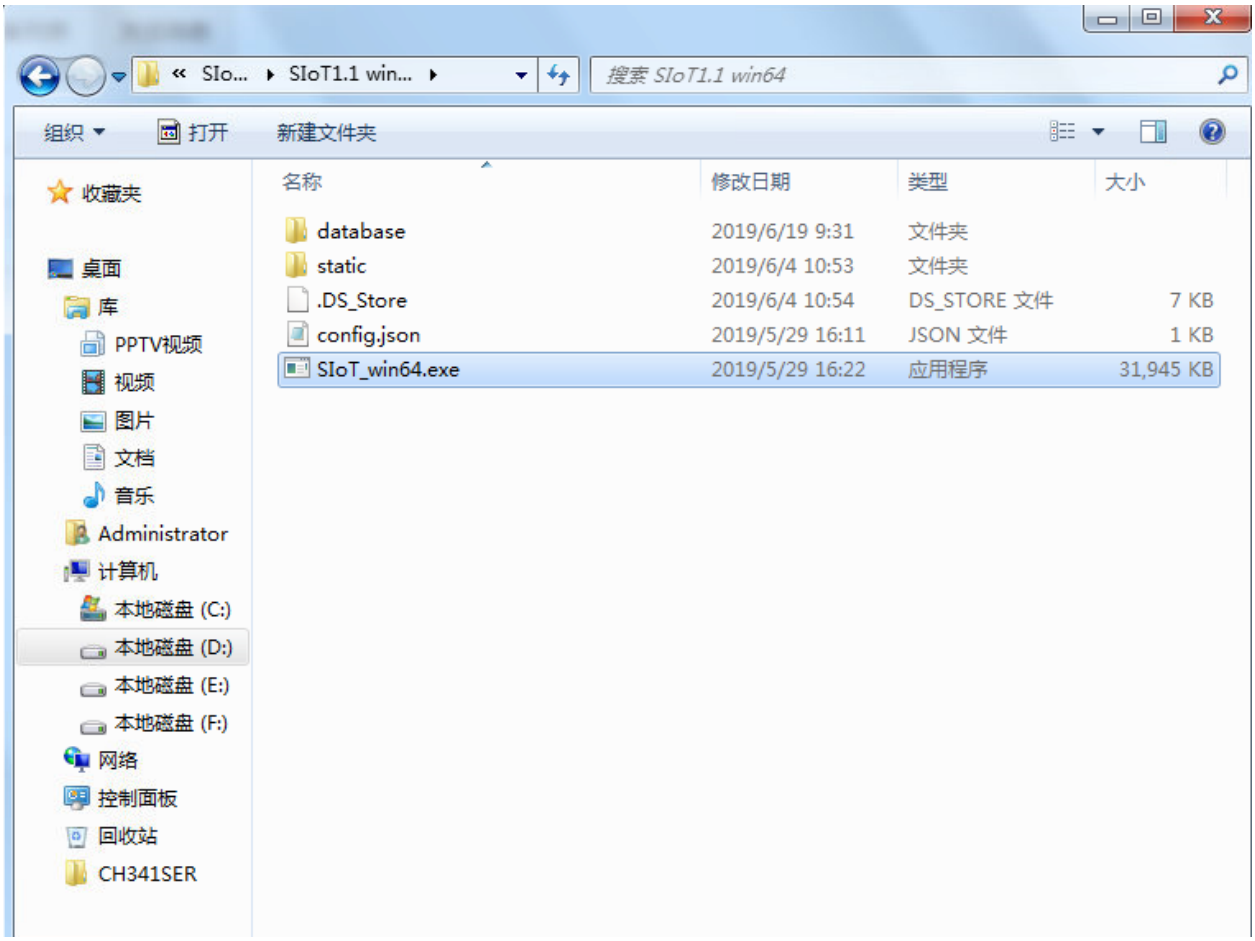

当 连接服务器按钮 .被点击
执行
如果 列表是否为空? 列表 取 global 物联网设置
则
调用 对话框1 .显示警告信息
通知 " 需先设置参数 "
否则，如果 是否为空 选择列表 取 global 物联网设置 中索引值为 1 的列表项
或 是否为空 选择列表 取 global 物联网设置 中索引值为 2 的列表项
或 是否为空 选择列表 取 global 物联网设置 中索引值为 3 的列表项
或 是否为空 选择列表 取 global 物联网设置 中索引值为 4 的列表项
或 是否为空 选择列表 取 global 物联网设置 中索引值为 5 的列表项
则
调用 对话框1 .显示警告信息
通知 " 设置项不能有空项 "
否则
设定随机数种子 为 调用 计时器1 .求系统时间
调用 MqttTCP1 .连接
ipAdreess 选择列表 取 global 物联网设置 中索引值为 1 的列表项
port 选择列表 取 global 物联网设置 中索引值为 2 的列表项
idDevice 随机整数从 1 到 9999
userName 选择列表 取 global 物联网设置 中索引值为 3 的列表项
password 选择列表 取 global 物联网设置 中索引值为 4 的列表项
调用 MqttTCP1 .Subscribe
topic 选择列表 取 global 物联网设置 中索引值为 5 的列表项
设置 global 连接状态 为 1

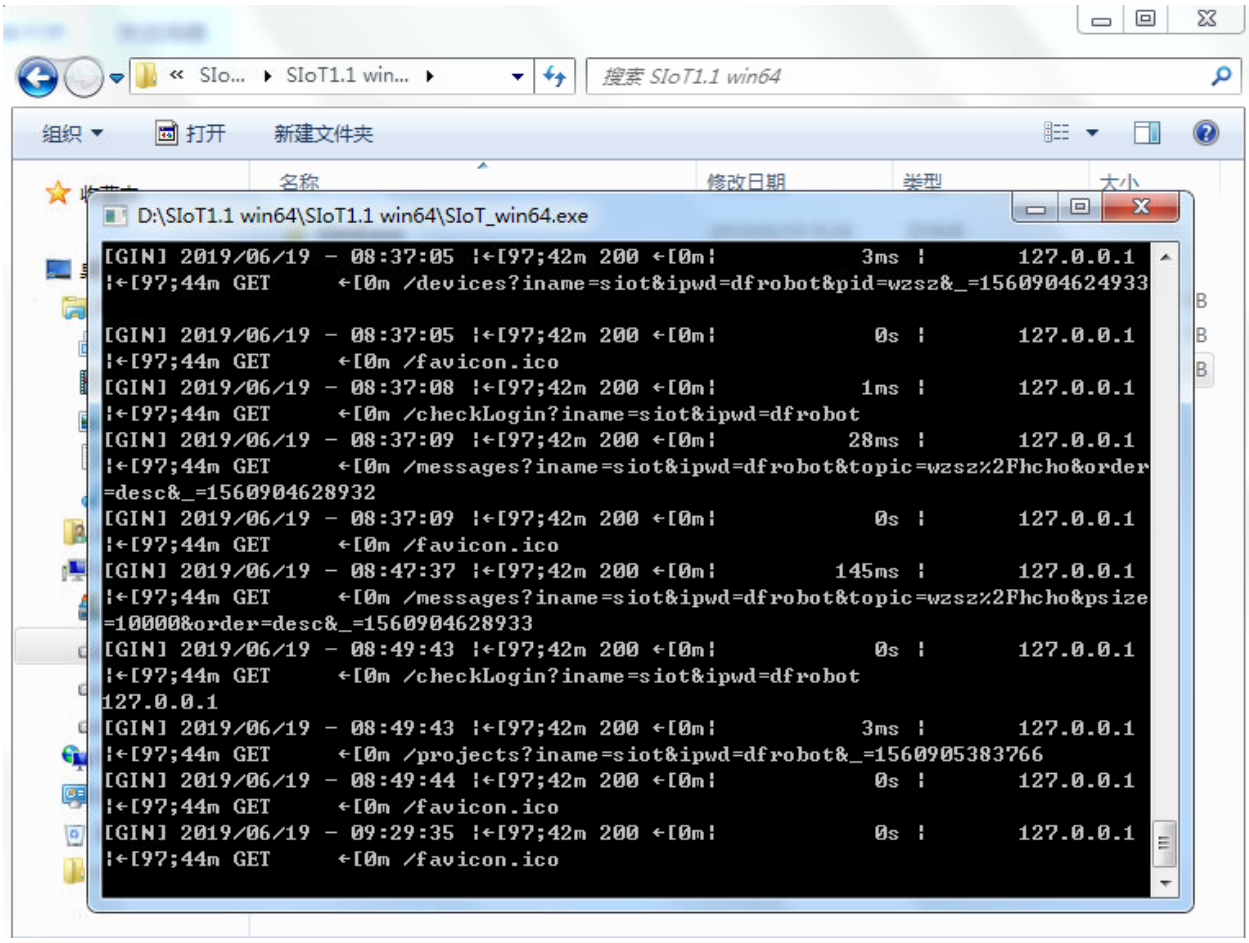

当 MqttTCP1 .ConnectionLost
error
执行 调用 Notifier1 .显示警告信息
通知 合并字符串 " error: "
取 error
当 MqttTCP1 .收到消息
topic 消息
执行 设置 接收到的消息显示标签 . 文本 为 合并字符串 接收到的消息显示标签 . 文本
" \r\n "
取 topic
" : "
取 消息
定义过程 发送消息 发送的消息
执行语句 如果 取 global 连接状态 = 1
则 调用 MqttTCP1 .Publish
topic 选择列表 取 global 物联网设置
中索引值为 5
的列表项
消息 取 发送的消息
retained 真
qos 0
否则 调用 对话框1 .显示警告信息
通知 " 物联网未连接 "



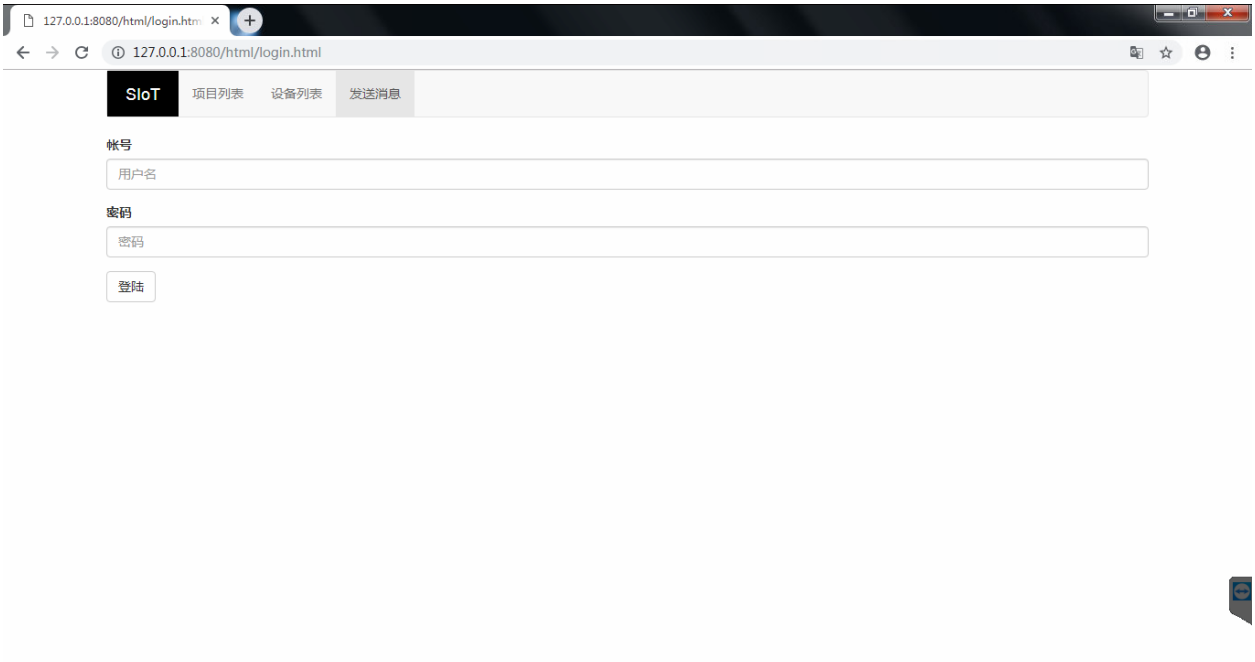
2. 本地物联网平台 SIoT 的运行并创建设备与主题：

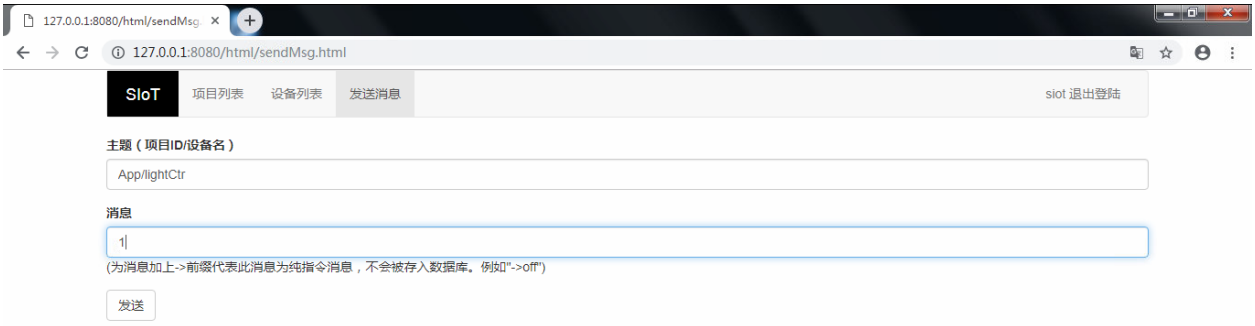
1）运行 SIoT 本地物联网平台：


« SIo... ▸ SIoT1.1 win... ▸
搜索 SIoT1.1 win64
组织
打开
新建文件夹
收藏夹
桌面
库
PPTV视频
视频
图片
文档
音乐
Administrator
计算机
本地磁盘 (C:)
本地磁盘 (D:)
本地磁盘 (E:)
本地磁盘 (F:)
网络
控制面板
回收站
CH341SER
名称
修改日期
类型
大小
database 2019/6/19 9:31 文件夹
static 2019/6/4 10:53 文件夹
.DS_Store 2019/6/4 10:54 DS_STORE 文件 7 KB
config.json 2019/5/29 16:11 JSON 文件 1 KB
SIoT_win64.exe 2019/5/29 16:22 应用程序 31,945 KB



2）登录 SIoT，并创建项目（App）与主题（lightCtr）：

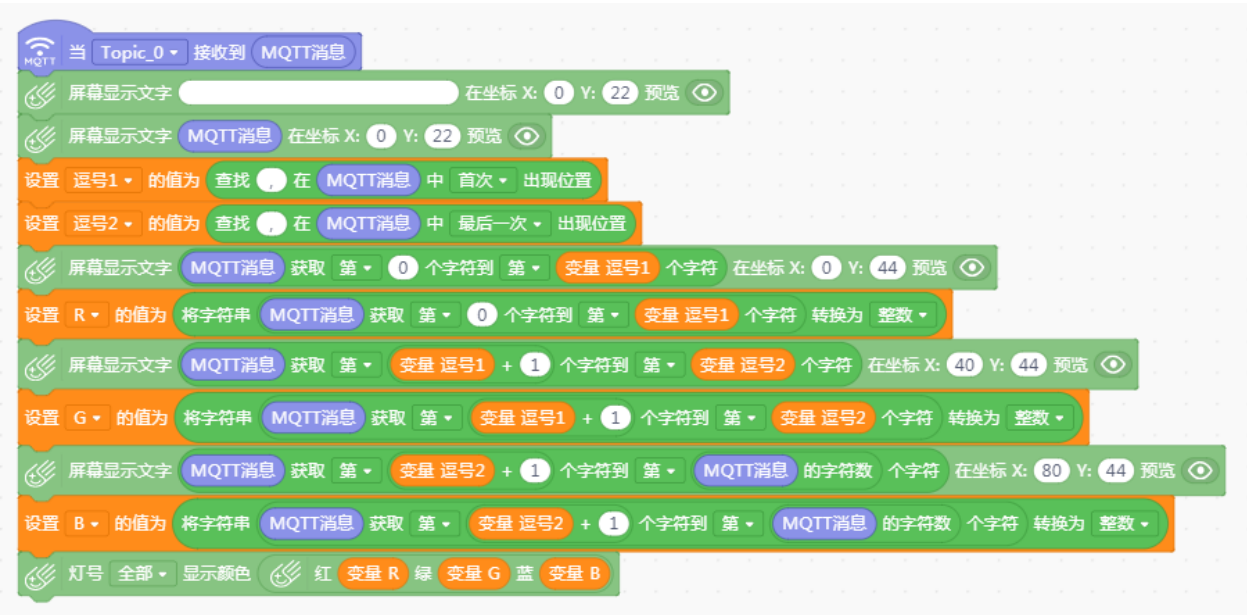




3. 掌控板端编程 (工具平台：mind+1.5.5)：


ESP32 主程序
屏幕显示为 全黑
屏幕显示图片 在坐标 x: 39 y: 7
WIFI 连接到 热点: miao 密码: 123456654
等待直到 WIFI 连接成功？
屏幕显示为 全黑
屏幕显示文字 WIFI已连接 在坐标 X: 0 Y: 0 预览
MQTT 初始化参数
MQTT 发起连接
等待直到 MQTT 连接成功？
屏幕显示为 全黑
屏幕显示文字 连接成功 在坐标 X: 0 Y: 0 预览


ESP32 主程序
屏幕显示为 全黑
屏幕显示图片 在坐标 x: 39 y: 7
WIFI 连接到 热点: miao 密码: 123456654
等待直到 WIFI 连接成功?
屏幕显示为 全黑
屏幕显示文字 WIFI已连接 在坐标 X: 0 Y: 0 预览
MQTT 初始化参数
物联网平台: SIOT
参数:
服务器地址: 178.16.206.80
账号: siot
密码: dfrobot
Topic_0: App/lightCtr
0 预览


当 Topic_0 接收到 MQTT消息
屏幕显示文字 在坐标 X: 0 Y: 22 预览
屏幕显示文字 MQTT消息 在坐标 X: 0 Y: 22 预览
设置 逗号1 的值为 查找 , 在 MQTT消息 中 首次 出现位置
设置 逗号2 的值为 查找 , 在 MQTT消息 中 最后一次 出现位置
屏幕显示文字 MQTT消息 获取 第 0 个字符到 第 变量 逗号1 个字符 在坐标 X: 0 Y: 44 预览
设置 R 的值为 将字符串 MQTT消息 获取 第 0 个字符到 第 变量 逗号1 个字符 转换为 整数
屏幕显示文字 MQTT消息 获取 第 变量 逗号1 + 1 个字符到 第 变量 逗号2 个字符 在坐标 X: 40 Y: 44 预览
设置 G 的值为 将字符串 MQTT消息 获取 第 变量 逗号1 + 1 个字符到 第 变量 逗号2 个字符 转换为 整数
屏幕显示文字 MQTT消息 获取 第 变量 逗号2 + 1 个字符到 第 MQTT消息 的字符数 个字符 在坐标 X: 80 Y: 44 预览
设置 B 的值为 将字符串 MQTT消息 获取 第 变量 逗号2 + 1 个字符到 第 MQTT消息 的字符数 个字符 转换为 整数
灯号 全部 显示颜色 红 变量 R 绿 变量 G 蓝 变量 B

4. 案例下载地址：https://github.com/vvlink/SIoT/tree/master/examples/Appinventor/%E7%89%A9%E8%81%94%E7%BD%91%E5%85%89%E7%BA%BF%E6%8E%A7%E5%88%B6

3.9 Python

Python 是一种计算机程序设计语言。是一种面向对象的动态类型语言，最初被设计用于编写自动化脚本(shell)，随着版本的不断更新和语言新功能的添加，越来越多被用于独立的、大型项目的开发。因为 Python 开源，很多人为 Python 开发了各种模块或者库，老而弥坚，越来越受到关注，被誉为人工智能编程方面的第一选择。

3.9.1 Python 的 MQTT 库

用 Python 连接 MQTT，有多个库可以选择，如 paho-mqtt 和 siot。

1.paho-mqtt 库

paho-mqtt 是一个 MQTT 官方团队开发的 python client 库，支持 mqtt 3.1/ 3.1.1 协议。

官方网站地址：http://mqtt.org/tag/paho

安装命令：

```
pip install siot
```

2.siot 库

siot 是虚谷物联团队写的一个 Python 库。为了让初学者能够写出更加简洁、优雅的 Python 代码，将 paho-mqtt 库进行了进一步封装。

官方网站地址：https://github.com/vvlink/siot

安装命令。

```
pip install siot
```

3.9.2 siot 的代码范例

1. “发送消息”参考代码

代码功能

连接服务器，发送消息。

```
import siot
import time

SERVER = "127.0.0.1"            #MQTT 服务器 IP
```

(下页继续)

(续上页)

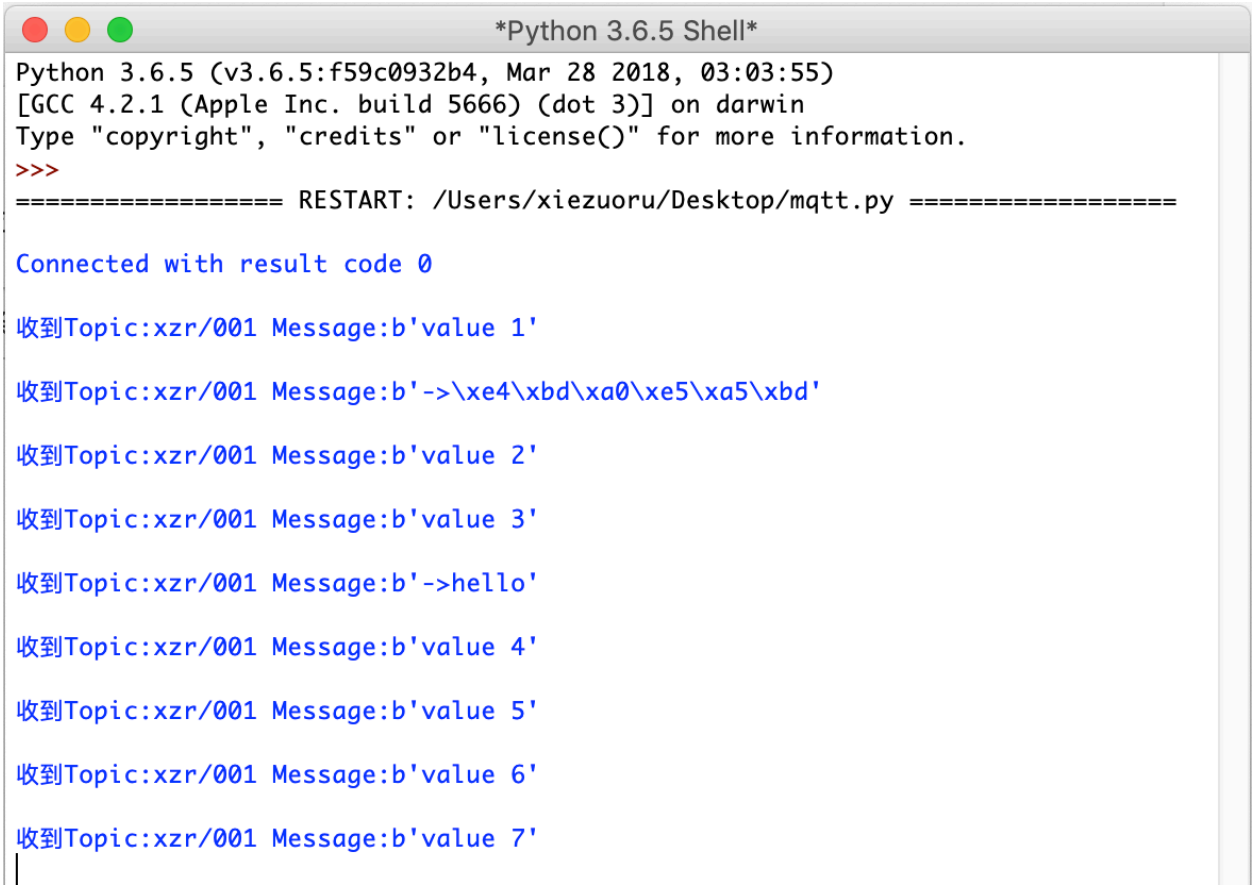
```
CLIENT_ID = ""                  # 在 SIoT 上，CLIENT_ID 可以留空
IOT_pubTopic  = 'xzr/001'       #“topic”为“项目名称/设备名称”
IOT_UserName ='siot'            # 用户名
IOT_PassWord ='dfrobot'         # 密码

siot.init(CLIENT_ID, SERVER, user=IOT_UserName, password=IOT_PassWord)
siot.connect()

tick = 0
while True:
  siot.publish(IOT_pubTopic, "value %d"%tick)
  time.sleep(1)           # 隔 1 秒发送一次
  tick = tick+1
```

2.“订阅消息”参考代码

连接服务器，订阅消息

```
import siot

SERVER = "127.0.0.1"        #MQTT 服务器 IP
CLIENT_ID = ""              # 在 SIoT 上，CLIENT_ID 可以留空
IOT_pubTopic  = 'xzr/001'   #“topic”为“项目名称/设备名称”
IOT_UserName ='siot'        # 用户名
IOT_PassWord ='dfrobot'     # 密码

def sub_cb(client, userdata, msg):# 定义收到消息时的提示信息
  print("\nTopic:" + str(msg.topic) + " Message:" + str(msg.payload))

siot.init(CLIENT_ID, SERVER, user=IOT_UserName, password=IOT_PassWord)
siot.connect()
siot.subscribe(IOT_pubTopic, sub_cb)
siot.loop()
```

测试效果



3.9.3 订阅消息动态绘图

1. 参考代码（动态绘制图表）

代码功能

连接服务器，根据订阅的消息，动态绘制出图表。

需要安装 matplotlib

参考命令：python3.6 -m pip install matplotlib

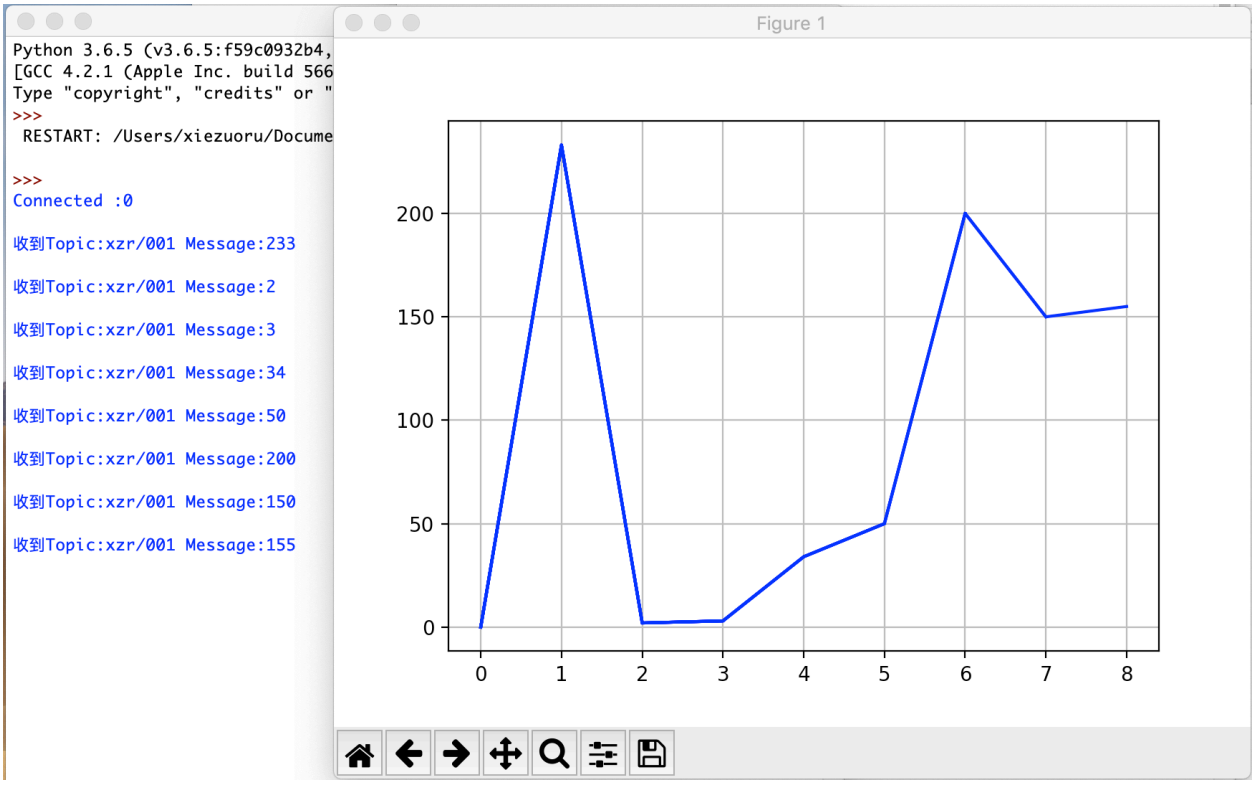
```
from pylab import *
import time,random
import siot

SERVER = "127.0.0.1"          #MQTT 服务器 IP
CLIENT_ID = ""                # 在 SIoT 上，CLIENT_ID 可以留空
IOT_pubTopic  = 'xzr/001'     # “topic” 为 “项目名称/设备名称”
IOT_UserName ='siot'          # 用户名
IOT_PassWord ='dfrobot'       # 密码
```

(下页继续)

(续上页)

```
def sub_cb(client, userdata, msg):# 定义收到消息时的提示信息
      print("\nTopic:" + str(msg.topic) + " Message:" + str(msg.payload))
      showplt(int(msg.payload)) # 开始绘图

siot.init(CLIENT_ID, SERVER, user=IOT_UserName, password=IOT_PassWord)
siot.connect()
siot.subscribe(IOT_pubTopic, sub_cb)
siot.loop()

def showplt(val):
    global x,y,i
    grid(True)
    plt.ion()
    x.append(i)
    i +=1
    y.append(val)
    ax.plot(x,y,'b')
    plt.pause(0.0001)
    #mac 系统请删除下方的 plt.ioff() 语句
    plt.ioff()
    plt.show()

if __name__ == '__main__':
      global x,y,i,fig, ax
      try:
        while True:
              fig, ax= plt.subplots()
              i=0
              x=[]
              y=[]
              showplt(0)
        except:
              siot.stop()
              print("disconnect seccused")
```

测试效果



3.10 Processing

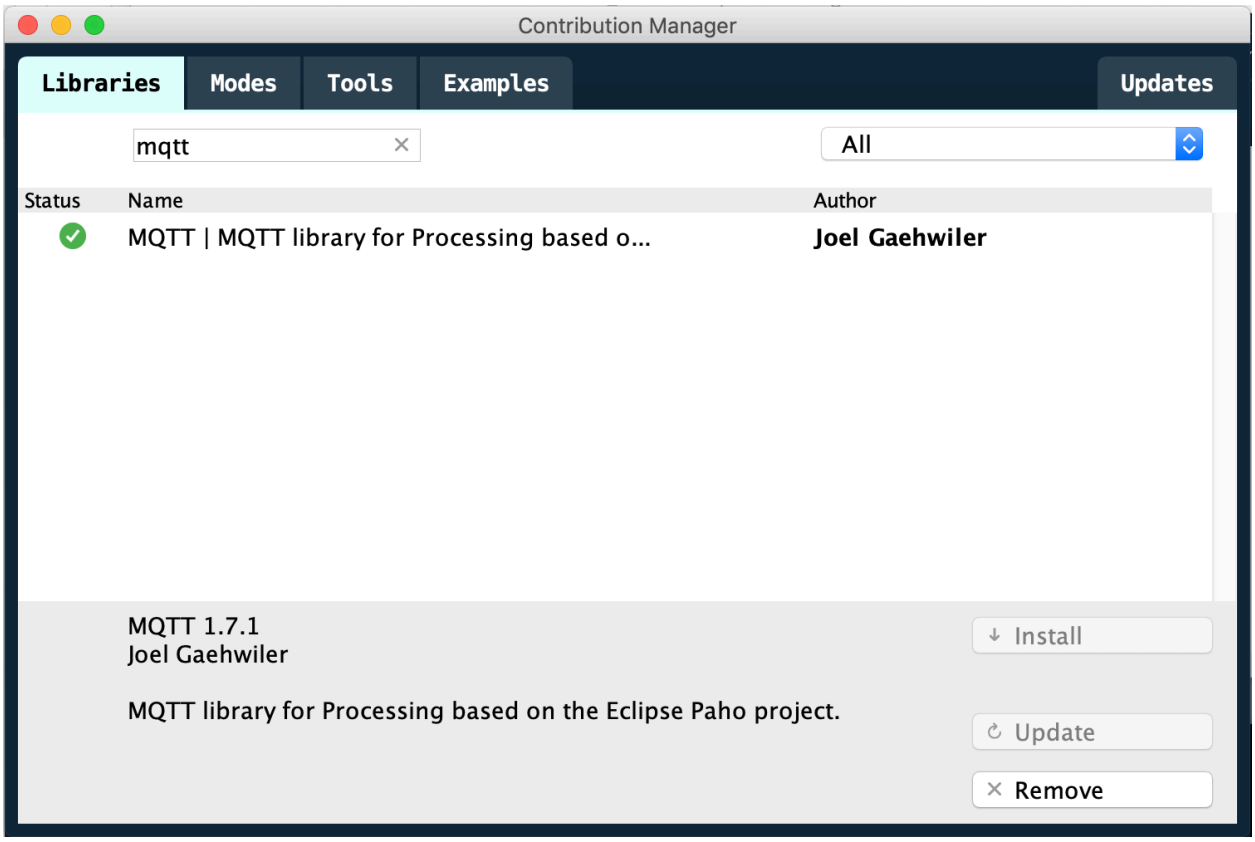
Processing 是一款专为设计师和艺术家使用的编程语言，由美国麻省理工学院媒体实验室（MIT）美学与运算小组创立的。Processing 的出现，被视为艺术设计创作的一场革命。利用 Processing，艺术家可以将抽象的数据呈现为生动的视觉形象。它不仅可以生成唯美的图形，还能编写出功能强大的互动艺术作品。

- Processing 下载地址：https://processing.org/

3.10.1 Processing 的 MQTT 库简介

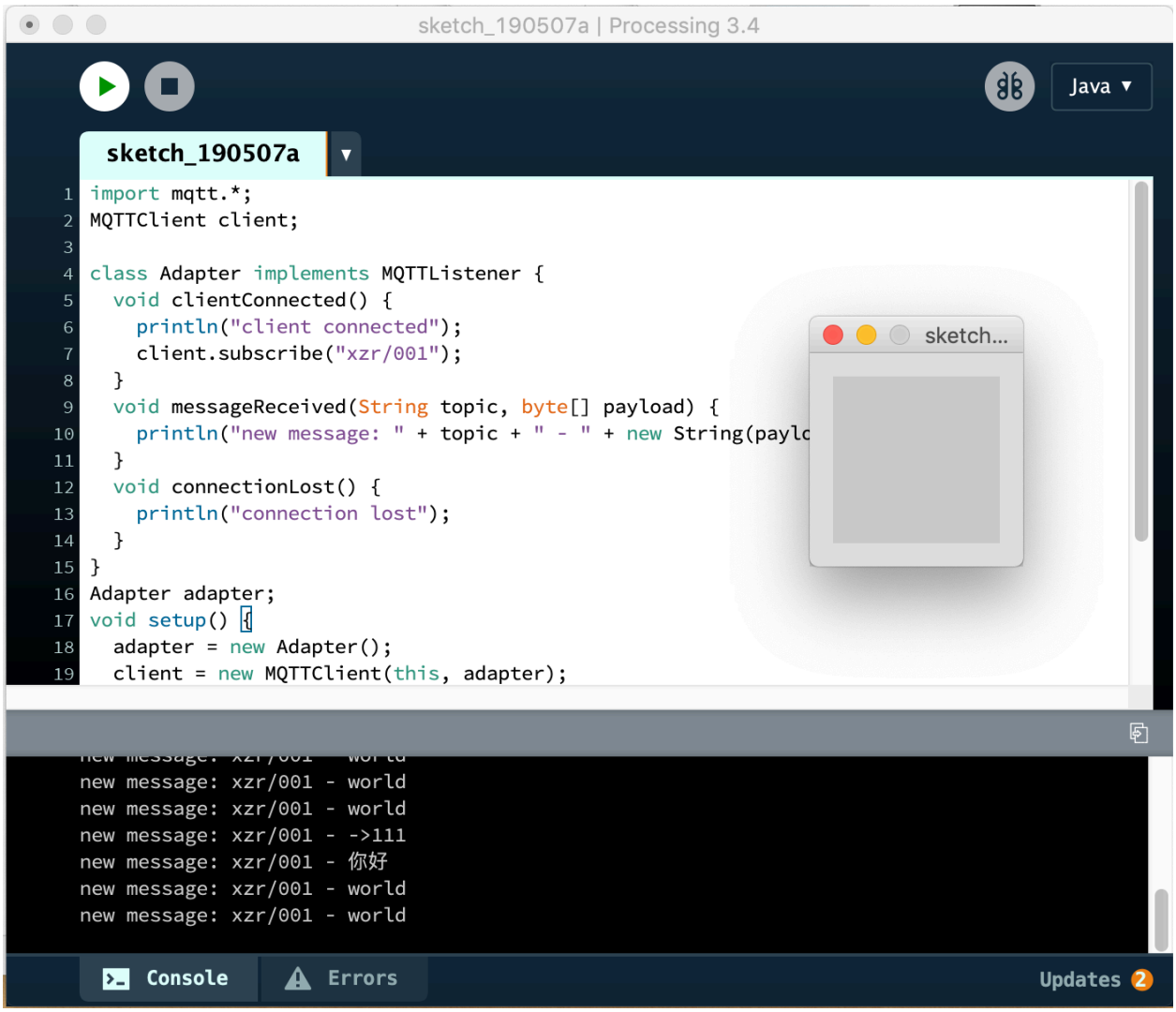
Processing 是一个开源的编程语言，有很多人为其开发了各种开源的库。借助 MQTT 库，Processing 即可与 MQTT 服务器进行交互。

在库文件中查找“MQTT”即可找到。



Processing 的 MQTT 库全名：MQTT library for Processing based on the Eclipse Paho project

库的开源地址：https://github.com/256dpi/processing-mqtt



3.10.2 参考代码

```
import mqtt.*;
MQTTClient client;

class Adapter implements MQTTListener {
  void clientConnected() {
    println("client connected");
    client.subscribe("xzr/001");//要订阅的消息名称
  }
  void messageReceived(String topic, byte[] payload) {
    println("new message: " + topic + " - " + new String(payload));
  }
```

(下页继续)

(续上页)

```
  void connectionLost() {
    println("connection lost");
  }
}
Adapter adapter;
void setup() {
  adapter = new Adapter();
  client = new MQTTClient(this, adapter);
  client.connect("mqtt://siot:dfrobot@127.0.0.1", "processing");//用户名为 siotd; 密码为
frobot
}
void draw() {}
void keyPressed() {
  client.publish("xzr/001", "world");//给名称为"xzr/001" 的 topic 发送消息"world"
}
```

3.11 其他软件

3.11.1 哪些软件可以连接 SIoT

只要能够访问网络，能做超级链接，就能和 SIoT 互动。比如，VB、PowerPoint、Word、Excel 等等。

3.11.2 SIoT 的 WebAPI 简介

VB（Visual Basic）程序中的 Microsoft.XMLHTTP 方法为 VB 软件与 SIoT 物联网平台的连接提供了支持。与此同时，powerpoint、word 和 excel 中的 VBA 程序与 VB 语言的语法和方法几乎一致。

Microsoft.XMLHTTP 方法简介 客户端调用 XMLHTTP 的过程很简单，只有 5 个步骤:

1、创建 XMLHTTP 对象

2、打开与服务端的连接，同时定义指令发送方式，服务网页 (URL) 和请求权限等。

客户端通过 Open 命令打开与服务端的服务网页的连接。与普通 HTTP 指令传送一样，可以用”GET”方法或”POST”方法指向服务端的服务网页。

3、发送指令。

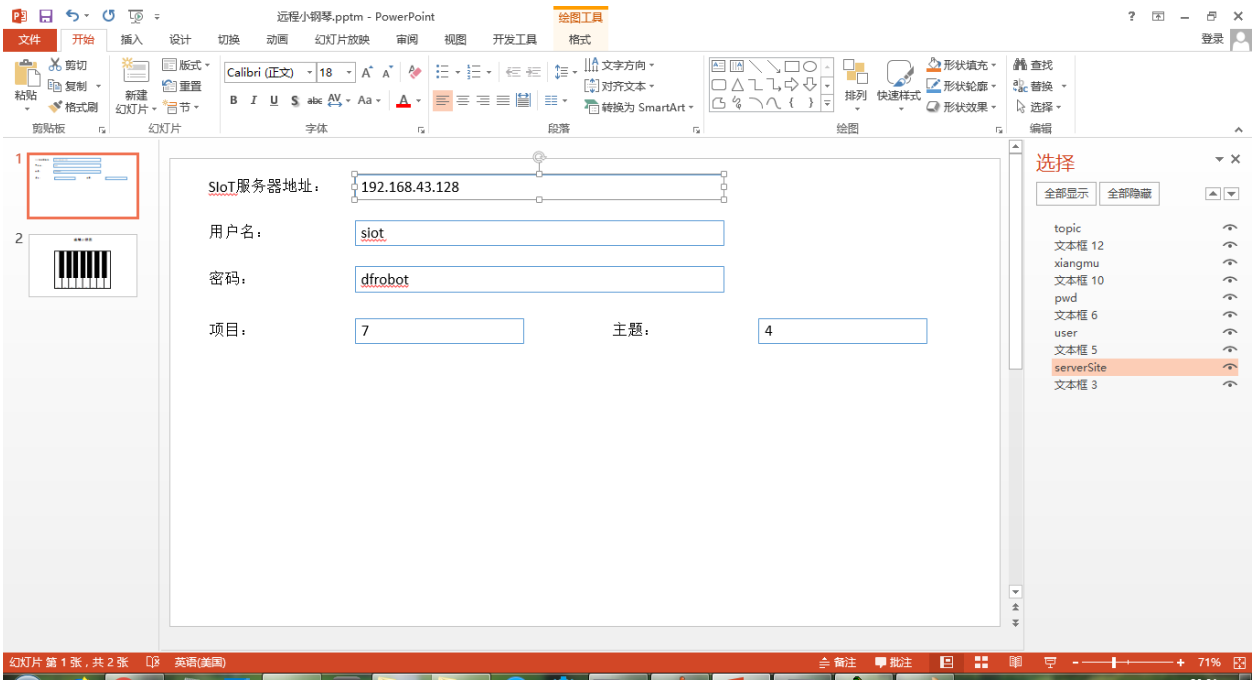
4、等待并接收服务端返回的处理结果。

5、释放 XMLHTTP 对象

3.11.3 具体例子（PPT）

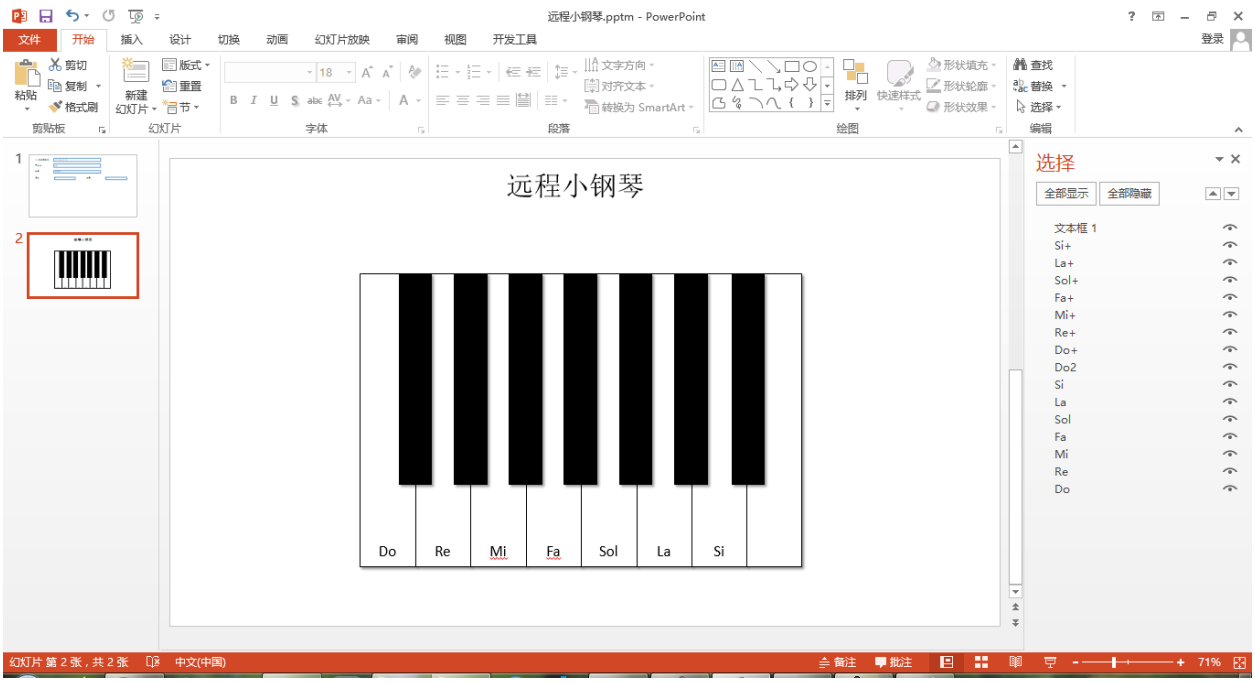
1、制作 ppt 页面

1）第一页：设置服务器连接信息



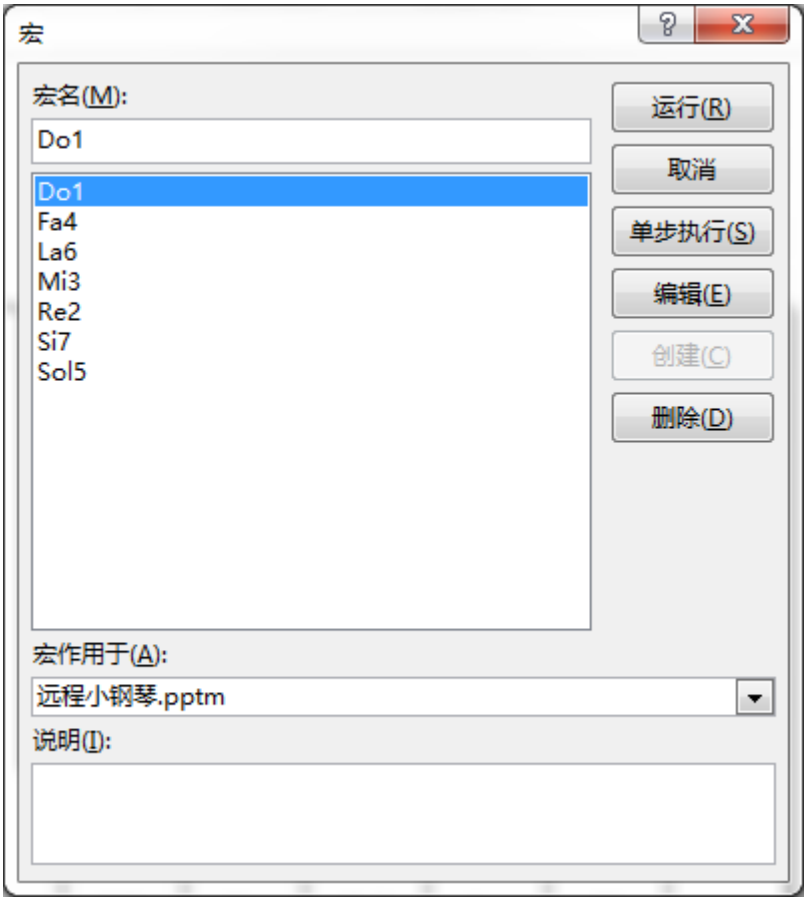
2）第二页：远程小钢琴界面

通过插入形状，制作远程小钢琴的界面



2、vba 编写程序

通过开发工具中的宏程序，创建 7 个音阶（do re mi fa sol la si）对应的程序。



```
Sub Do1()
    Dim rNum As String
    Randomize
    rNum = Int((9999) * Rnd(Now()) + 1)

    Dim strURL As String
    strURL = "http://" & ActivePresentation.Slides(1).Shapes(2).TextFrame.TextRange.Text
    strURL = strURL & ":8080/publish?topic=" & ActivePresentation.Slides(1).Shapes(8).TextFrame.TextRange.Text & "/" & ActivePresentation.Slides(1).Shapes(10).TextFrame.TextRange.Text
    strURL = strURL & "&msg=1&iname=" & ActivePresentation.Slides(1).Shapes(4).TextFrame.TextRange.Text & "&ipwd=" & ActivePresentation.Slides(1).Shapes(6).TextFrame.TextRange.Text & "&num=" & rNum


    Set xmlobject1 = CreateObject("Microsoft.XMLHTTP")

    xmlobject1.Open "GET", strURL, False
    xmlobject1.send

    '这条判断在此并无用处，可去掉
    If xmlobject1.ReadyState = 4 Then

    End If

    Set xmlobject1 = Nothing

End Sub
```

```
Sub Re2()
    Dim rNum As String
    Randomize
    rNum = Int((9999) * Rnd(Now()) + 1)

    Dim strURL As String
    strURL = "http://" & ActivePresentation.Slides(1).Shapes(2).TextFrame.TextRange.Text
    strURL = strURL & ":8080/publish?topic=" & ActivePresentation.Slides(1).Shapes(8).TextFrame.TextRange.Text & "/" & ActivePresentation.Slides(1).Shapes(10).TextFrame.TextRange.Text
    strURL = strURL & "&msg=2&iname=" & ActivePresentation.Slides(1).Shapes(4).TextFrame.TextRange.Text & "&ipwd=" & ActivePresentation.Slides(1).Shapes(6).TextFrame.TextRange.Text & "&num=" & rNum

    Set xmlobject2 = CreateObject("Microsoft.XMLHTTP")

    xmlobject2.Open "GET", strURL, False
    xmlobject2.send

    '这条判断在此并无用处，可去掉
    If xmlobject2.ReadyState = 4 Then

    End If

    Set xmlobject2 = Nothing

End Sub
```

```
Sub Mi3()
    Dim rNum As String
    Randomize
    rNum = Int((9999) * Rnd(Now()) + 1)

    Dim strURL As String
    strURL = "http://" & ActivePresentation.Slides(1).Shapes(2).TextFrame.TextRange.Text
    strURL = strURL & ":8080/publish?topic=" & ActivePresentation.Slides(1).Shapes(8).TextFrame.TextRange.Text & "/" & ActivePresentation.Slides(1).Shapes(10).TextFrame.TextRange.Text
    strURL = strURL & "&msg=3&iname=" & ActivePresentation.Slides(1).Shapes(4).TextFrame.TextRange.Text & "&ipwd=" & ActivePresentation.Slides(1).Shapes(6).TextFrame.TextRange.Text & "&num=" & rNum

    Set xmlobject3 = CreateObject("Microsoft.XMLHTTP")

    xmlobject3.Open "GET", strURL, False
    xmlobject3.send

    '这条判断在此并无用处，可去掉
    If xmlobject3.ReadyState = 4 Then

    End If

    Set xmlobject3 = Nothing
End Sub
```

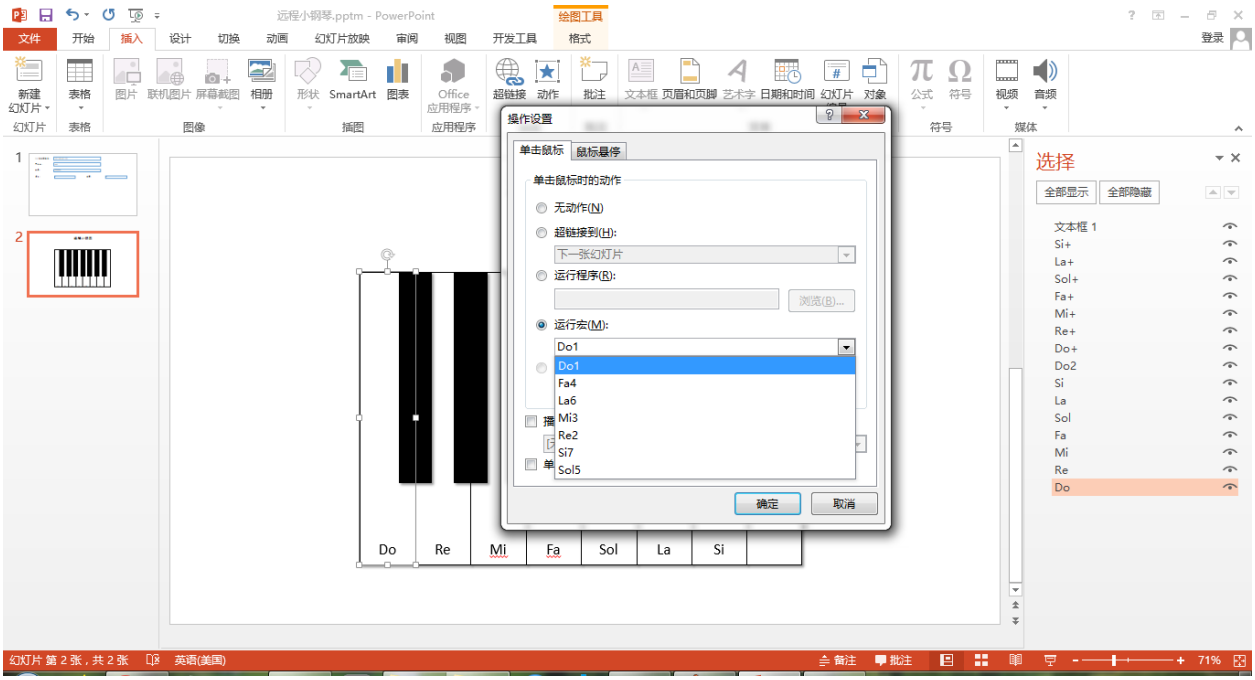
```
Sub Fa4()
    Dim rNum As String
    Randomize
    rNum = Int((9999) * Rnd(Now()) + 1)

    Dim strURL As String
    strURL = "http://" & ActivePresentation.Slides(1).Shapes(2).TextFrame.TextRange.Text
    strURL = strURL & ":8080/publish?topic=" & ActivePresentation.Slides(1).Shapes(8).TextFrame.TextRange.Text & "/" & ActivePresentation.Slides(1).Shapes(10).TextFrame.TextRange.Text
    strURL = strURL & "&msg=4&iname=" & ActivePresentation.Slides(1).Shapes(4).TextFrame.TextRange.Text & "&ipwd=" & ActivePresentation.Slides(1).Shapes(6).TextFrame.TextRange.Text & "&num=" & rNum

    Set xmlobject4 = CreateObject("Microsoft.XMLHTTP")

    xmlobject4.Open "GET", strURL, False
    xmlobject4.send

    '这条判断在此并无用处，可去掉
    If xmlobject4.ReadyState = 4 Then

    End If

    Set xmlobject4 = Nothing

End Sub
```

```
Sub Sol5()
    Dim rNum As String
    Randomize
    rNum = Int((9999) * Rnd(Now()) + 1)

    Dim strURL As String
    strURL = "http://" & ActivePresentation.Slides(1).Shapes(2).TextFrame.TextRange.Text
    strURL = strURL & ":8080/publish?topic=" & ActivePresentation.Slides(1).Shapes(8).TextFrame.TextRange.Text & "/" & ActivePresentation.Slides(1).Shapes(10).TextFrame.TextRange.Text
    strURL = strURL & "&msg=5&iname=" & ActivePresentation.Slides(1).Shapes(4).TextFrame.TextRange.Text & "&ipwd=" & ActivePresentation.Slides(1).Shapes(6).TextFrame.TextRange.Text & "&num=" & rNum

    Set xmlobject5 = CreateObject("Microsoft.XMLHTTP")

    xmlobject5.Open "GET", strURL, False
    xmlobject5.send

    '这条判断在此并无用处，可去掉
    If xmlobject5.ReadyState = 4 Then

    End If

    Set xmlobject5 = Nothing

End Sub
```

```
Sub La6()
    Dim rNum As String
    Randomize
    rNum = Int((9999) * Rnd(Now()) + 1)

    Dim strURL As String
    strURL = "http://" & ActivePresentation.Slides(1).Shapes(2).TextFrame.TextRange.Text
    strURL = strURL & ":8080/publish?topic=" & ActivePresentation.Slides(1).Shapes(8).TextFrame.TextRange.Text & "/" & ActivePresentation.Slides(1).Shapes(10).TextFrame.TextRange.Text
    strURL = strURL & "&msg=6&iname=" & ActivePresentation.Slides(1).Shapes(4).TextFrame.TextRange.Text & "&ipwd=" & ActivePresentation.Slides(1).Shapes(6).TextFrame.TextRange.Text & "&num=" & rNum

    Set xmlobject6 = CreateObject("Microsoft.XMLHTTP")

    xmlobject6.Open "GET", strURL, False
    xmlobject6.send

    '这条判断在此并无用处，可去掉
    If xmlobject6.ReadyState = 4 Then

    End If

    Set xmlobject6 = Nothing
End Sub

Sub Si7()
    Dim rNum As String
    Randomize
    rNum = Int((9999) * Rnd(Now()) + 1)

    Dim strURL As String
    strURL = "http://" & ActivePresentation.Slides(1).Shapes(2).TextFrame.TextRange.Text
    strURL = strURL & ":8080/publish?topic=" & ActivePresentation.Slides(1).Shapes(8).TextFrame.TextRange.Text & "/" & ActivePresentation.Slides(1).Shapes(10).TextFrame.TextRange.Text
    strURL = strURL & "&msg=7&iname=" & ActivePresentation.Slides(1).Shapes(4).TextFrame.TextRange.Text & "&ipwd=" & ActivePresentation.Slides(1).Shapes(6).TextFrame.TextRange.Text & "&num=" & rNum

    Set xmlobject7 = CreateObject("Microsoft.XMLHTTP")

    xmlobject7.Open "GET", strURL, False
    xmlobject7.send

    '这条判断在此并无用处，可去掉
    If xmlobject7.ReadyState = 4 Then

    End If

    Set xmlobject7 = Nothing
End Sub
```

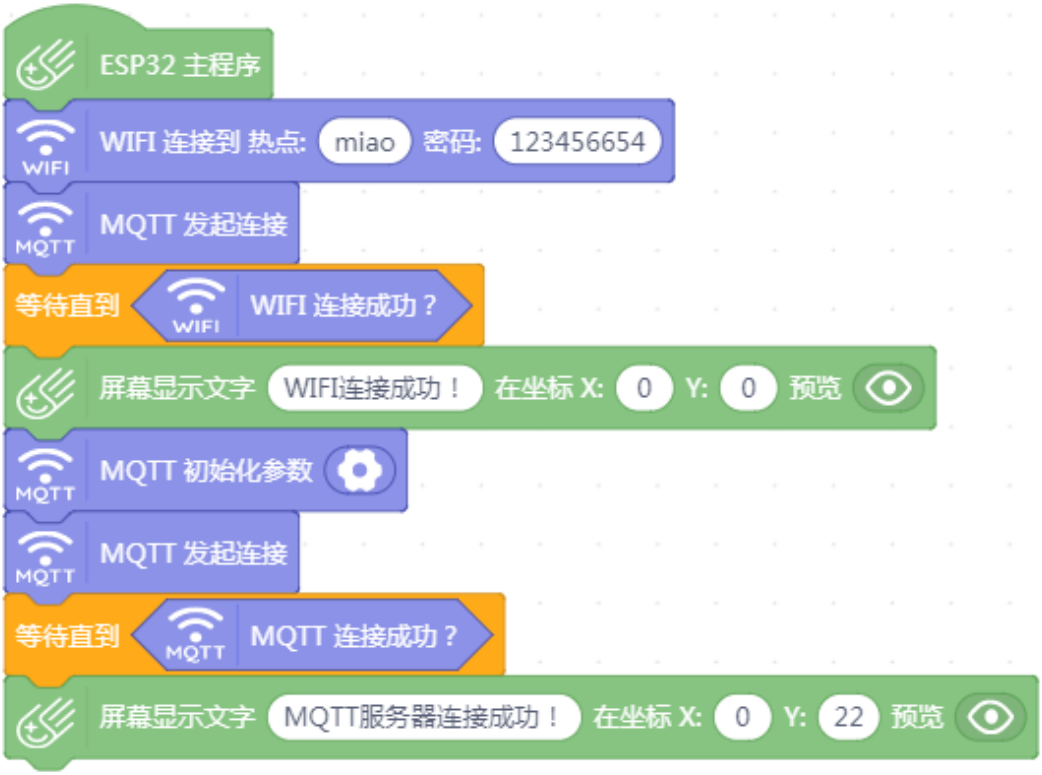
3、ppt 页面形状设置动作

将 ppt 第二页中 7 个钢琴键的形状分别插入动作，并选择对应的运行程序。



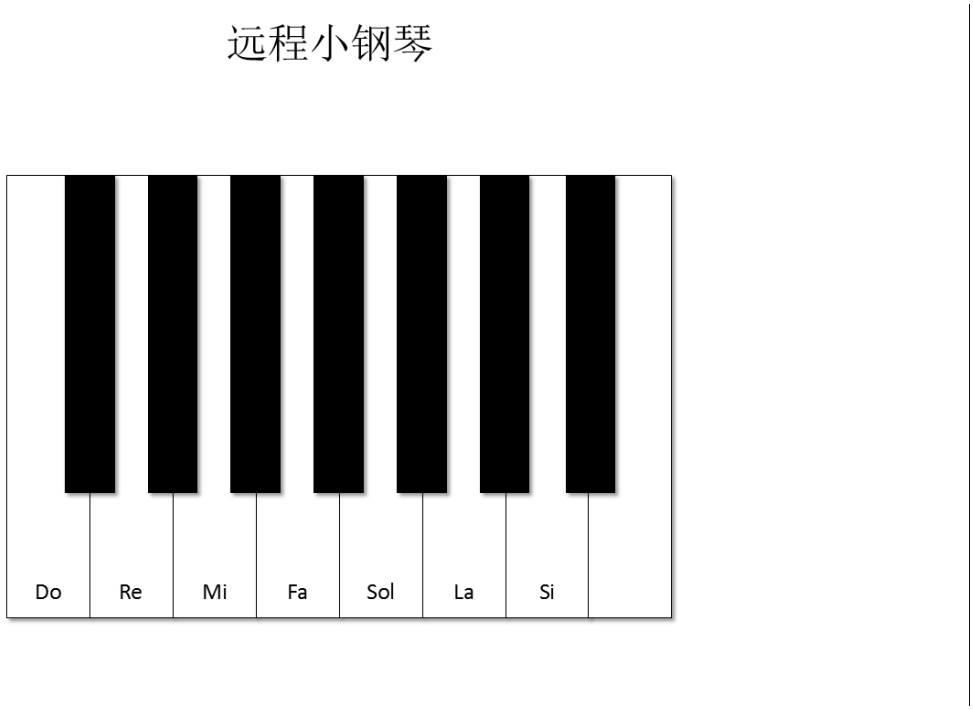
4、掌控板端编写音乐程序

设置掌控板连接 WIFI 和 SIoT 物联网平台



在掌控板端编写对应的音乐播放程序，当分别接收到 7 个音阶（do re mi fa sol la si）所对应的指令（1、2、3、4、5、6、7）时，播放相应的音乐。

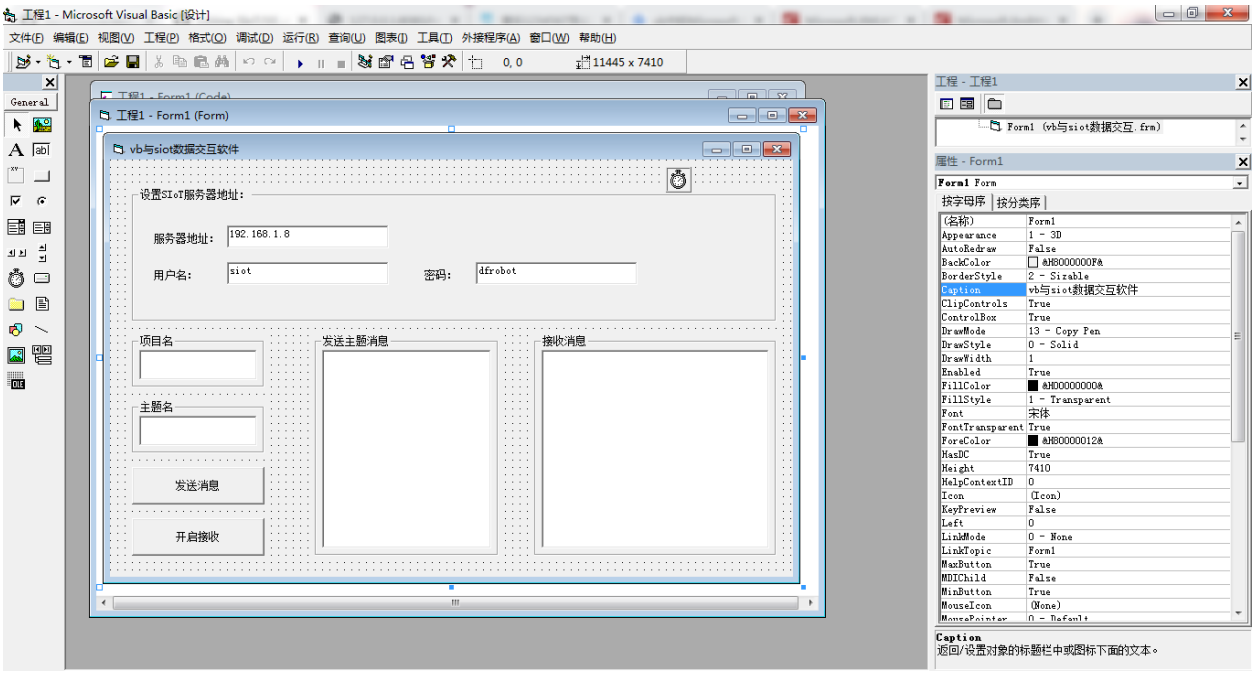
5、播放 ppt 远程弹奏小钢琴



6、案例下载地址：下载

3.11.4 具体例子（VB）

1、VB 程序界面设计



2、发送消息按钮编程

```
Private Sub Command1_Click()
    Dim topicstr As String, strURL As String


    topicstr = Text3.Text & "/" & Text4.Text

    Set xmlobject1 = CreateObject("Microsoft.XMLHTTP")

    strURL = "Http://" & ServerSiteText.Text & ":8080/publish?topic=" & topicstr & "&msg=" & Text1.Text & "&iname=" & userText.Text & "&ipwd=" & pwdText.Text


    xmlobject1.Open "GET", strURL, False
    xmlobject1.send
    If xmlobject1.ReadyState = 4 Then

    End If

    Set xmlobject1 = Nothing

    Text1.Text = ""


End Sub
```

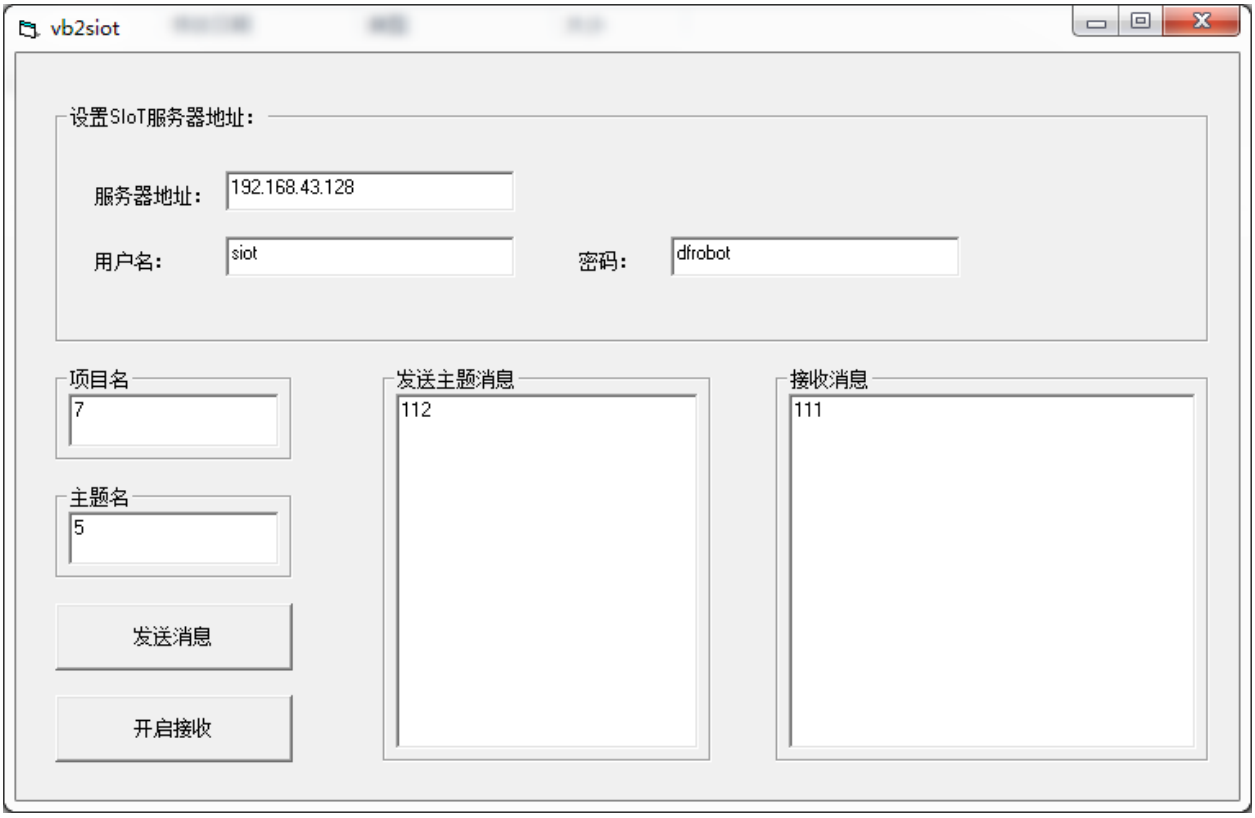
3、开启接收按钮编程

```
Private Sub Command2_Click()
Timer1.Enabled = True    '注意Timer控件的enable属性在初始化状态下为false
MsgBox ("服务器连接成功!")



End Sub
```

4、Timer 事件编程

```
Private Sub Timer1_Timer()
    Dim rNum As String
    Randomize
    rNum = Int((9999) * Rnd(Now()) + 1)

    Dim searchstring, searchchar1, searchchar2, charpos1, charpos2, char1, strURL2
    Dim topicstr As String


    topicstr = Text3.Text & "/" & Text4.Text

    searchchar1 = "Content"
    searchchar2 = "Created"

    Set xmlobject2 = CreateObject("Microsoft.XMLHTTP")


    strURL2 = "Http://" & ServerSiteText.Text & ":8080/lastmessage?topic=" & topicstr & "&iname=" & userText.Text & "&ipwd=" & pwdText.Text & "&num=" & rNum


    xmlobject2.Open "GET", strURL2, False

    xmlobject2.send

    If xmlobject2.ReadyState = 4 Then

       searchstring = xmlobject2.responseText
       charpos1 = InStr(1, searchstring, searchchar1, 1)
       charpos2 = InStr(1, searchstring, searchchar2, 1)
       char1 = Mid(searchstring, charpos1 + 10, charpos2 - 3 - charpos1 - 10)   '解析从SIoT接收到的消息

       Text2.Text = char1

    End If

    Set xmlobject2 = Nothing
End Sub
```

5、运行界面



6、案例下载

[下载](https://github.com/vvlink/SIoT/tree/master/examples/VB/%E6%8E%8C%E6%8E%A7%E6%9D%BF%E7%BB%93%E5%90%88SIoT%E6%A8%A1%E6%8B%9F%E9%94%AE%E9%BC%A0)

CHAPTER 4

典型应用案例

这一部分主要介绍各种利用物联网技术实现的典型应用案例，重点关注如何利用物联网技术进行科学探究。核心工具以虚谷号和掌控板为主。

4.1 科学探究之一天的温度分布

虚谷物联项目的核心工作就是利用物联网技术采集数据。借助 SIoT 物联网平台，学生不用注册和设置，一键启动，随时使用。通过对实时收集的数据进行合理分析，学生可以得出科学的分析结果，养成“用数据说话”的意识和习惯。

** 案例提供：天津师范大学

4.1.1 案例描述

我们在高中地理课中学过“一天之中，气温最高值出现在午后 14 时，气温最低值出现在日出前后”，基于此我们通过收集数据和数据分析进一步探索一天之中气温最高值是否出现在午后 14 时、气温最低值是否出现在日出前后、昼夜温差如何。所以本次活动的研究目的、研究内容如下：

研究目的：探索一天之中，气温最高值是否出现在午后 14 时、气温最低值是否出现在日出前后。

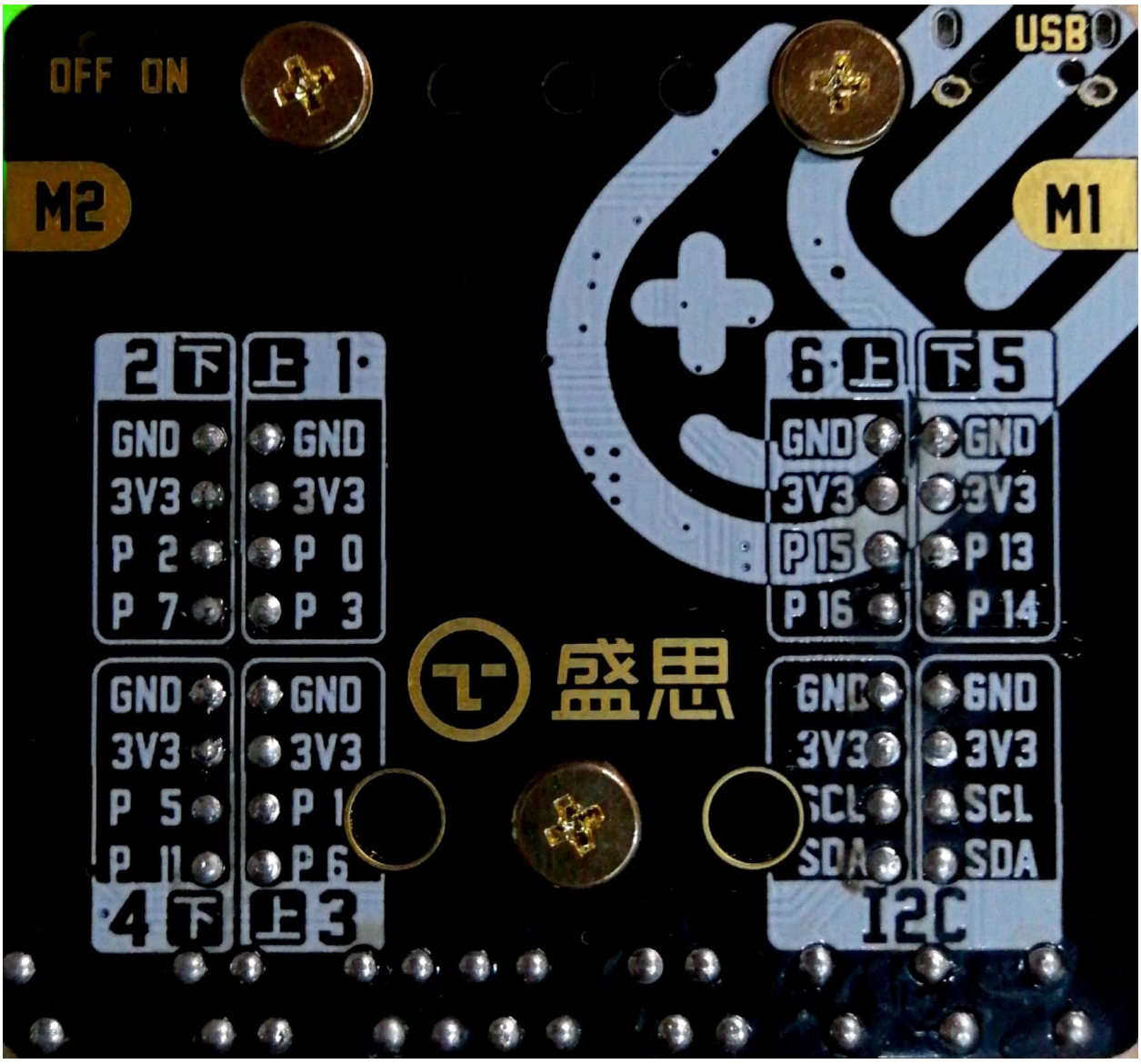
研究内容：以测量天津师范大学一天的室外温度为例，利用掌控板和温度传感器定时测量温度。测量地点为天津师范大学立教楼外，测量时间为 2019 年 5 月 19 日 00：00—24:00，收集数据时间间隔为 10s，数据收集完成后从 SIoT 物联网平台下载数据进行筛选和分析。

4.1.2 准备工作

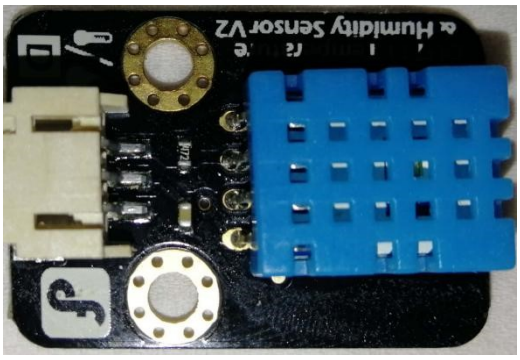
（一）硬件准备

掌控板及其连接线

扩展板及其连接线



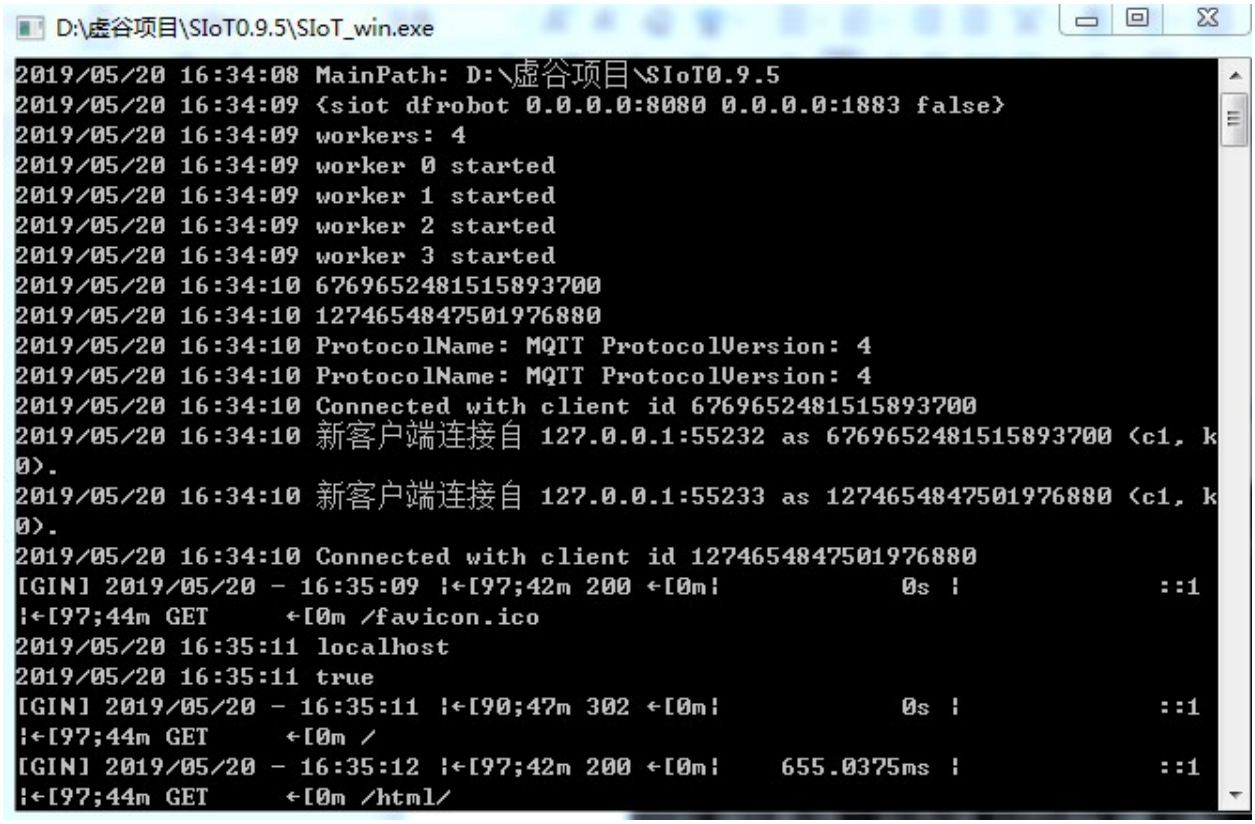
温湿度传感器及其连接线



（二）软件准备

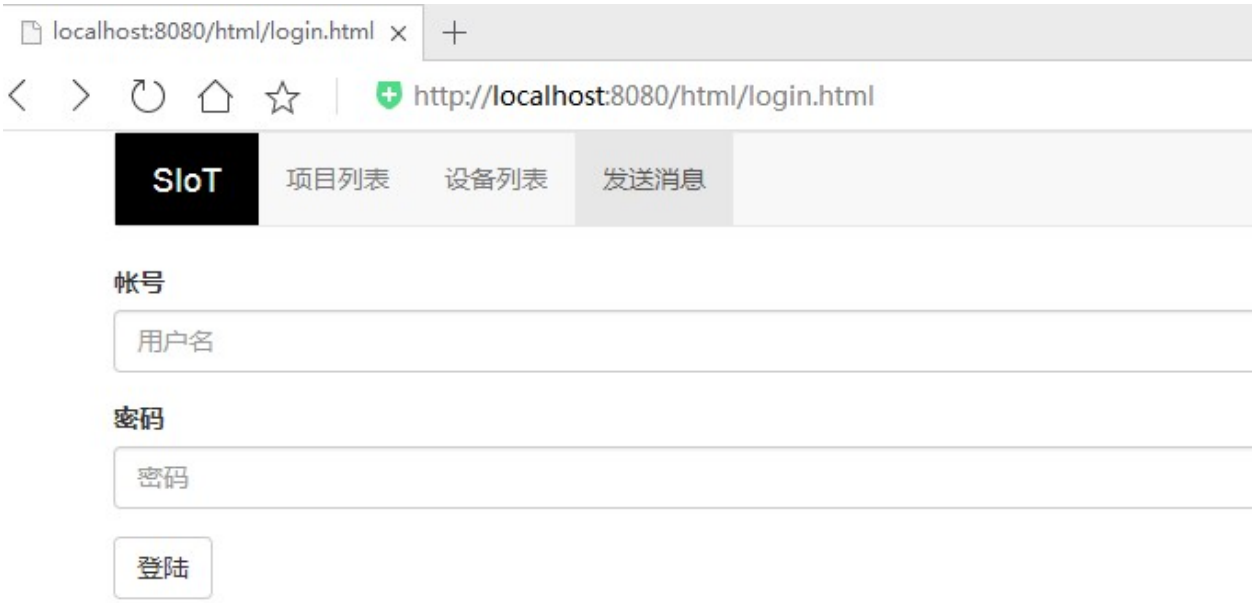
1. 搭建 SIoT 服务器

直接双击点击与系统匹配的 SIoT 运行文件，屏幕会弹出一个黑色的 CMD 窗口，在配置中请不要关闭它。

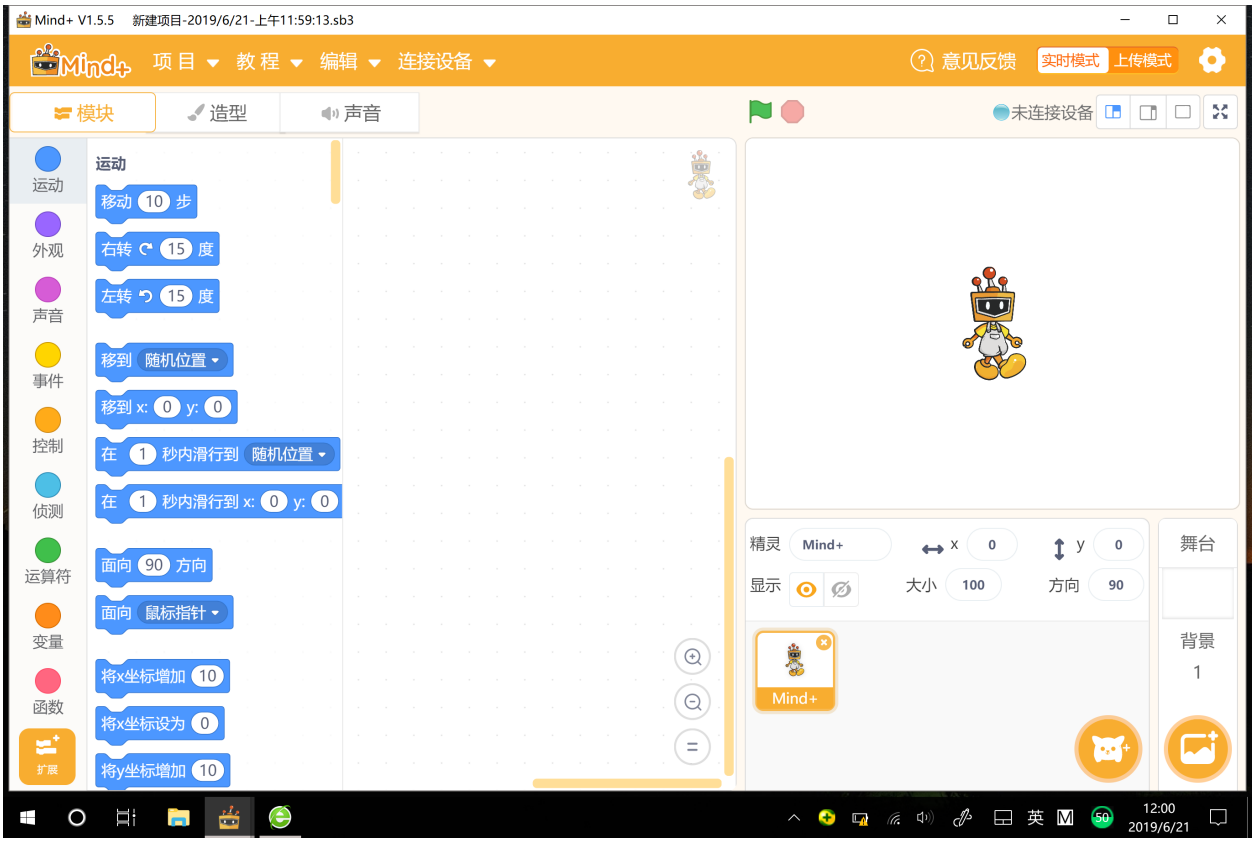


2. 登录 SIoT 平台

打开浏览器，输入 url：http://localhost:8080。

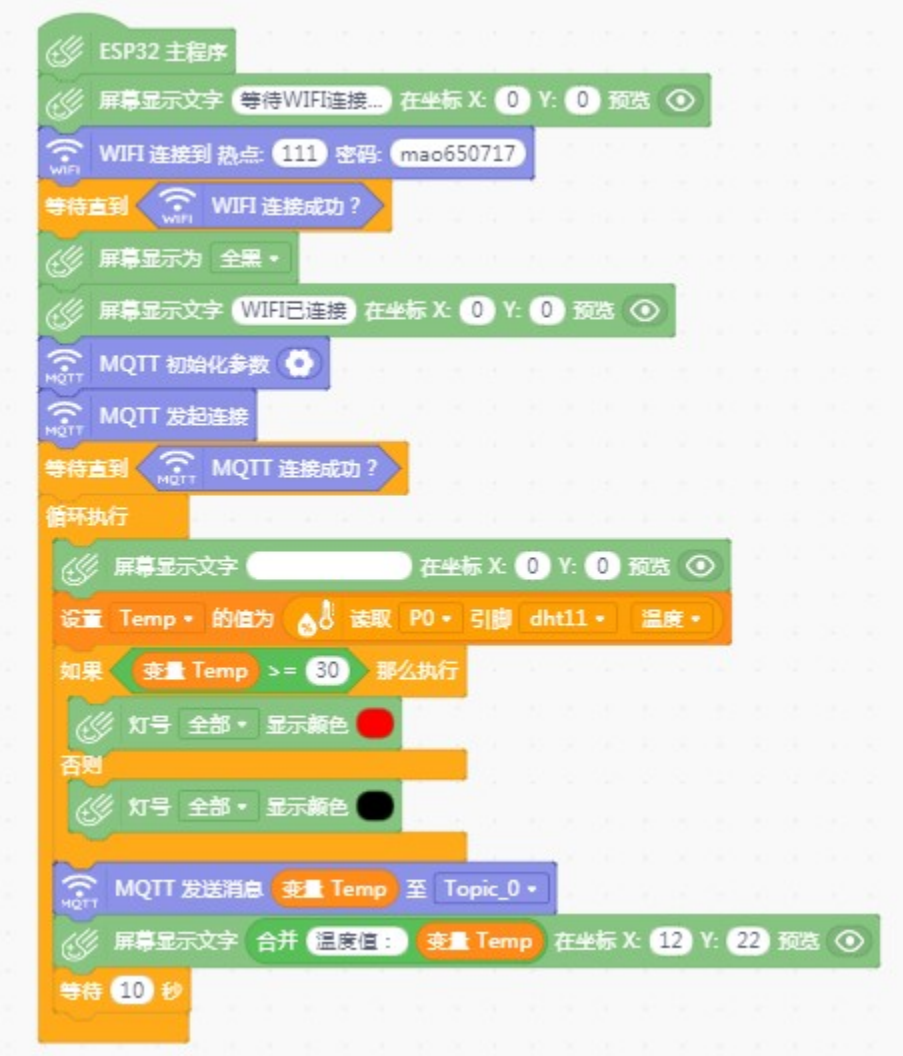


3. 打开 Mind+ V1.5.5 编写程序



4.1.3 实施步骤

(一) 参考程序



（二）具体操作 1. 首先打开 Mind+ 软件，在“上传模式”下，在“扩展”中选择“主控板-掌控板”与“网络服务-WiFi、MQTT”进行安装。将掌控板通过数据线连接到电脑，驱动安装完成后，点击“连接设备”中“COMxx-CP210x”即可。

2. 手动修改可连接的 WiFi 热点名与密码。

3. 设置 MQTT 初始化参数。选择 SIoT 物联网平台，服务器地址为本地 IP 地址，账号密码即 SIoT 使用的账号密码，Topic_0 为 “项目 ID/名称”。

4. 将程序 “上传到设备” 进行测试。

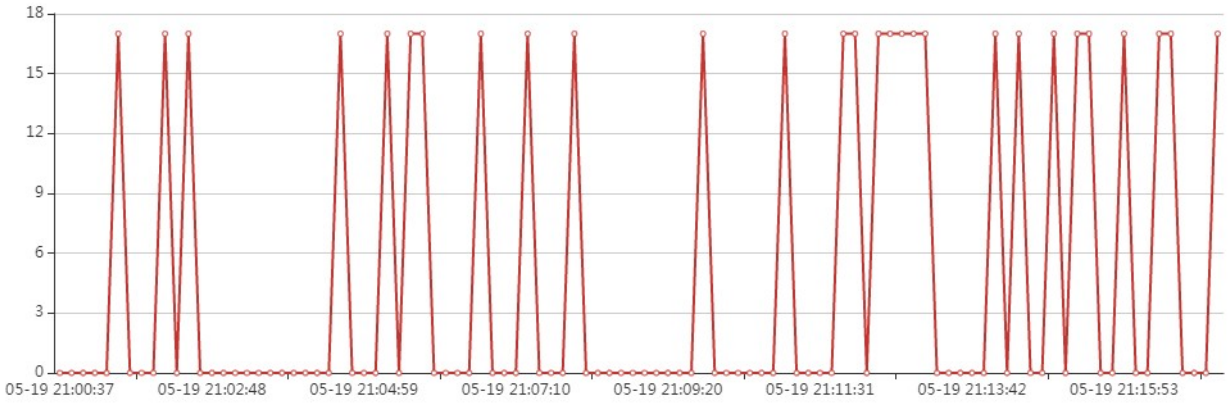
（三）运行结果

1. 掌控板屏幕显示当前温度值。

2. 当温度值高于 30 摄氏度时，红灯全部亮。

说明此时环境温度较高，用户应做好防晒及防中暑工作。

3.SIoT 平台设备每间隔 10 秒接受一条信息。



（四）数据导出

在设备列表界面，点击“导出查询结果”可将消息记录导出。

导出后自动生成 Excel 文件，用户可继续使用表格工具对数据进行分析和处理。

ID	主题	消息	时间
14406	my/002	0	2019/5/19 21:15
14405	my/002	17	2019/5/19 21:15
14404	my/002	17	2019/5/19 21:15
14403	my/002	17	2019/5/19 21:15
14402	my/002	17	2019/5/19 21:14
14401	my/002	16	2019/5/19 21:14
14400	my/002	16	2019/5/19 21:14
14399	my/002	17	2019/5/19 21:14
14398	my/002	17	2019/5/19 21:14
14397	my/002	17	2019/5/19 21:14
14396	my/002	17	2019/5/19 21:13
14395	my/002	17	2019/5/19 21:13
14394	my/002	17	2019/5/19 21:13
14393	my/002	17	2019/5/19 21:13
14392	my/002	0	2019/5/19 21:13
14391	my/002	15	2019/5/19 21:13
14390	my/002	17	2019/5/19 21:12
14389	my/002	17	2019/5/19 21:12
14388	my/002	17	2019/5/19 21:12
14387	my/002	16	2019/5/19 21:12
14386	my/002	0	2019/5/19 21:12
14385	my/002	17	2019/5/19 21:12
14384	my/002	17	2019/5/19 21:11

（五）数据分析（天津师范大学一天室外温度）

1. 数据筛选

使用 Excel “筛选” 功能将可用的温度信息筛选出来（温度为 0 的情况推测为掌控板与传感器间的引脚接触不良或网络不稳定等原因造成，故而将温度为 0 的数据进行筛选删除）。

2. 制作图表

使用 Excel “插入图表” 功能，绘制折线图。

3. 一天当中最高温

分析天津师范大学室外温度（2019 年 5 月 19 日）折线图中数据可知，11:57 开始出现当天最高温度 35℃，12:00—13:00 的温度值稳定在 34℃ 左右，14:09 再一次出现 35℃，之后温度值徘徊在 34℃—35℃，15:00 后温度值开始降低。

由上述分析可得，5 月 19 日这一天，接近正午 12:00 时第一次出现当天温度最高值，较为集中的温度最高值在午后 14：00 左右。这一结果与“一天之中，气温最高值出现在午后 14 时”的说法基本一致。测量结果受地区、气候等因素的影响，且此次测量结果仅为一天的测量数据，缺少一定的普适性，因此存在微小差异尚在情理之中。

4. 一天当中最低温

分析上述折线图可知，晚上 20:30 出现当天最低温度 17℃，日出前后温度值稳定在 19℃ 左右。此结果与“一天之中，气温最低值出现在日出前后”的说法存在差异。分析差异原因可能为：当天晚上 20:00 左右天气突变，开始下雨，21:00 左右雨停，所以导致气温最低温度出现在晚上 20:30，而非日出前后。与最高温度一样，此次测量结果仅为一天的测量数据，缺少一定的普适性。

5. 当日温差

当日温差在 8 ℃ 左右，用户应注意添加衣物。

4.1.4 代码分享

代码下载地址：https://github.com/vvlink/SIoT/tree/master/examples/Mind%2B

4.2 科学探究之热辐射实验

虚谷物联项目的核心工作就是利用物联网技术采集数据。借助 SIoT 物联网平台，学生不用注册和设置，一键启动，随时使用。通过对实时收集的数据进行合理分析，学生可以得出科学的分析结果，养成“用数据说话”的意识和习惯。

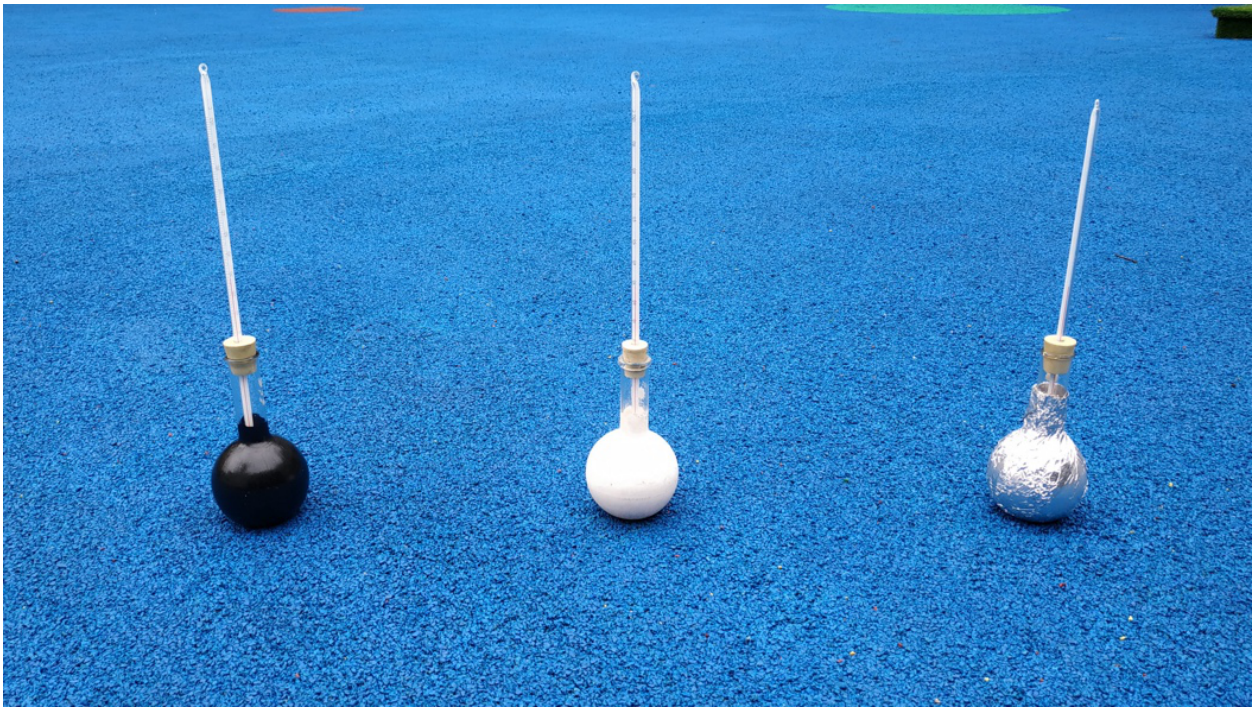
** 案例提供：狄勇（宁波市海曙区广济中心小学）

4.2.1 案例描述

热辐射问题在小学五年级、初中科学课中都有涉及。科学课堂上的实验方法，一般是采用不同颜色的纸袋包裹温度计，或者如下图将温度计插入涂成不同颜色的烧瓶，放太阳下暴晒，随时间推移记录温度数据，以验

证不同颜色物体吸热本领的大小。

以往实际教学中采用的这些实验方法，需要学生长时间在阳光下暴晒观察，依靠人工读数、计时、记录，不但精度不足，也难以在有限的课堂时间内获得明显的实验结果。实验中还会因学生不经意对阳光的遮挡等因素，增加影响实验准确性的多余变量。如果能由学生亲手搭建一套可自动计时、记录温度的装置，不但能实现实验观察、值守的无人化，还能通过这种 DIY 数字化实验工具进行科学探究的方式，培养学生着眼于未来的核心素养。随着 SIoT 开源物联网平台的推出，小学生也可以基于掌控板便捷使用物联网，使得前述构想可以在课堂轻松实现。



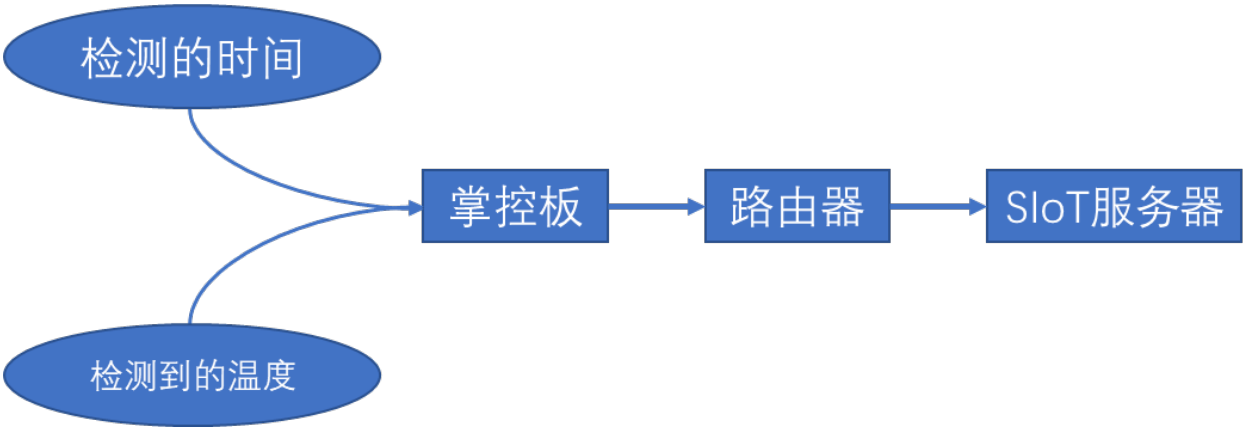
4.2.2 实验过程

一、用 SIoT 与掌控板做热辐射实验的原理

在动手搭建实验平台前，我们先梳理一下制作思路。参考教育科学出版社小学《科学》五年级上册“怎样得到更多的光和热”一课的实验记录表，可见在装置设计时，需要在物联网平台记录时间和对应的温度两项数据。教材中设计的 2 分钟间隔，对于已实现自动记录的实验平台而言有些过长，我们可以设计为 1 分钟，甚至 10 秒钟的时间间隔，让细微的温度变化都得以呈现。

纸的种类	刚开始的温度（℃）	2 分钟	4 分钟	6 分钟	8 分钟	10 分钟	我们的发现
黑色							
粉色							
铝箔纸							
黑色蜡光纸							
白纸							

用于记录数据的 SIoT 服务器应与掌控板部署在同一个局域网内，我们可以在教室台式机、教师笔记本电脑上轻松搭建 SIoT 服务器，其他设备在知道路由器分配给这台电脑的 IP 地址后，可以利用 WIFI 访问 SIoT 服务器。这些设备可以是电脑、手机、micro:bit、Arduino 等，当然也包括本文采用的自带 WIFI 模块的掌控板。装置工作流程如下图：



用于检测温度的传感器有不少选择，比如 DHT11、BMP280、LM35 等。考虑到 LM35 传感器更为常见，几乎是所有 Arduino 套件的标配，且测量温度范围满足需求，所以本实验采用 LM35 线性温度传感器。

二、检测装置的硬件搭建

我们将课程设定为面向全体学生的 STEM 课程，所以建议采用模块化的 LM35 传感器，尽可能避免因为线路连接上的不便导致无法在核心内容上给予学生必要的正向反馈。器材需求视实验分组数量而定，建议每个小组与测试的颜色一一对应。单组所需材料包括：micro I/O extend 扩展板 x1

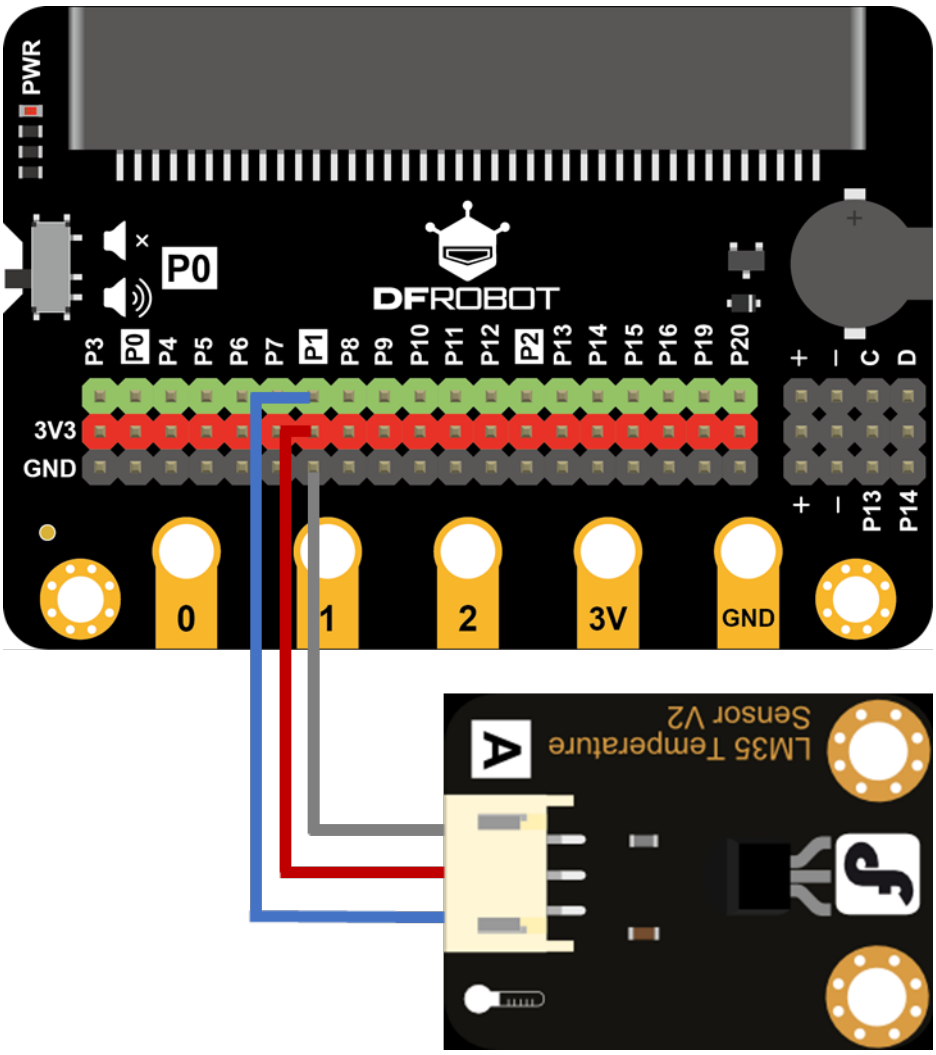
掌控板 x1

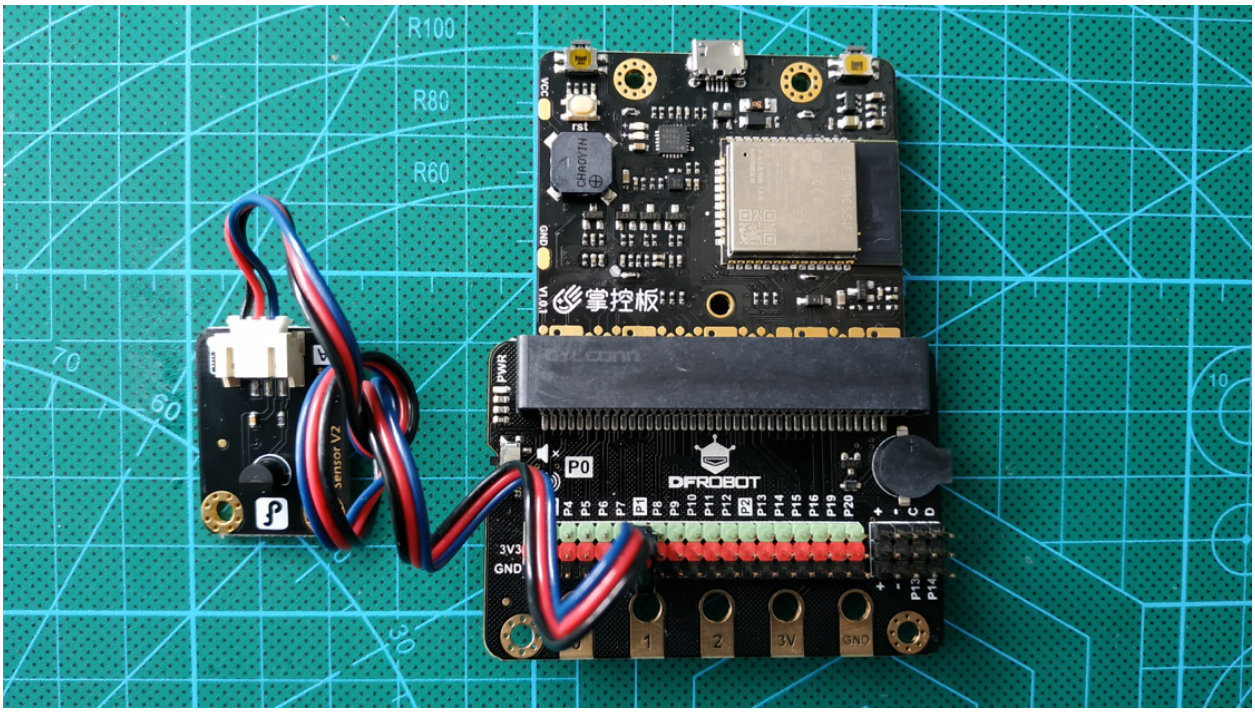
LM35 线性温度传感器 ×1

烧瓶 ×1

轻质黏土若干

电路连接示意图：


PWR
P0
DFROBOT
P3 P0 P4 P5 P6 P7 P1 P8 P9 P10 P11 P12 P2 P13 P14 P15 P16 P19 P20
3V3
GND
+ − C D
+ − P13 P14
0
1
2
3V
GND
LM35 Temperature Sensor V2



大部分支持 micro:bit 的扩展板可以兼容掌控板，但需要注意的要将掌控板“反插”到插槽。

将 LM35 传感器放入烧瓶后，需要用轻质黏土封堵瓶口，避免瓶内空气与外界对流，以获得更好的实验效果。

三、SIoT 服务器搭建

SIoT 的使用手册可通过以下地址访问：https://siot.readthedocs.io/zh_CN/latest/

作为一个开源项目，SIoT 存放于 GitHub，点击使用手册的“文件下载”，根据电脑的操作系统选择相应版本软件包即可获得服务器程序。SIoT 支持 Linux、Mac、Windows，全面覆盖了常见操作系统。

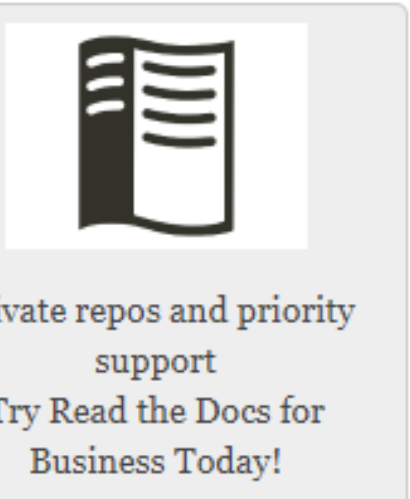
SIoT使用手册

导航

简介
安装和运行
客户端连接范例
典型应用案例
高级操作技巧
附录

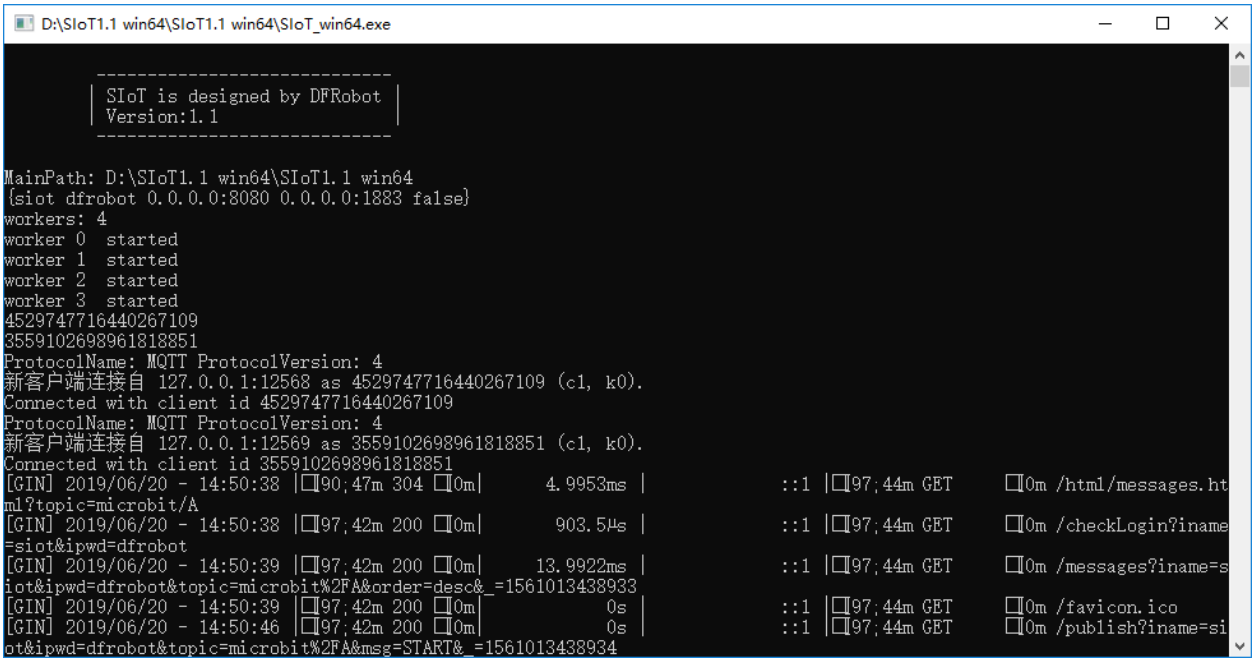
快速搜索

转向



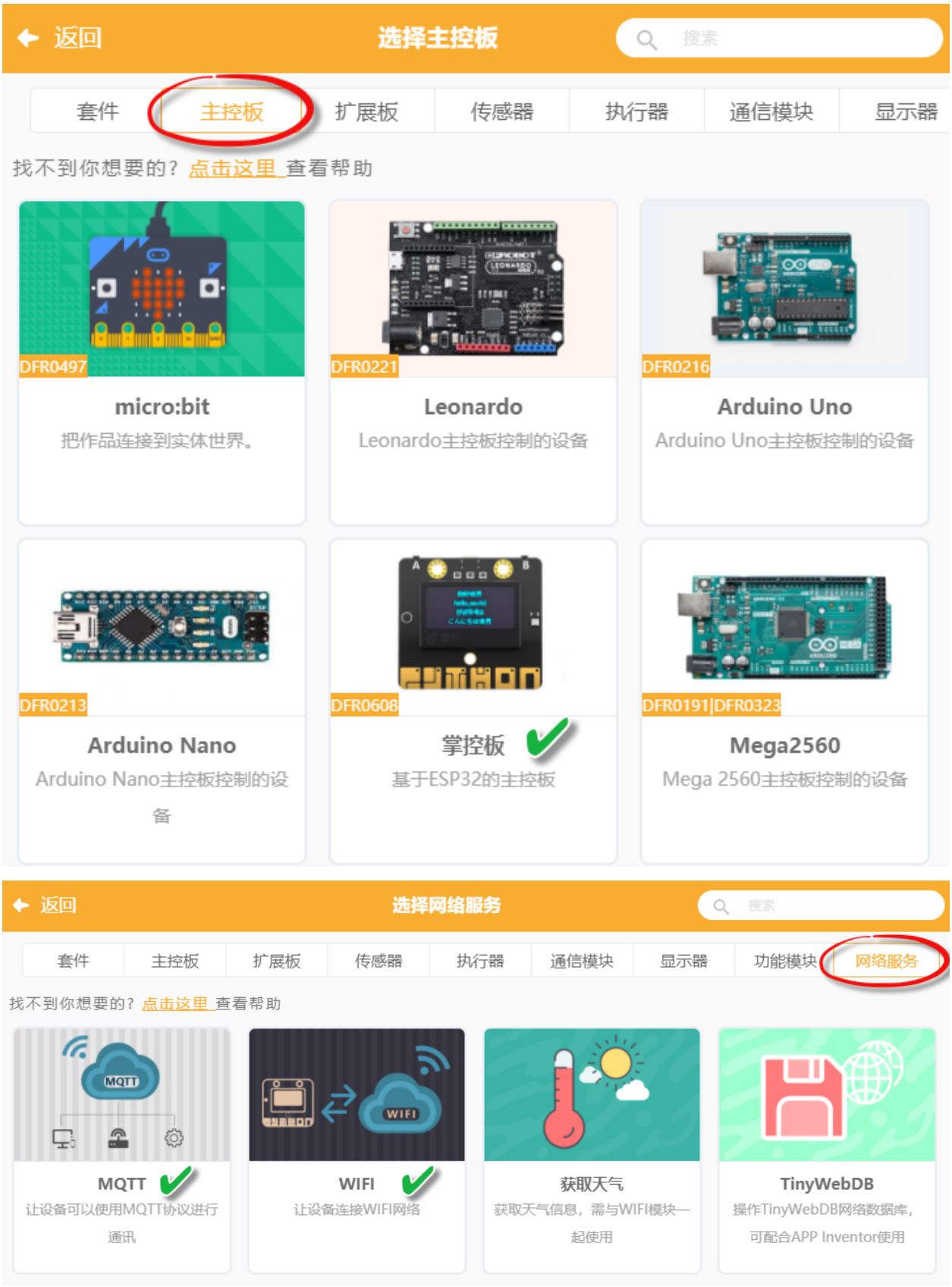
目录

不同于通常配置服务器的繁冗，部署 SIoT 服务器只需解压文件包后，双击运行服务器端程序即可。随后系统会弹出一个控制台窗口，滚屏显示日志信息，这样就算部署完毕了。

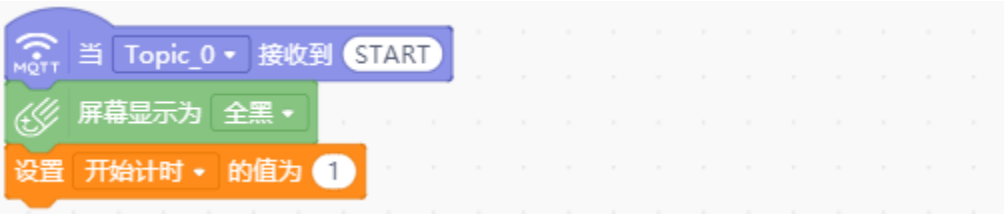


四、实验装置的程序设计

实验装置的程序，使用了 DFrobot 的 Mind+ 来编写。作为一款基于 Scratch3.0 开发的编辑器，Mind+ 可以让学生轻松地从 Scratch 迁移经验，非常适合普适性的 STEM 课程。将 Mind+ 切换到“上传模式”后，依次添加掌控板、MQTT、WIFI 三个模块。


返回
选择主控板
搜索
套件
主控板
扩展板
传感器
执行器
通信模块
显示器
找不到你想要的？点击这里 查看帮助
DFR0497
micro:bit
把作品连接到实体世界。
DFR0221
Leonardo
Leonardo主控板控制的设备
DFR0216
Arduino Uno
Arduino Uno主控板控制的设备
DFR0213
Arduino Nano
Arduino Nano主控板控制的设备
DFR0608
掌控板
基于ESP32的主控板
DFR0191|DFR0323
Mega2560
Mega 2560主控板控制的设备
返回
选择网络服务
搜索
套件
主控板
扩展板
传感器
执行器
通信模块
显示器
功能模块
网络服务
找不到你想要的？点击这里 查看帮助
MQTT
MQTT
让设备可以使用MQTT协议进行通讯
WIFI
WIFI
让设备连接WIFI网络
获取天气
获取天气信息，需与WIFI模块一起使用
TinyWebDB
操作TinyWebDB网络数据库，可配合APP Inventor使用

为了实验中可以将烧瓶摆放到位后再记录数据，程序设计为如果装置接收到“START”指令，才开始发送数据给 SIoT，避免了通电就发送无效数据。完整程序见下图：



要确保掌控板连上 SIoT，务必正确配置 MQTT 的初始化参数，初始化信息解读如图：

其中运行 SIOT 的服务器 IP 地址，可在安装了服务器的电脑上运行命令提示符，使用 ipconfig 命令获得。

将程序上传到掌控板后，如果配置正确，且局域网网络通畅，根据我们设计的程序，掌控板的 OLED 屏应显示提示信息—— “SIOT 已连接”。

五、系统测试

1. 登录服务器

打开浏览器，如在服务器端，访问：http://localhost:8080，如通过局域网内其他设备访问，将地址中的“localhost”替换为服务器 ip 地址即可。

2. 登录后可以看到项目列表中出现了 myPython，这便是我们新建的项目。在掌控向 SIOT 服务器发送第一条数据时（一般会将这个“握手信息”放在主程序 MQTT 连接成功后），便会在服务器建立掌控板程序中项目 ID 对应的项目。

点击“查看项目列表”——“查看消息”

3. 根据我们设计的程序，发送消息“START”后，实验装置开始上传数据。刷新页面后，可以看到更新后的数据记录，还能导出为 Excel 表格进行数据分析。

Topic	消息	时间
myPython/A	28.35938	2019-06-26 08:33:42
myPython/A	28.35938	2019-06-26 08:33:40
myPython/A	28.35938	2019-06-26 08:33:40
myPython/A	28.35938	2019-06-26 08:33:39
myPython/A	28.35938	2019-06-26 08:33:37
myPython/A	28.35938	2019-06-26 08:33:36
myPython/A	28.19824	2019-06-26 08:33:35
myPython/A	28.35938	2019-06-26 08:33:34
myPython/A	25.13672	2019-06-26 08:33:33
myPython/A	START	2019-06-26 08:33:33

	A	B	C	D
1	ID	主题	消息	时间
2	6393	myPython/A	25.45898	2019/6/25 20:50
3	6392	myPython/A	25.78125	2019/6/25 20:50
4	6391	myPython/A	25.78125	2019/6/25 20:50
5	6390	myPython/A	25.45898	2019/6/25 20:50
6	6389	myPython/A	25.45898	2019/6/25 20:50
7	6388	myPython/A	25.45898	2019/6/25 20:50
8	6387	myPython/A	25.78125	2019/6/25 20:50
9	6386	myPython/A	25.45898	2019/6/25 20:50
10	6385	myPython/A	25.29785	2019/6/25 20:50
11	6384	myPython/A	24.81445	2019/6/25 20:50
12	6383	myPython/A	25.94238	2019/6/25 20:50
13	6382	myPython/A	25.13672	2019/6/25 20:50
14	6381	myPython/A	25.62012	2019/6/25 20:50
15	6380	myPython/A	25.78125	2019/6/25 20:50
16	6379	myPython/A	25.62012	2019/6/25 20:50
17	6378	myPython/A	25.78125	2019/6/25 20:50
18	6377	myPython/A	25.45898	2019/6/25 20:50
19	6376	myPython/A	22.55859	2019/6/25 20:50
20	6375	myPython/A	START	2019/6/25 20:50

通过测试我们发现 SIOT 的出现，让课堂搭建物联网服务器轻而易举，即便没有任何信息技术学科背景的师生都可一键完成服务器部署，突破了公网物联网平台应用于课堂教学时账号注册、账号管理、数据容量限制的掣肘，恰到好处地满足了日常教学需求。人民教育出版社高中《物理》第一册中，有篇题为“借助传感器用计算机测速度”的内容，其中提及“随着信息技术的发展，中学物理的实验手段也在不断进步。”，并指出这种实验手段的进步，使得“同学们可以减少重复性操作，用更多的时间和精力对物理过程进行分析”。从中我们可以看到科学学科对于信息技术的关注，而信息技术也推动和影响着其它学科的变革，SIOT 的出现将加速这种变革。如果我们从 STEM 的角度出发，让孩子们自行 DIY 数字化实验装置，其过程价值更是不言而喻。数字化实验室出于成本一直难以普及，但是掌控板 +SIOT 可以替代其中很大一部分功能，加上扩展板后，原有的 Arduino 传感器基本上可以通用，轻松实现编程、接线、联网，小学生都容易上手，成本低到农村学校也买得起。这会不会是国内 STEM 课程普及和落地的一条务实路径呢?

4.3 科学探究之通用定时测量工具

一位做科学教育的朋友，听说虚谷物联项目后，非要我现场演示一番。演示完后，又提出需求，说科学老师不会写代码，能不能提供一个写好代码的通用系统，能够实现自动采集的功能。

认真想了想，我认为这个要求貌似并不过分。因为科学采集无非就是那几个常见的传感器。除了 DH11 以外，与科学实验相关的大部分传感器都是通用的模拟量传感器。我可以在代码中确定某一个特定的引脚，使用的时候就往这个引脚接传感器就行了。

案例作者：谢作如，浙江省温州中学

4.3.1 案例描述

这是一个通用的物联网数据采集工具，基于掌控板和 SIoT 服务器程序。使用者只需将代码做简单的参数配置，如服务器 IP、Wi-Fi 的 ssid 和密码之类，然后下载程序即可采集数据。

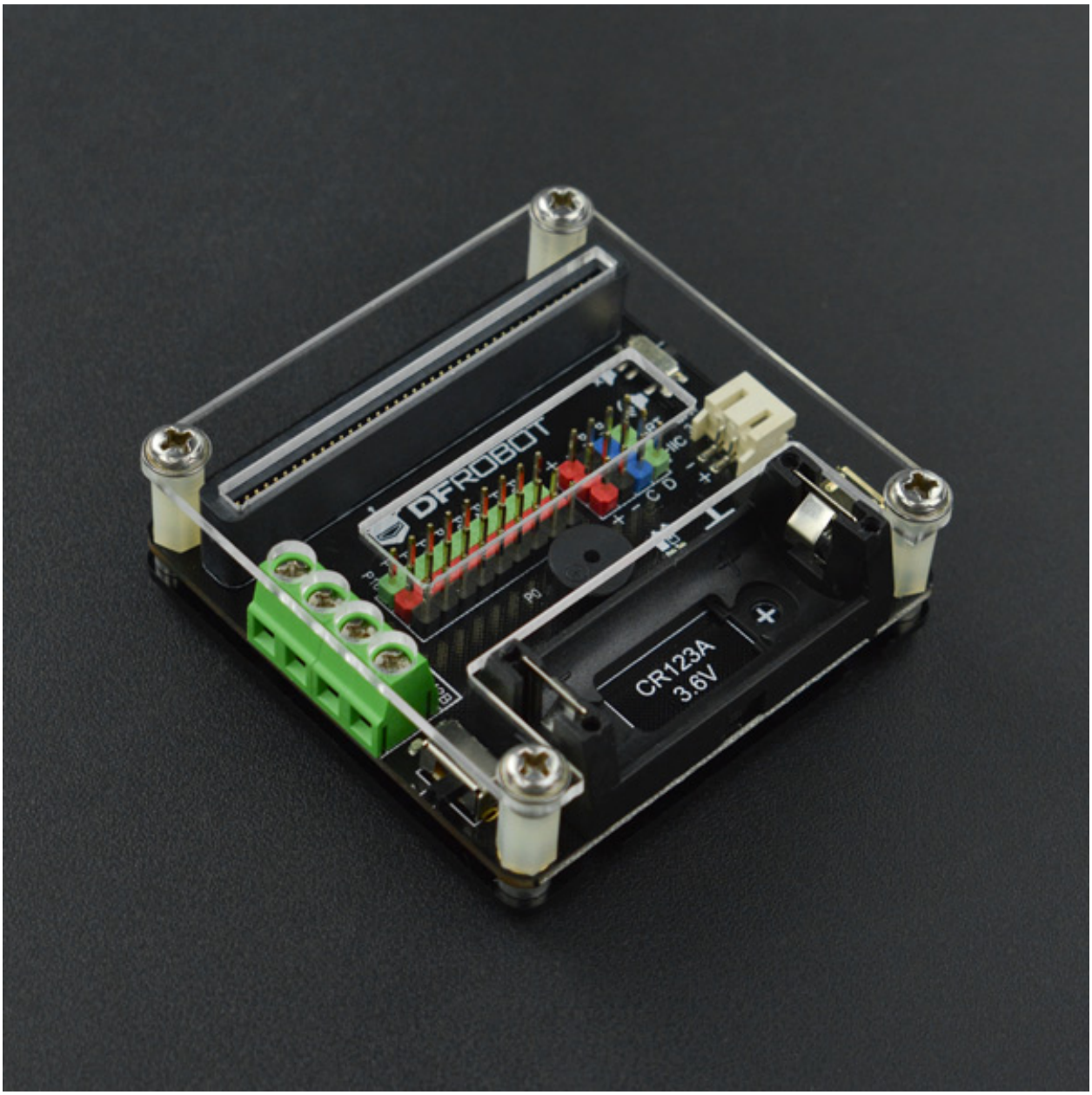
本程序默认采集 P1 引脚，借助 IO 扩展板，掌控板能够使用绝大多数的 Arduino 模拟传感器。

4.3.2 准备工作

1. 硬件准备

需要掌控板、相关扩展板和传感器，及其连接线。要测量什么就找什么传感器。

http://www.dfrobot.com.cn/category-223.html

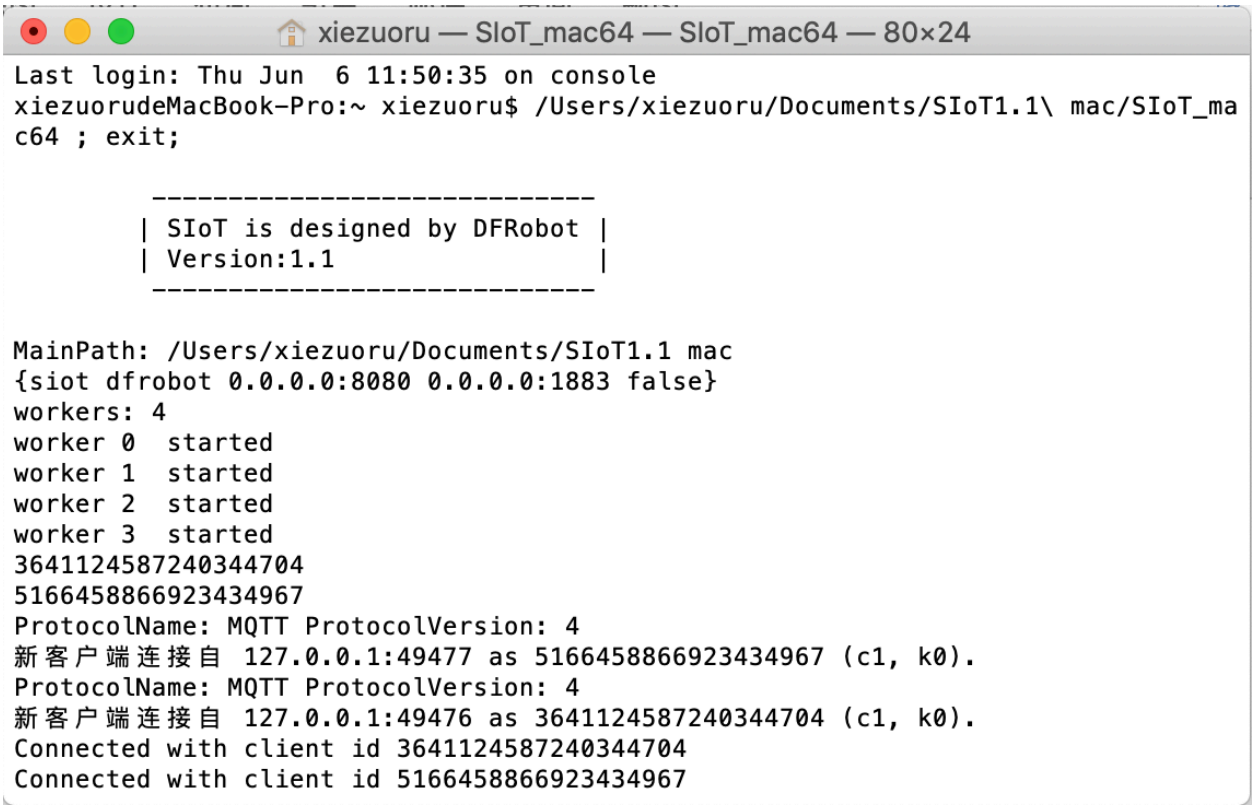


DF 的这个扩展板（micro:IO-BOX 电机驱动扩展板）特别好用，自带了一个可充电的 CR123A 锂电池。
http://www.dfrobot.com.cn/goods-1923.html

2. 软件准备

1) 搭建 SIoT 服务器

直接双击点击与系统匹配的 SIoT 运行文件，屏幕会弹出一个黑色的 CMD 窗口，在配置中请不要关闭它。



2) 修改 mPythonX 代码

代码用 mPythonX 编写，要根据具体情况修改代码中的服务器 IP、Wi-Fi 的 ssid 和密码等信息。


连接 Wi-Fi 名称 jf 密码 20040404
OLED 第 1 行显示 Wi-Fi 配置信息 IP 模式 普通
OLED 显示生效
MQTT-Easy IoT
服务器 172.20.10.2
Client ID xzr
Iot_id siot
Iot_pwd dfrobot
随便填写，可以留空
连接 MQTT
OLED 第 2 行显示 连接MQTT服务器成功 模式 普通
OLED 显示生效
一直重复
执行 等待主题消息

代码解释

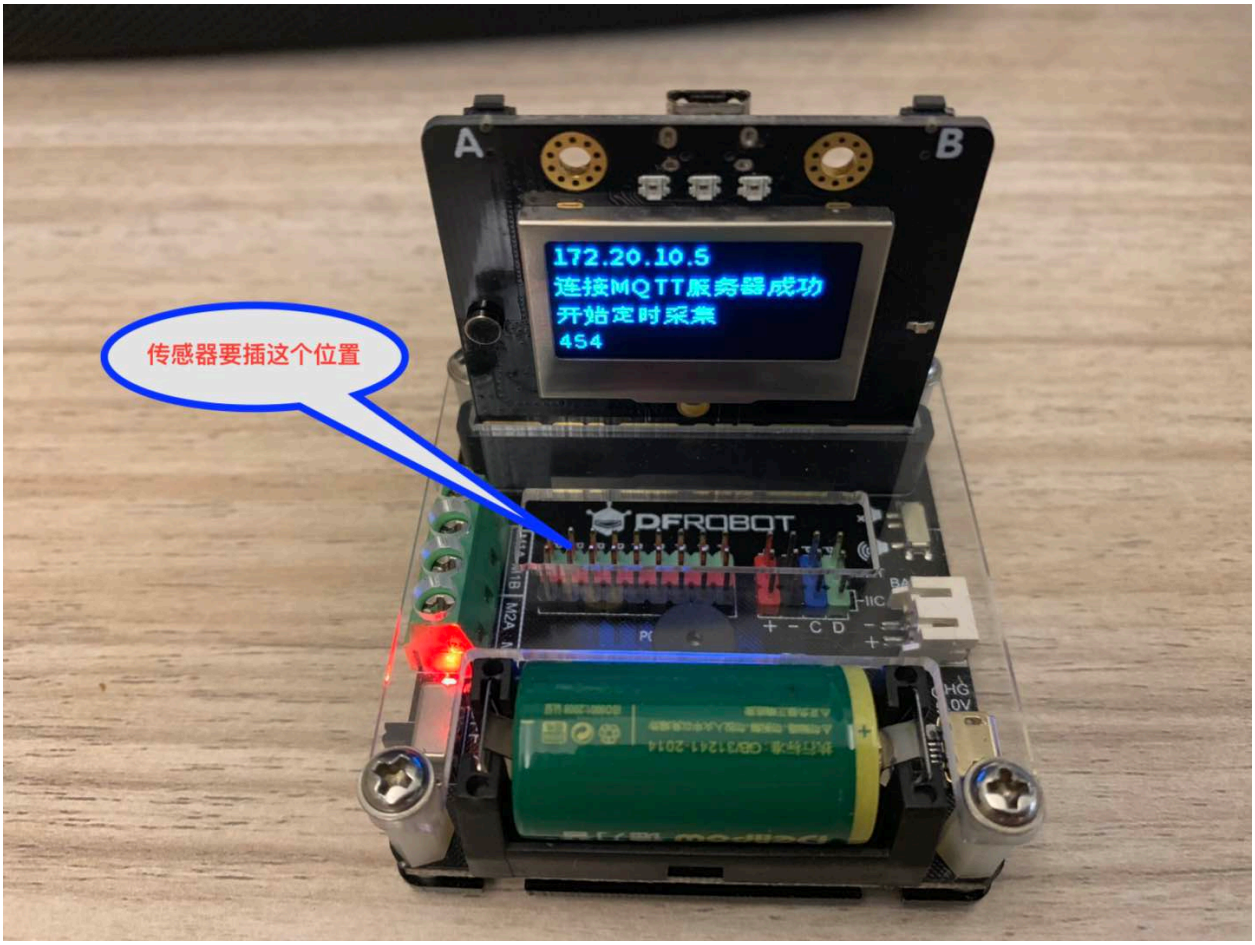
我设定的 Topicid（主题）是“stem/p1”，表示传感器要接到 P1。按下 A 按键开始工作，每隔 2 秒采集一次；按下 B 键则停止采集。

4.3.3 使用步骤

1. 采集数据

接上电源后，掌控板的显示屏上会先出现 IP 地址，表示连上了 Wi-Fi。然后显示“连接 MQTT 服务器成功”。

按下 A 键，掌控板开始工作了。如果发送消息成功，显示屏最下面一行会出现数字。



这个图里，我还没有插入传感器啊。

DF 的这个扩展板（micro:IO-BOX 电机驱动扩展板）特别好用，自带了一个可充电的 CR123A 锂电池。这个图里，我还没有插入传感器啊。

2. 查看数据

输入 http://127.0.0.1:8080，在网页中找到 stem 项目的 p1 主题，就可以看到一个图表，直观显示数据。

3. 导出数据

这些数据都可以通过这个网页导出为 xls 文件。

Topic	消息	时间
stem/p1	19	2019-06-07 00:50:45
stem/p1	22	2019-06-07 00:50:44
stem/p1	69	2019-06-07 00:50:42
stem/p1	32	2019-06-07 00:50:40
stem/p1	29	2019-06-07 00:50:37
stem/p1	39	2019-06-07 00:50:36
stem/p1	71	2019-06-07 00:50:33
stem/p1	64	2019-06-07 00:50:32
stem/p1	59	2019-06-07 00:50:29
stem/p1	64	2019-06-07 00:50:27
stem/p1	48	2019-06-07 00:50:25
stem/p1	44	2019-06-07 00:50:24
stem/p1	47	2019-06-07 00:50:22

4.3.4 其他说明

1. 一般来说，掌控板和电脑连接 Wi-Fi，IP 地址是不会变化的，我常常使用手机开热点的形式。

2. 如果要采集的数据时间间隔很多，一秒钟要很多次的那种，请先合并在一个 json 中，再提交。

4.3.5 代码分享

代码下载地址：https://github.com/vvlink/SIoT/tree/master/examples/mPythonX

4.4 互动媒体之物联网数据呈现

4.4.1 准备工作

步骤————=——

4.4.2 参考代码

4.5 智能家居之远程控制

利用 SIoT，可以搭建一个支持远程控制的智能家居模型。

4.5.1 准备工作

在 Mind+ V1.5.5 的物联网模块——MQTT 中，可以通过添加扩展掌控后提供的 WIFI 模块，与物联网平台连接，从而实现掌控板与 web 端、app 端的互联。用户可以根据实际需要选择阿里云、Easy IoT、OneNet 和 SIoT 四个平台物联网平台。我们在这里简单介绍如何使用 SIoT 平台来远程控制掌控板、Mind+，实现掌控板将数据上传到 SIoT，SIoT 同时将数据反馈到 Mind+ 界面的效果。SIoT 可以同时将消息发送至掌控板与 Mind+。

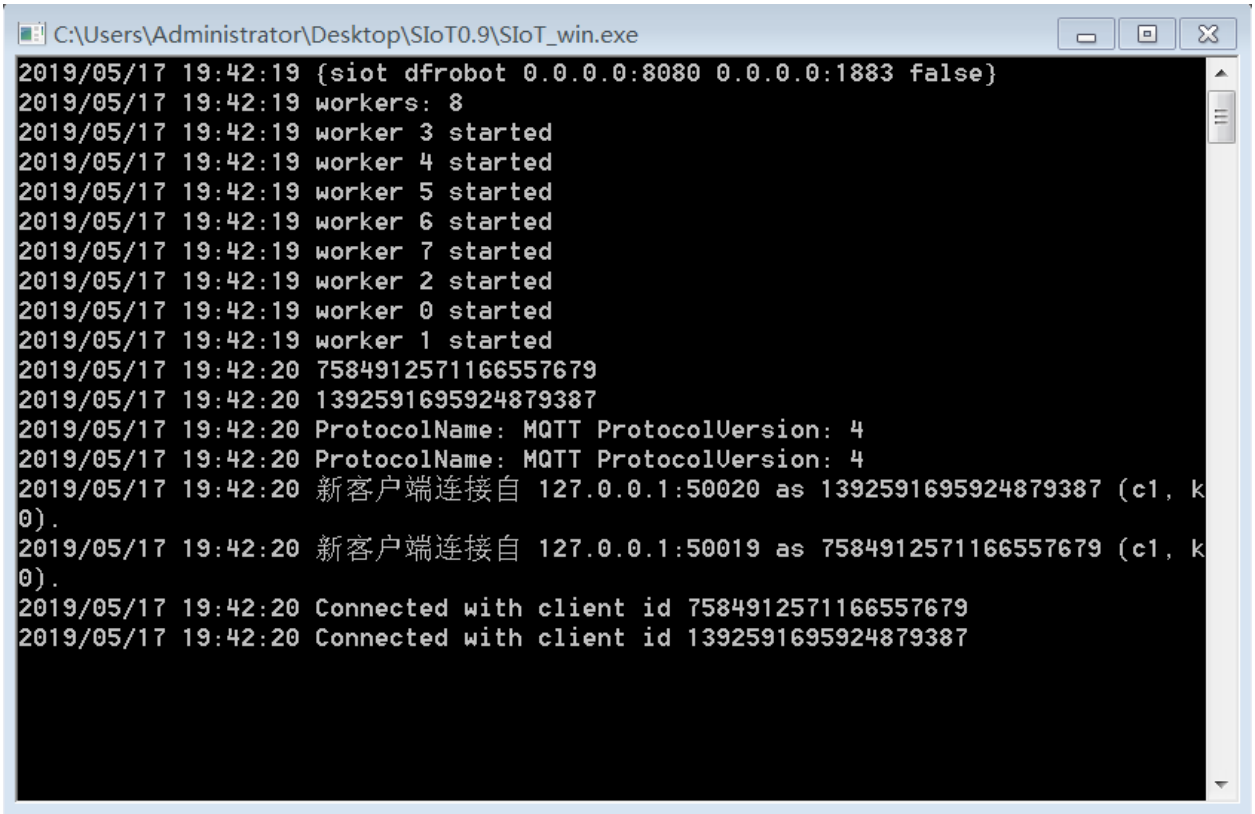
（一）硬件准备

掌控板及其连接线

（二）软件准备

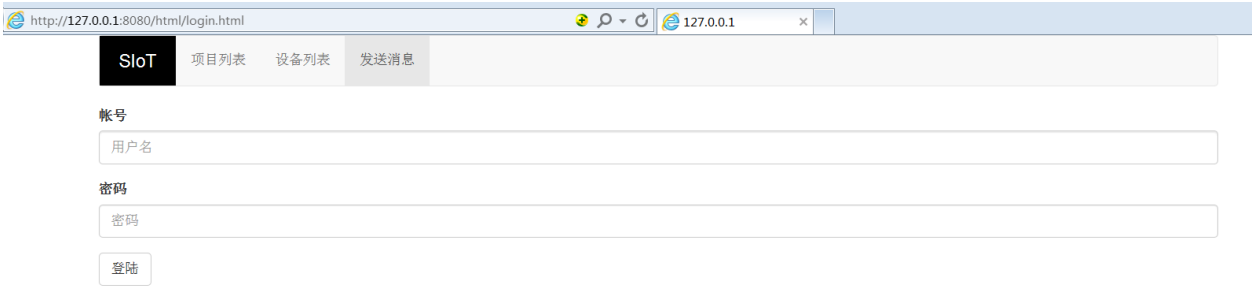
1. 搭建 SIoT 服务器

直接双击点击与系统匹配的 SIoT 运行文件，屏幕会弹出一个黑色的 CMD 窗口，在使用过程中请不要关闭它。

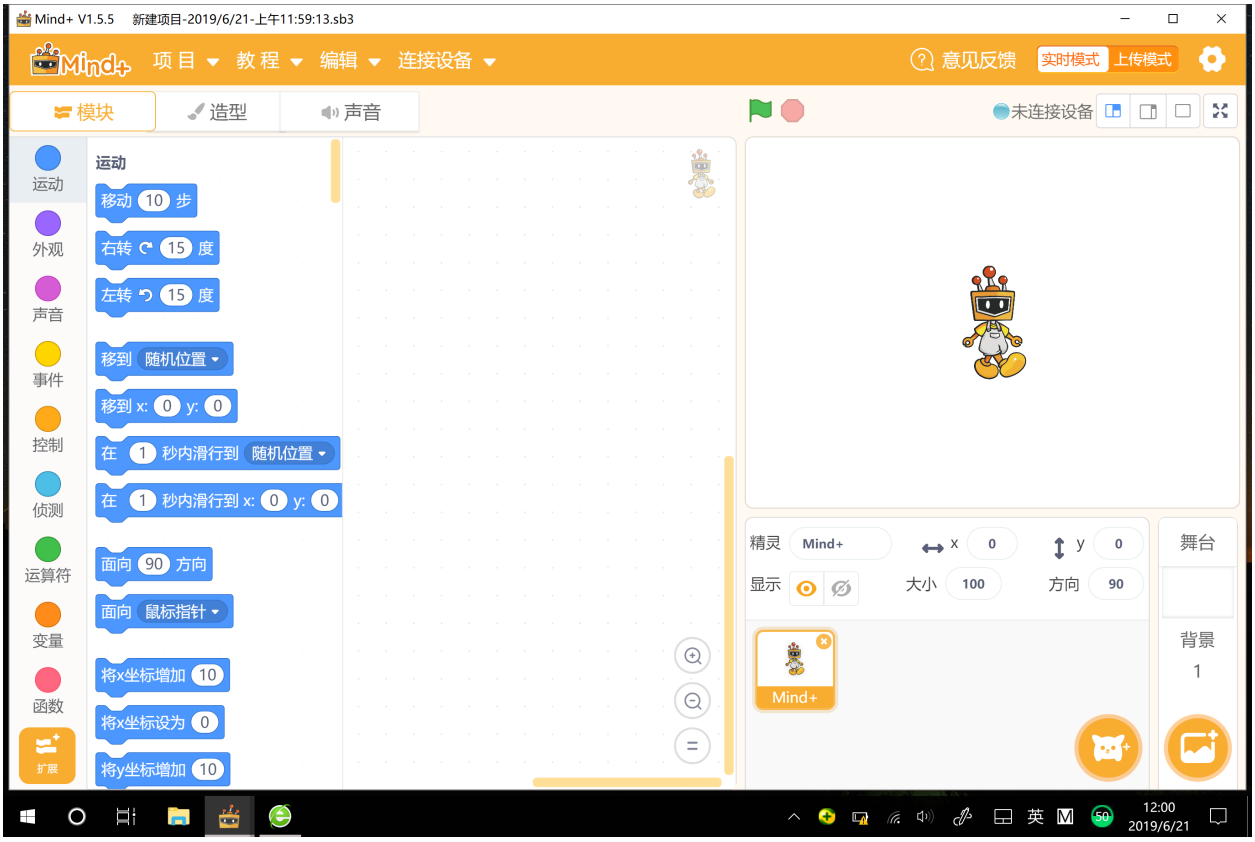


2. 登录 SIoT 平台

打开浏览器，输入 url：http://localhost:8080。



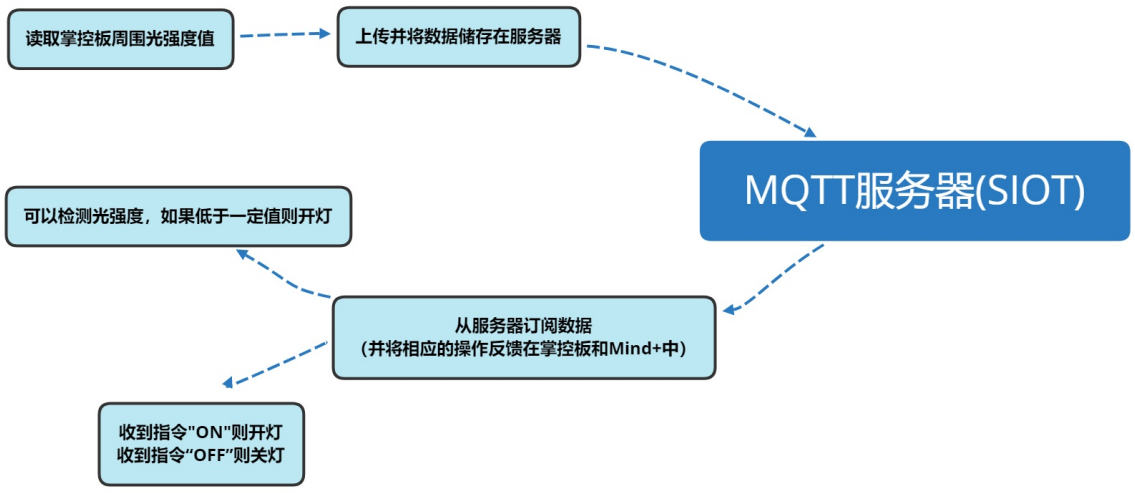
3. 打开 Mind+ V1.5.5 编写程序



4.5.2 步骤

为了方便大家的理解，我们将远程控制实现的功能用以下系统流程图进行展示。

系统流程图



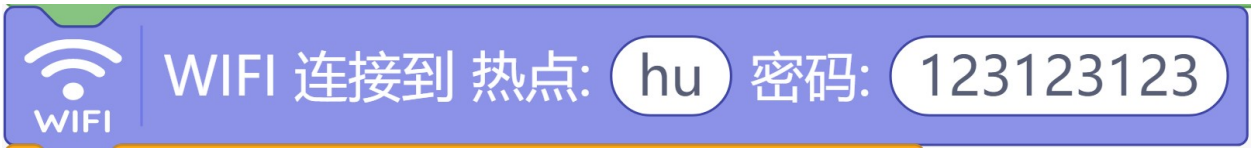
（一）受控制端（掌控板）

1. 参考程序

2. 具体操作

(1) 打开 Mind+ 后，在“上传模式”下，在“扩展”中选择“主控板-掌控板”与“网络服务-WiFi、MQTT”进行安装。将掌控板通过数据线连接到电脑，驱动安装完成后，点击“连接设备”中“COMxx-CP210x”即可。

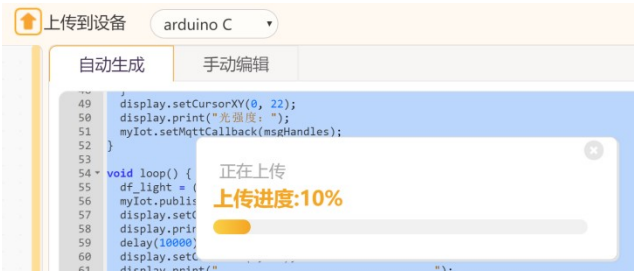
（2）手动修改可连接的 WiFi 热点名与密码。



（3）设置 MQTT 初始化参数。选择 SIoT 物联网平台，服务器地址为本地 IP 地址，账号密码即 SIoT 使用的账号密码，Topic_0 为 “项目 ID/名称”。

（4）当 MQTT 接收到消息 on 时，全部灯泡亮白光；当 MQTT 接收到 off 时，全部灯泡亮红光。消息内容可以进行修改。

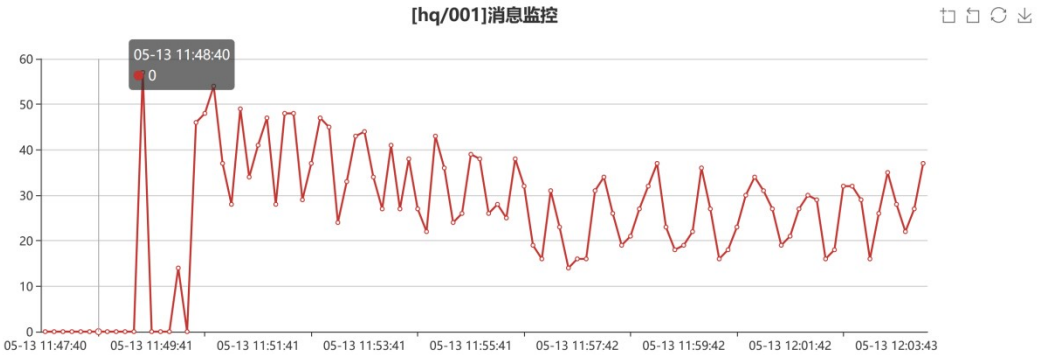
（5）将程序 “上传到设备” 进行测试。



3. 运行结果

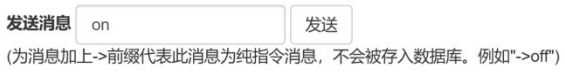
掌控板屏幕上显示以下内容。

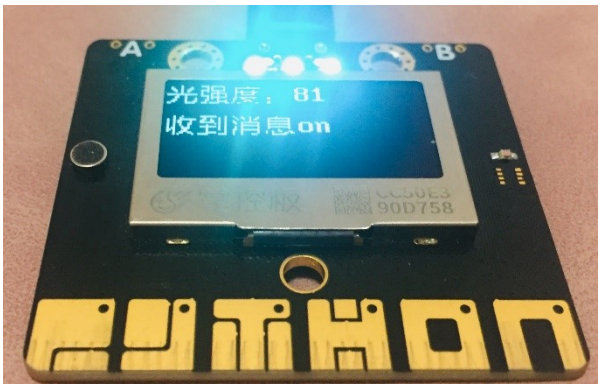
SIoT 平台设备每间隔 10 秒接受一条信息。



Topic	消息	时间
hq/001	37	05-13 12:04:12

在 SIoT 平台给掌控板发送消息“on”。掌控板的小灯全部变成了白色灯，显示文字。相同操作，发送消息“off”，灯泡变色。测试成功。



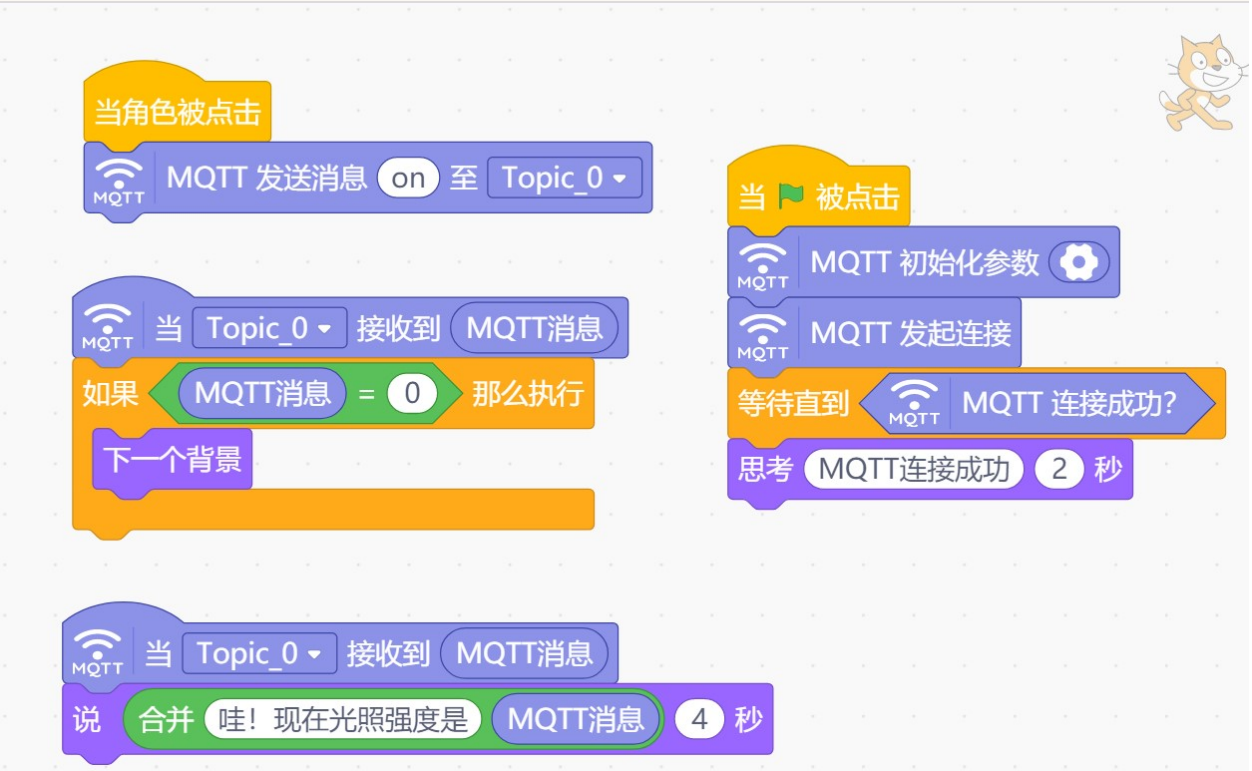


当对 P、O、H、N、T、Y 按键进行触摸时，掌控板发出相应声响。

当 A 按钮被按下时，屏幕显示文字“开心快乐每一天”。当 B 按钮按下时，显示彩灯。当 A+B 按钮按下时，灯全部灭掉。

（二）控制端（Mind+ 实时模式）

1. 参考程序



2. 具体操作

（1）单击 Mind+“实时模式”进行代码编写。

（2）设置 MQTT 初始化参数。选择 SIoT 物联网平台，服务器地址为本地 IP 地址，账号密码即 SIoT 使用的账号密码，Topic_0 与“上传模式”下 Topic_0 保持一致。

(3) 点击小猫，发送消息“on”至 SIoT，对掌控板作出指令。消息内容可以进行修改。

3. 运行结果

单击绿旗，角色显示“MQTT 连接成功”。角色说“哇！现在光照强度是 210”，由掌控板测得的数据反馈至 SIoT，通过 MQTT 同步给小猫角色，数据实时更新。

当光照强度为 0 时，舞台背景更换。


哇！现在光照强度是0

单击角色，发送消息“on”至 Topic_0，掌控板全部灯泡变为白色，测试成功。

4.5.3 参考代码

代码下载地址：https://github.com/vvlink/SIoT/tree/master/examples/Mind%2B

4.6 互动游戏之联机足球对战

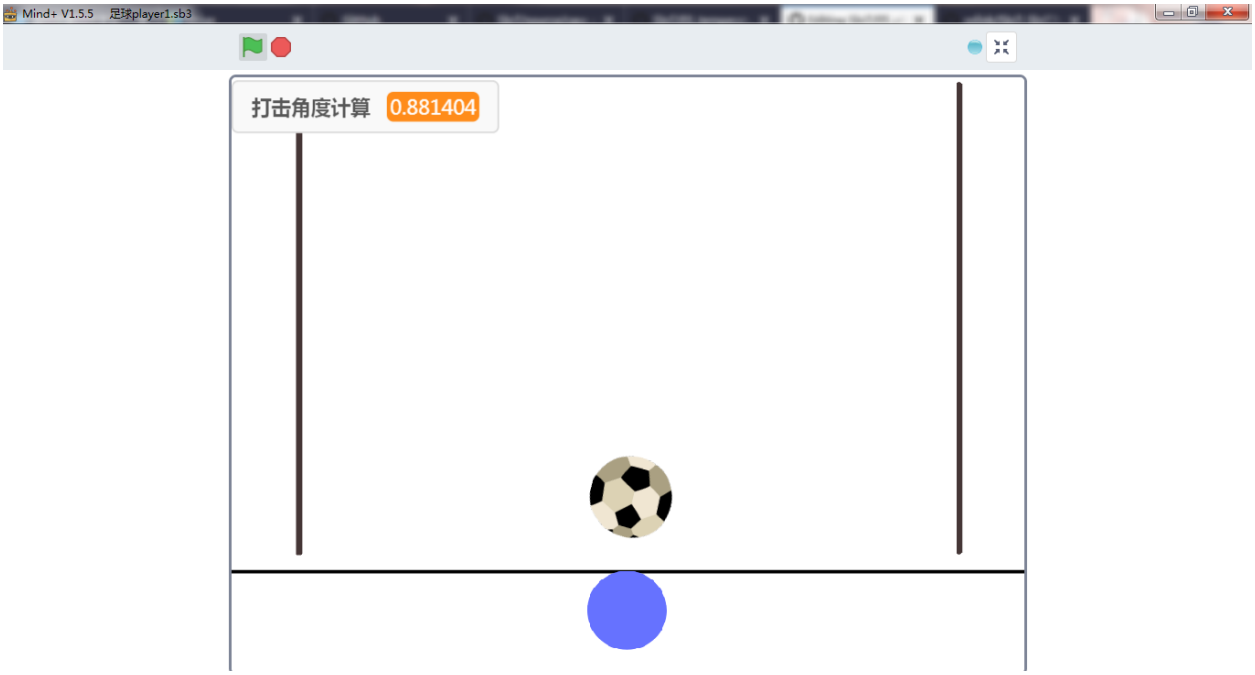
基于 SIoT 和和掌控板，可以设计一些多人竞赛的游戏，然后借助 Mind+ 实时呈现出来。

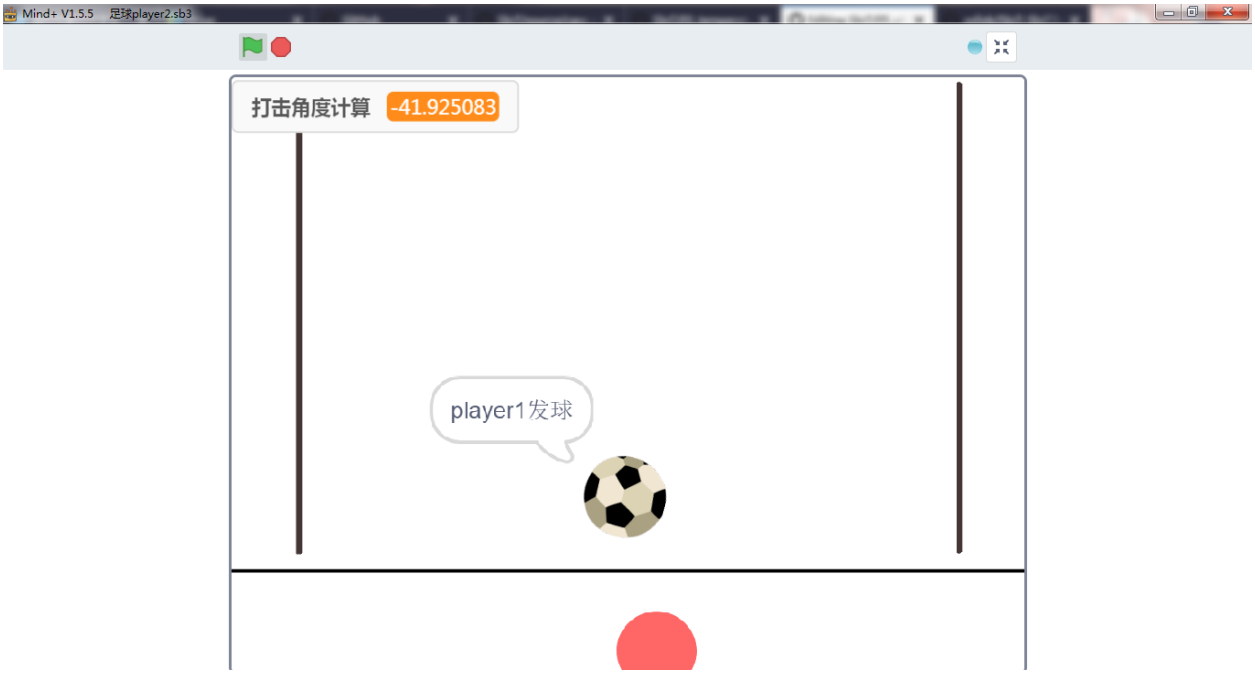
**案例作者：郑祥 **

4.6.1 案例 1：双人足球联机对战

游戏规则：

1. 有两个玩家：玩家 1 和玩家 2，由玩家 1 先发球。
2. 游戏开始，移动光标寻找合适角度触碰足球即可将球踢出去。
3. 场地中由三条黑线限制足球的移动范围，若触碰或超出黑线，则游戏结束。





4.6.2 原理介绍

本案例分为两个终端，分别为玩家 1（player1）和玩家 2（player2）。player1 端和 player2 端通过物联网平台 MQTT（SIoT）进行数据的交换，从而完成联机对战的功能。

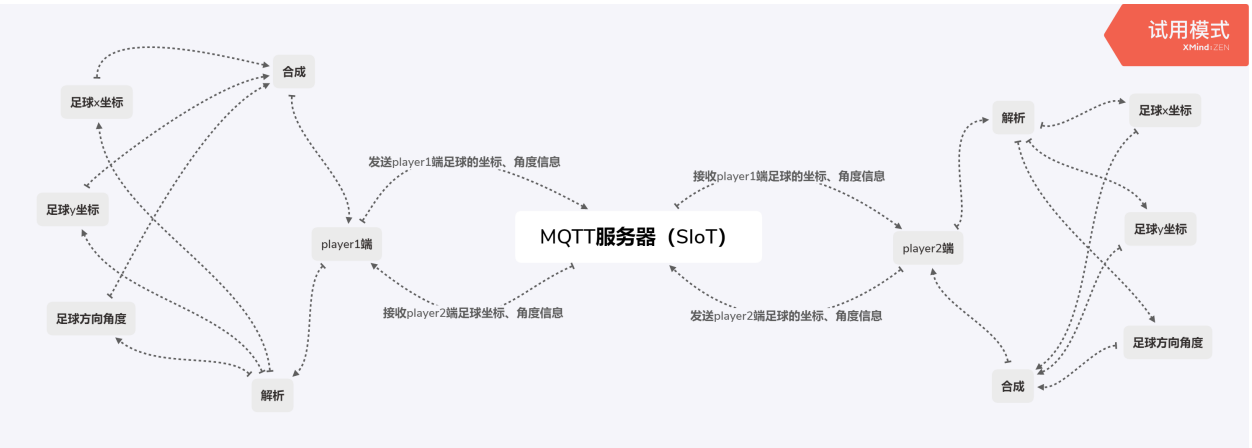
1、player1 端、player2 端与 MQTT 服务器（SIoT）端的数据交换

1）player1 端击球后将足球的坐标（x，y）和击球角度调制成一个字符串（后面用“string1”代替）发送至 MQTT 服务器（SIoT）。

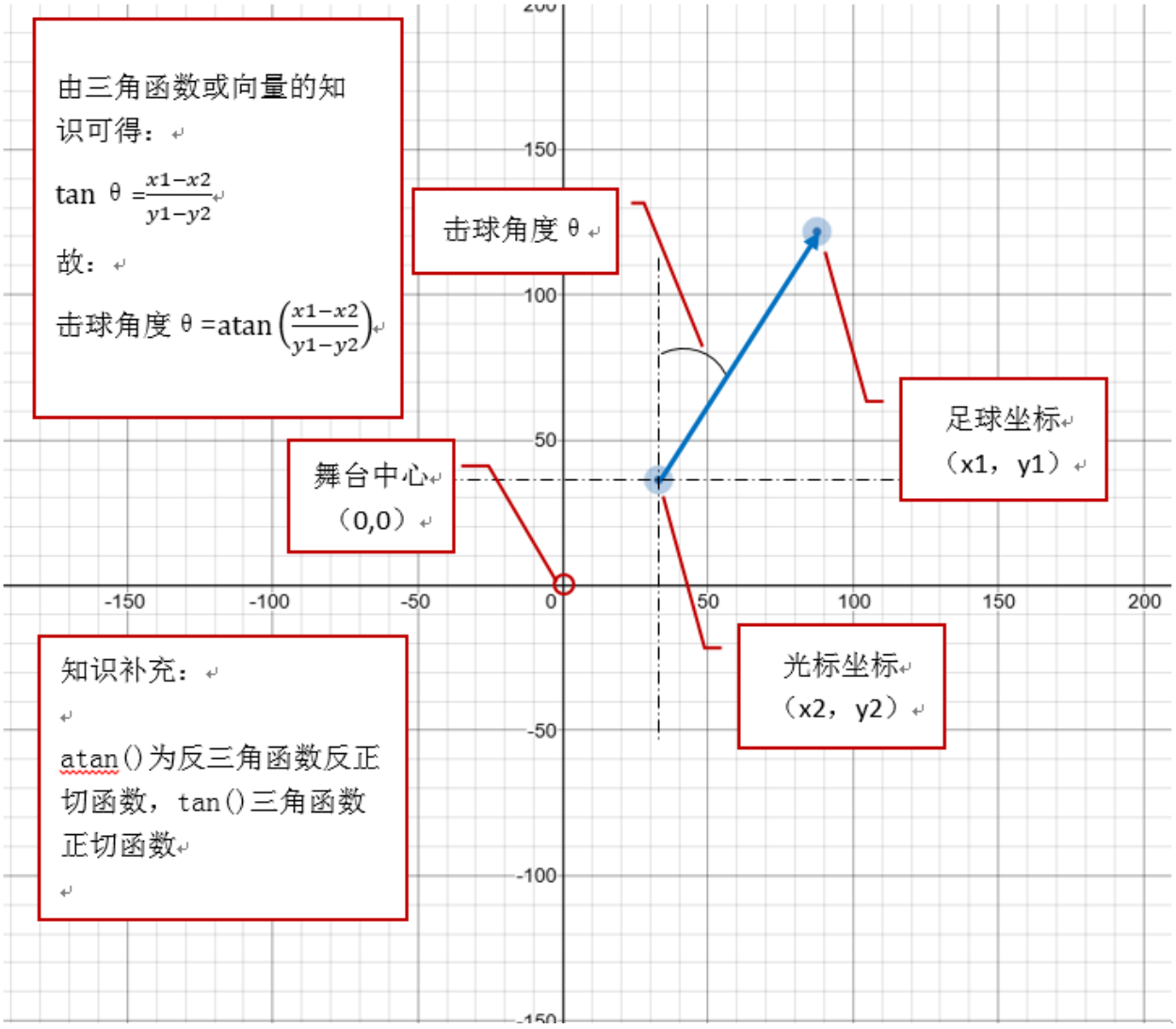
2）player2 端从 MQTT 服务器（SIoT）端获取字符串 string1，并解析出足球的坐标（x，y）和击球的角度，在屏幕上还原足球的位置和移动方向。

3）与 player1 端一样，player2 端击球后将足球的坐标（x，y）和击球角度调制成一个新的字符串（后面用“string2”代替）发送至 MQTT 服务器（SIoT）。

4）与 player2 端一样，player1 端从 MQTT 服务器（SIoT）端获取字符串 string2，并解析出足球的坐标（x，y）和击球的角度，在屏幕上还原足球的位置和移动方向。



2、击球角度的计算击球角度的计算，是本案例的一个十分关键的部分。本案例中通过已知的光标坐标（x2，y2）、足球坐标（x1，y1）和中学中所学的数学知识反三角函数（反正切函数 atan()）即可得出击球角度。



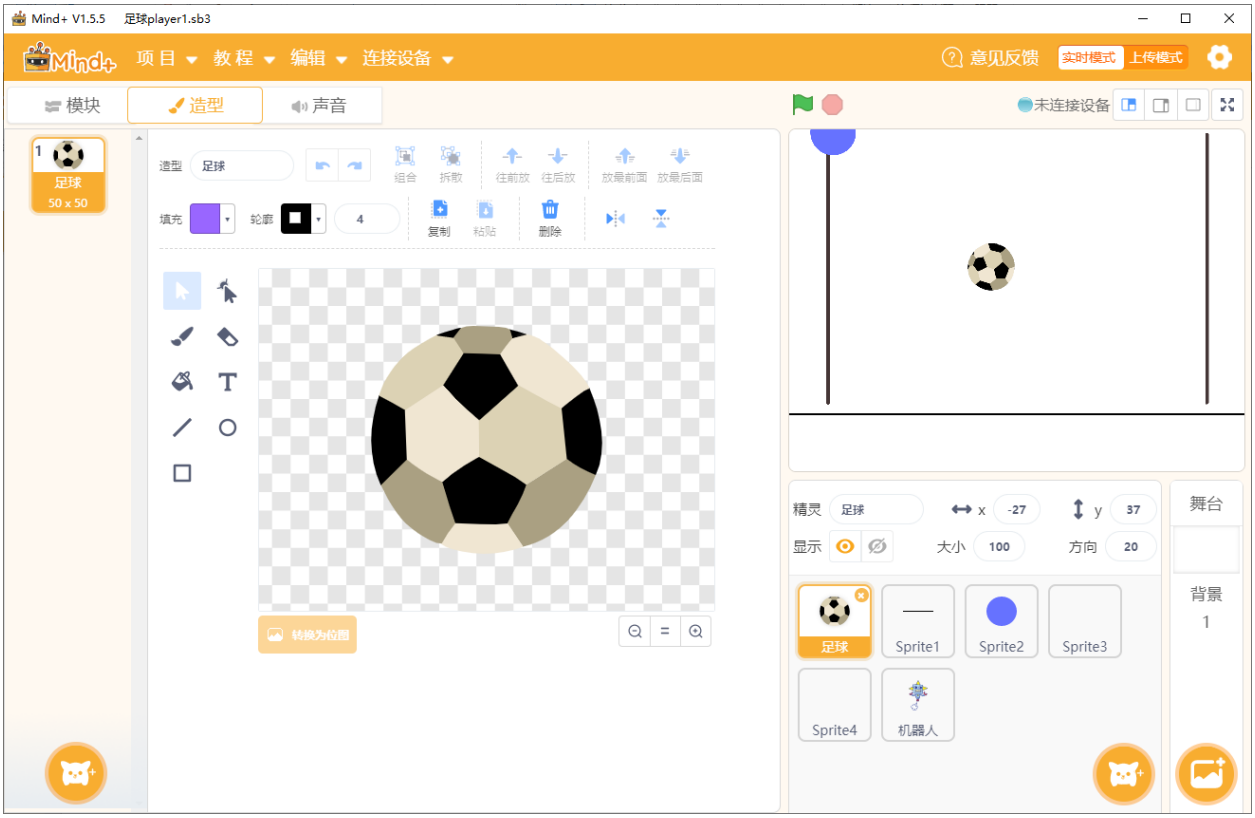
4.6.3 准备工作

1、下载并安装 Mind+1.5.5 及以上的版本

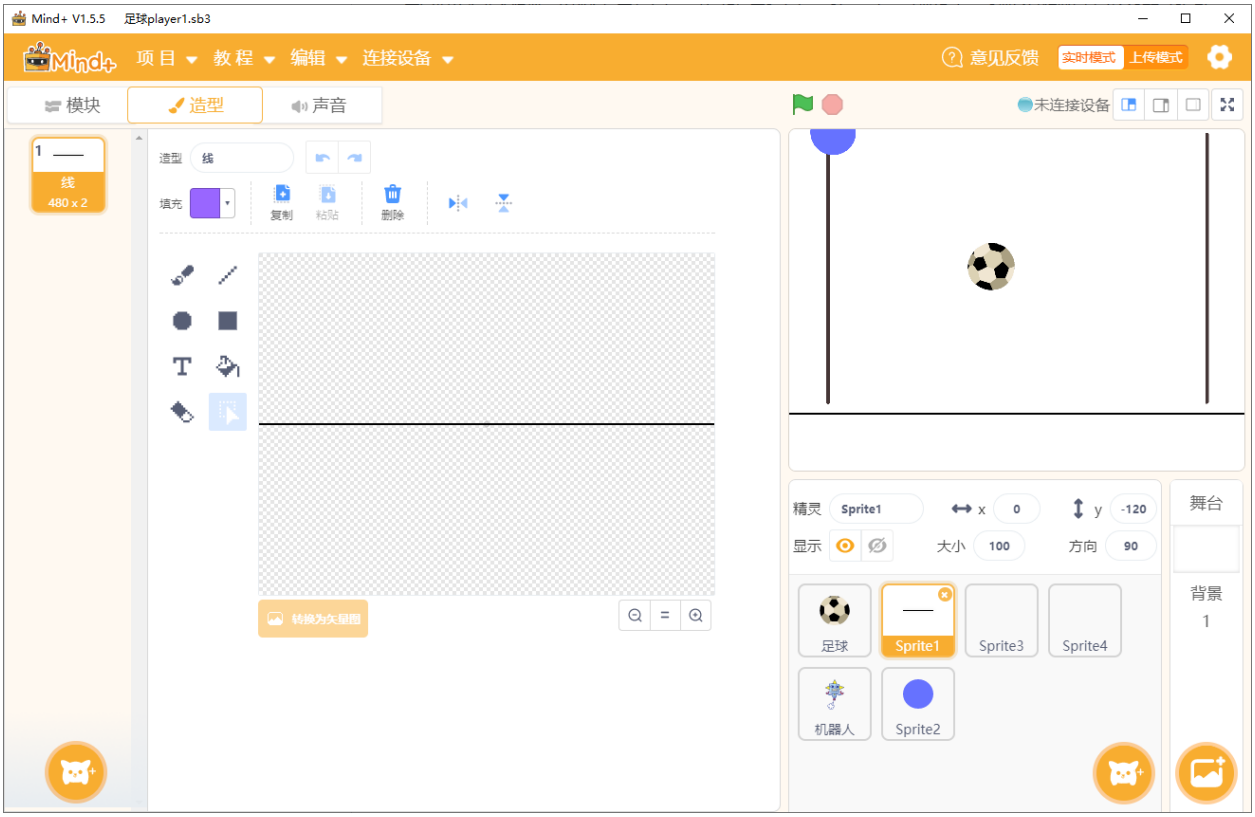
2、注册一个物联网（easySIoT）平台的账号，本案例以 DFrobot 公司搭建的 easySIoT 物联网平台（地址：http://iot.dfrobot.com.cn）为例。

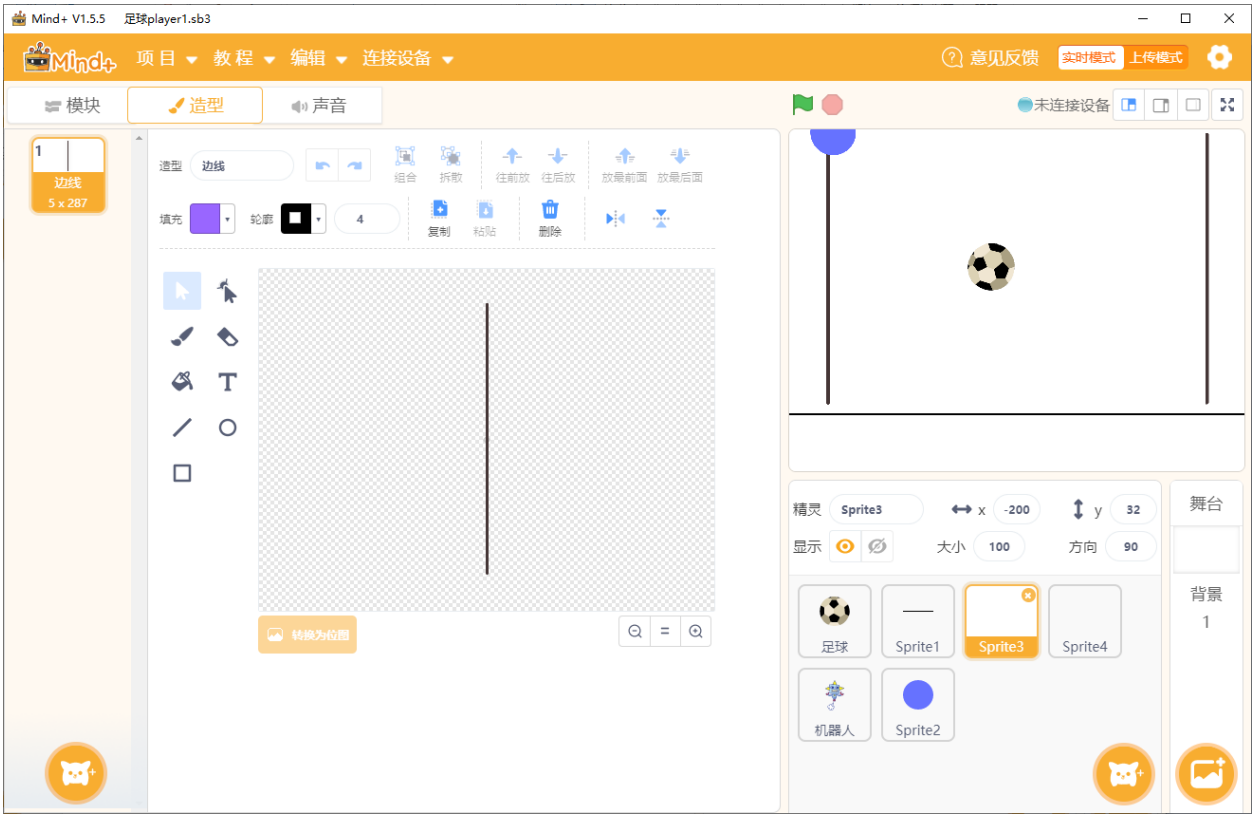
4.6.4 步骤

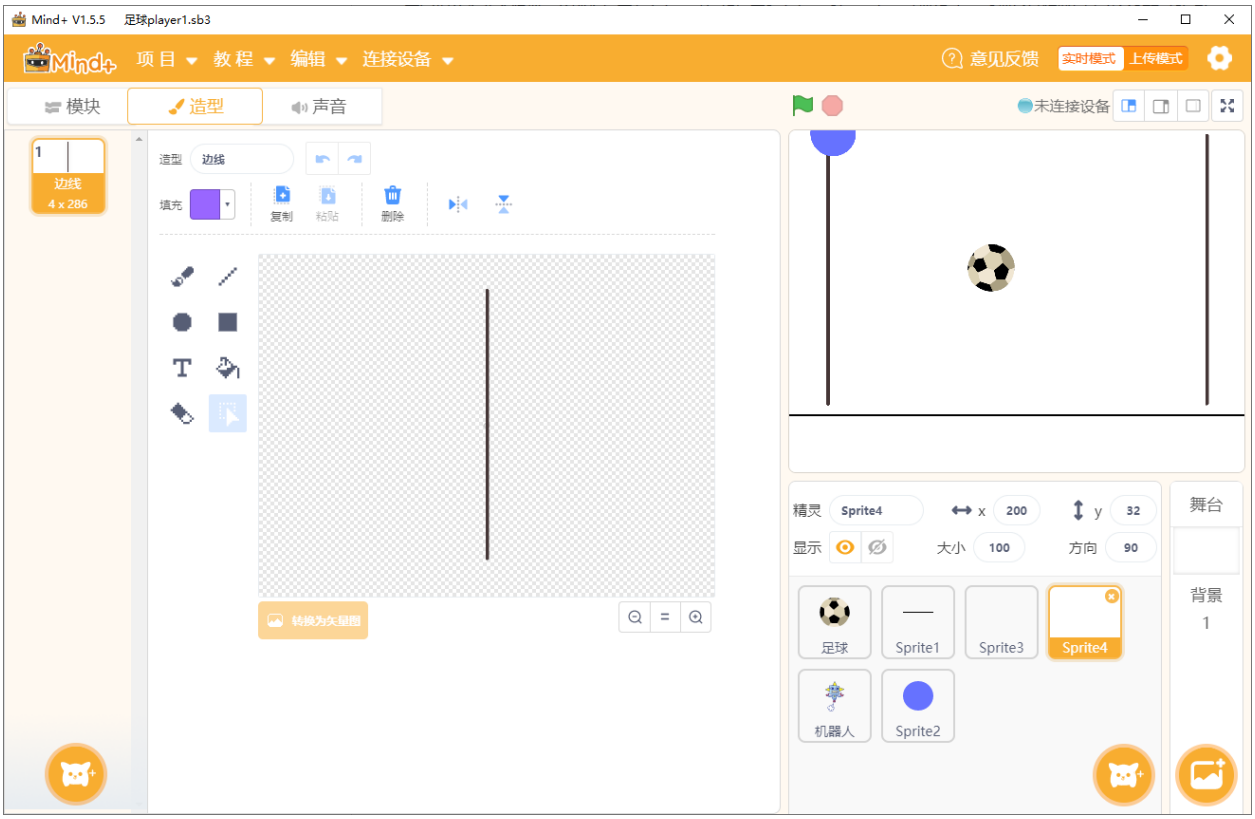
1、绘制足球


Mind+ V1.5.5 足球player1.sb3
项目
教程
编辑
连接设备
意见反馈
实时模式
上传模式
模块
造型
声音
未连接设备
1
足球
50 x 50
造型 足球
组合
拆散
往前放
往后放
放最前面
放最后面
填充
轮廓
4
复制
粘贴
删除
转换为位图
精灵 足球
x -27
y 37
显示
大小 100
方向 20
舞台
背景
1
足球
Sprite1
Sprite2
Sprite3
Sprite4
机器人

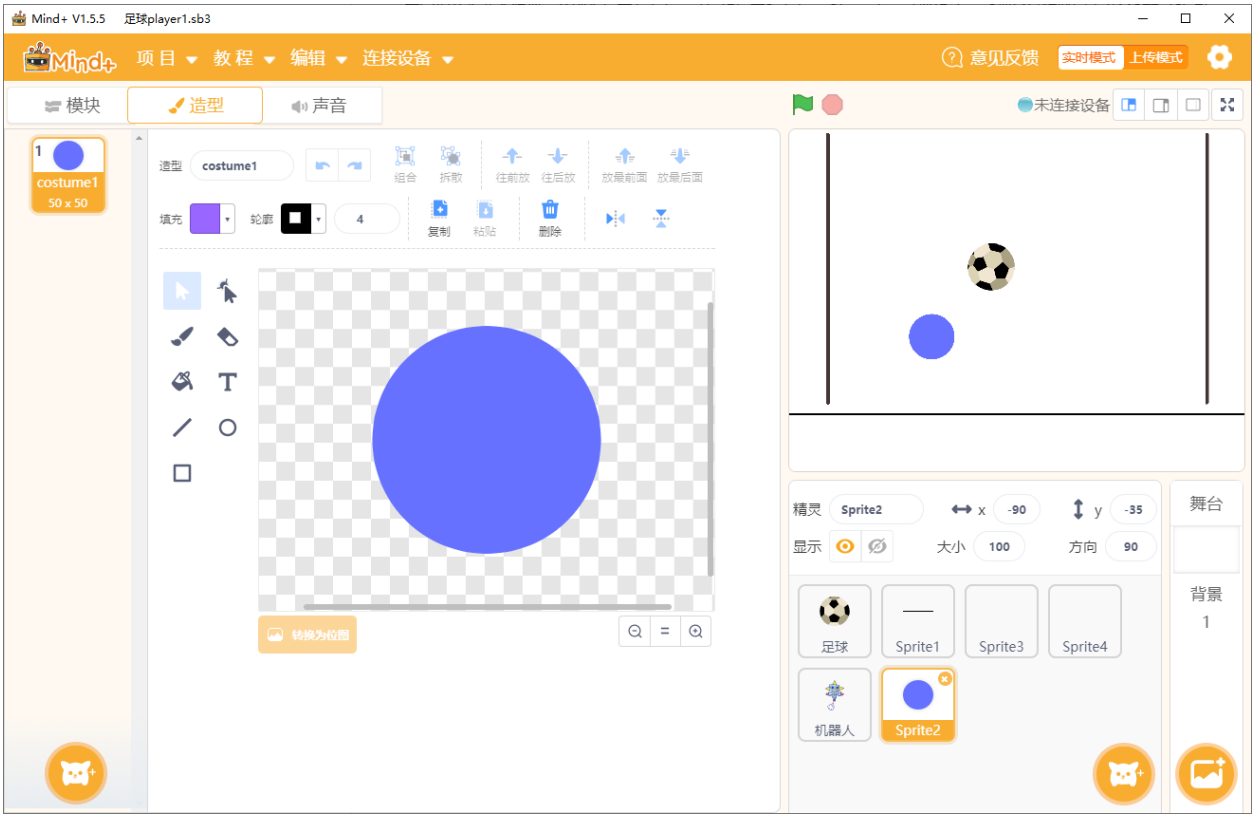
2、绘制边界线


Mind+ V1.5.5 足球player1.sb3
项目
教程
编辑
连接设备
意见反馈
实时模式
上传模式
模块
造型
声音
未连接设备
1
线
480 x 2
造型 线
填充
复制
粘贴
删除
转换为矢量图
精灵 Sprite1
x 0
y -120
显示
大小 100
方向 90
舞台
背景
1
足球
Sprite1
Sprite3
Sprite4
机器人
Sprite2

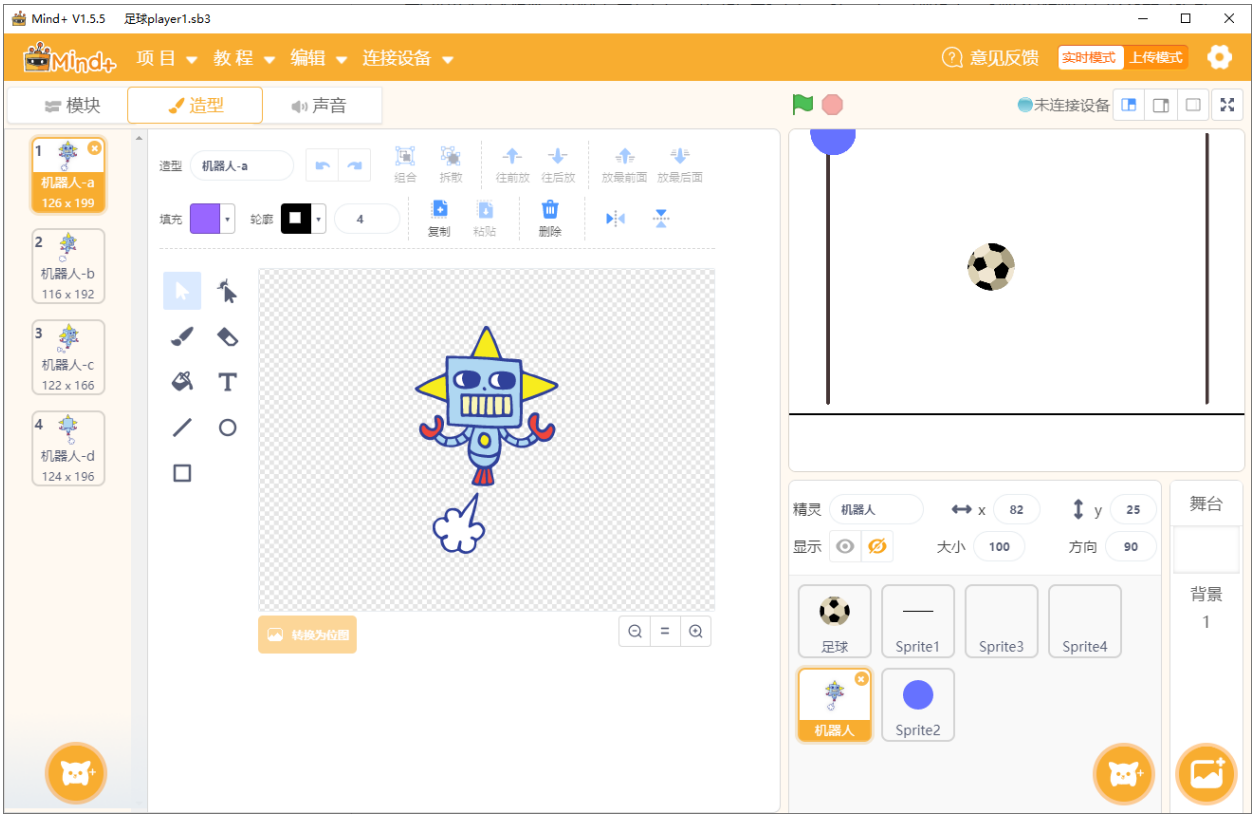

Mind+ V1.5.5 足球player1.sb3
项目 教程 编辑 连接设备
意见反馈
实时模式 上传模式
模块 造型 声音
未连接设备
造型 边线
填充 轮廓 4
复制 粘贴 删除
转换为位图
精灵 Sprite3 x -200 y 32
显示 大小 100 方向 90
足球 Sprite1 Sprite3 Sprite4
机器人 Sprite2
舞台
背景 1


Mind+ V1.5.5 足球player1.sb3
项目 教程 编辑 连接设备
意见反馈
实时模式 上传模式
模块 造型 声音
未连接设备
造型 边线
填充
复制 粘贴 删除
转换为矢量图
精灵 Sprite4 x 200 y 32
显示 大小 100 方向 90
足球 Sprite1 Sprite3 Sprite4
机器人 Sprite2
舞台
背景 1

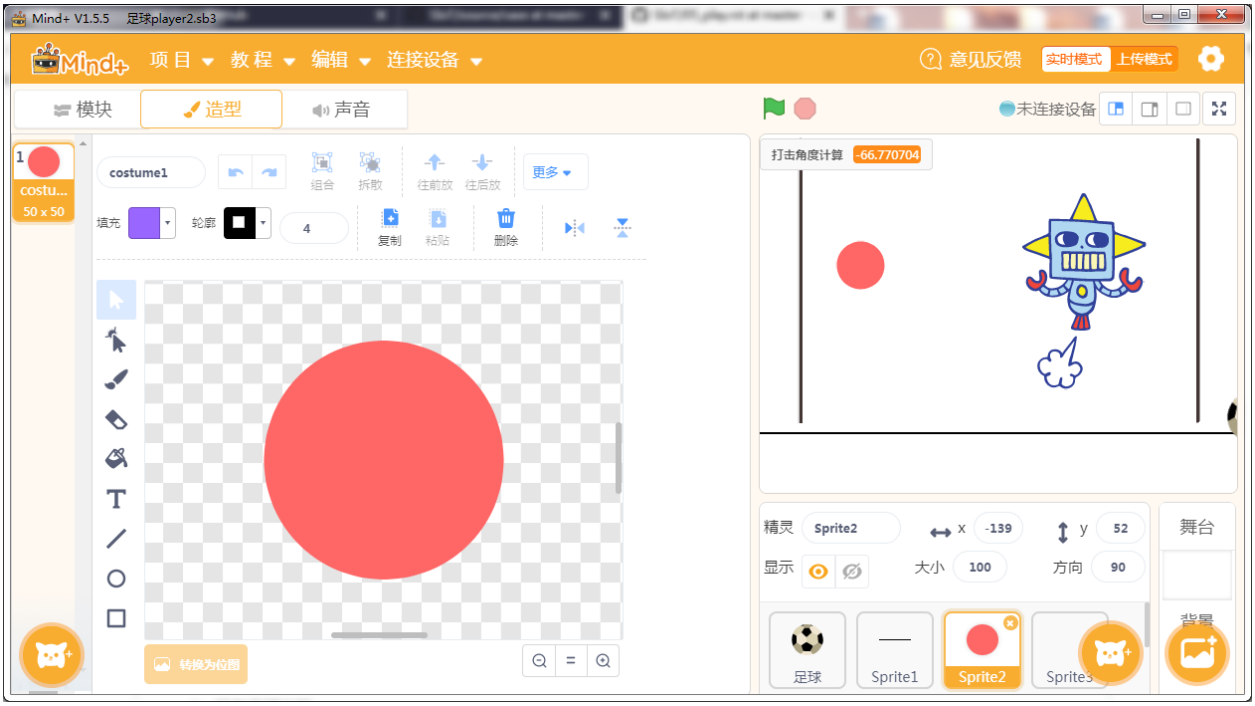
3、绘制击球光标


Mind+ V1.5.5 足球player1.sb3
项目
教程
编辑
连接设备
意见反馈
实时模式
上传模式
模块
造型
声音
未连接设备
costume1
50 x 50
造型 costume1
组合
拆散
往前放
往后放
放最前面
放最后面
填充
轮廓
4
复制
粘贴
删除
转换为位图
精灵 Sprite2
x -90
y -35
显示
大小 100
方向 90
舞台
背景 1
足球
Sprite1
Sprite3
Sprite4
机器人
Sprite2

4、结束角色


Mind+ V1.5.5 足球player1.sb3
项目
教程
编辑
连接设备
意见反馈
实时模式
上传模式
模块
造型
声音
未连接设备
1 机器人-a 126 x 199
2 机器人-b 116 x 192
3 机器人-c 122 x 166
4 机器人-d 124 x 196
造型 机器人-a
组合
拆散
往前放
往后放
放最前面
放最后面
填充
轮廓
4
复制
粘贴
删除
转换为位图
精灵 机器人
x 82
y 25
显示
大小 100
方向 90
舞台
背景 1
足球
Sprite1
Sprite3
Sprite4
机器人
Sprite2

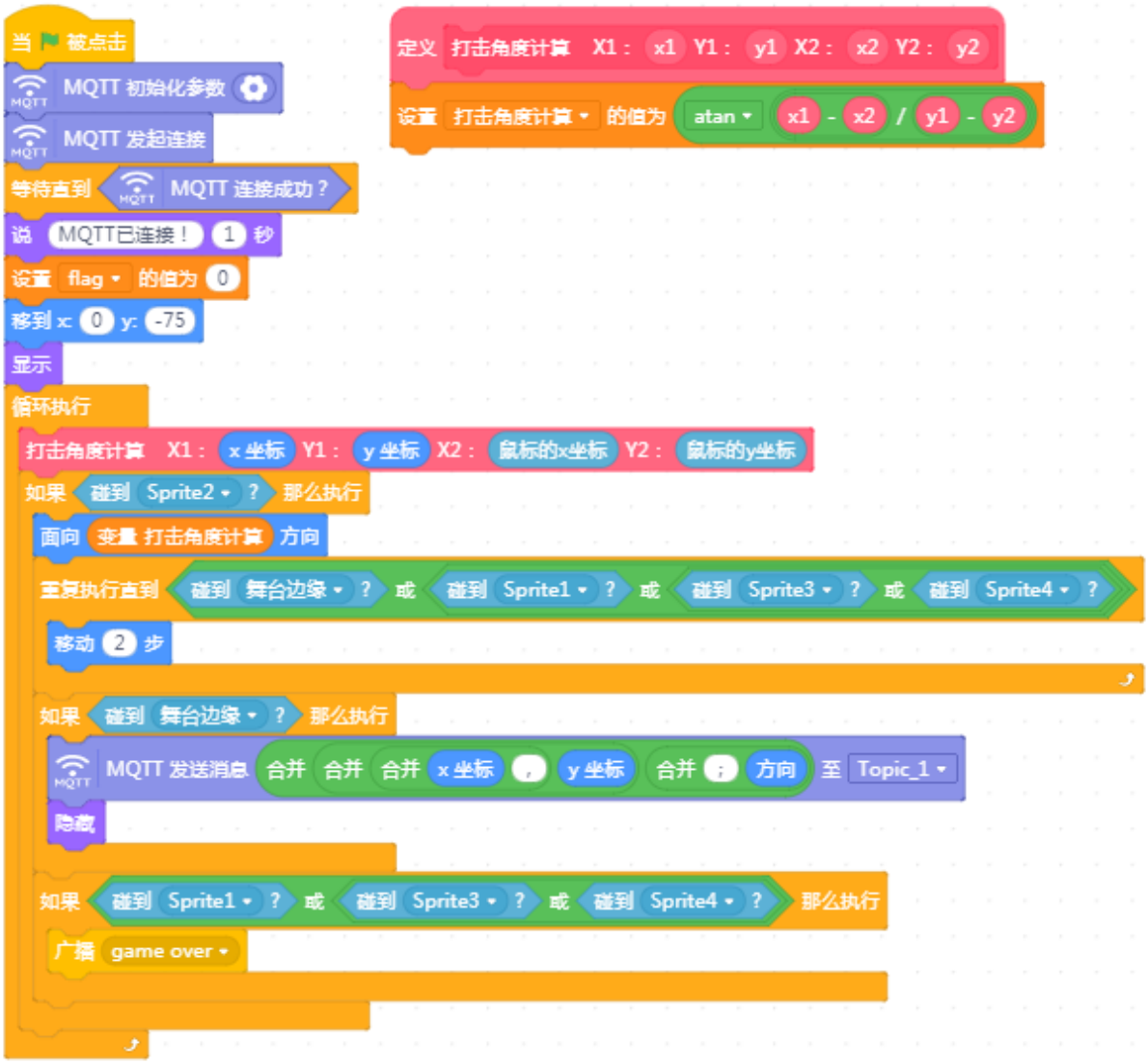
5、player2 端与 player1 端的角色基本相同，就光标的颜色须调成其他颜色，如：红色。

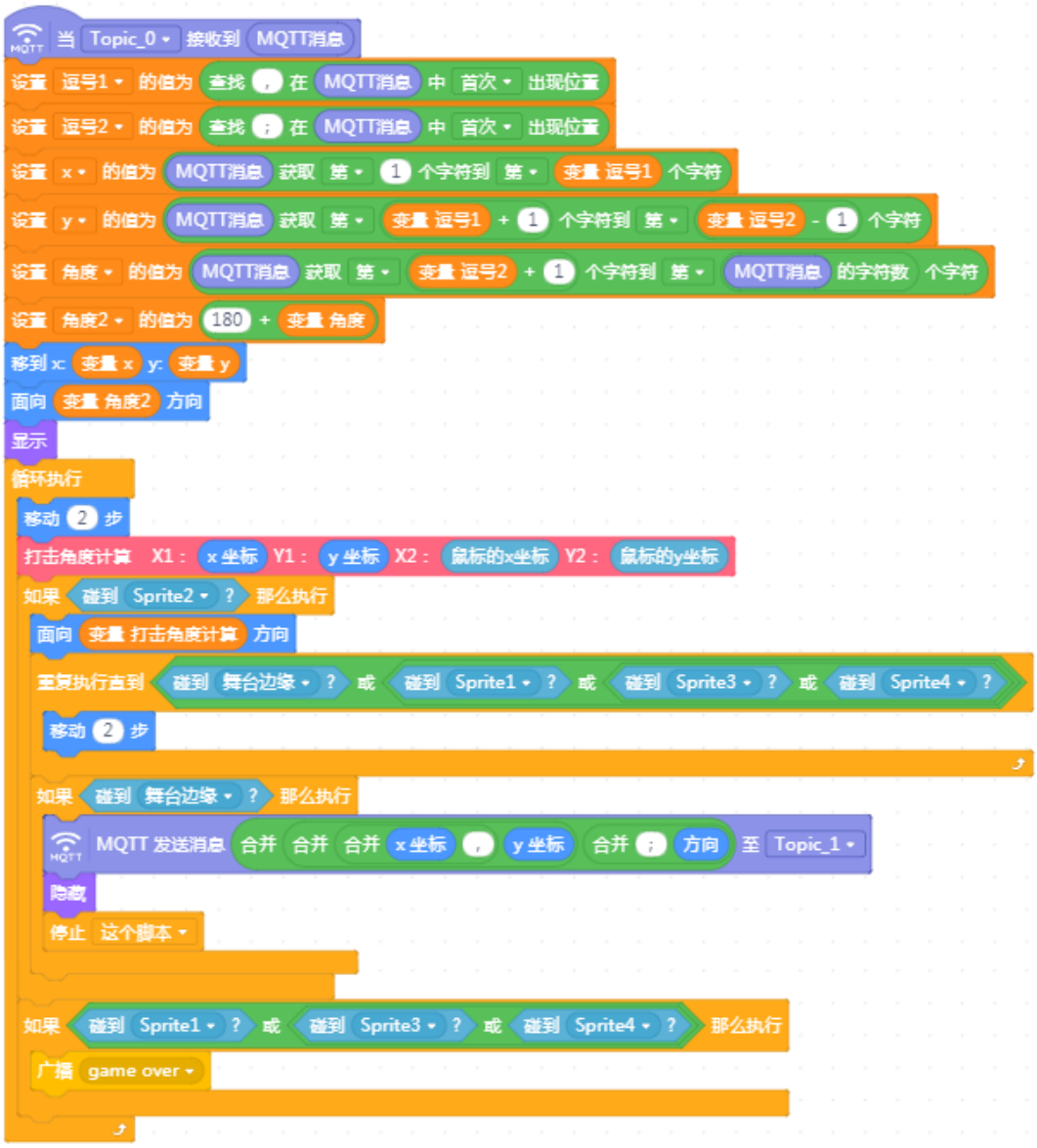


4.6.5 参考代码

player1 端

1、足球角色程序代码


当 被点击
MQTT 初始化参数
MQTT 发起连接
等待直到 MQTT 连接成功？
说 MQTT已连接！ 1 秒
设置 flag 的值为 0
移到 x: 0 y: -75
显示
循环执行
打击角度计算 X1： x 坐标 Y1： y 坐标 X2： 鼠标的x坐标 Y2： 鼠标的y坐标
如果 碰到 Sprite2 ？ 那么执行
面向 变量 打击角度计算 方向
重复执行直到 碰到 舞台边缘 ？ 或 碰到 Sprite1 ？ 或 碰到 Sprite3 ？ 或 碰到 Sprite4 ？
移动 2 步
如果 碰到 舞台边缘 ？ 那么执行
MQTT 发送消息 合并 合并 合并 x 坐标 , y 坐标 合并 ; 方向 至 Topic_1
隐藏
如果 碰到 Sprite1 ？ 或 碰到 Sprite3 ？ 或 碰到 Sprite4 ？ 那么执行
广播 game over
定义 打击角度计算 X1： x1 Y1： y1 X2： x2 Y2： y2
设置 打击角度计算 的值为 atan x1 - x2 / y1 - y2



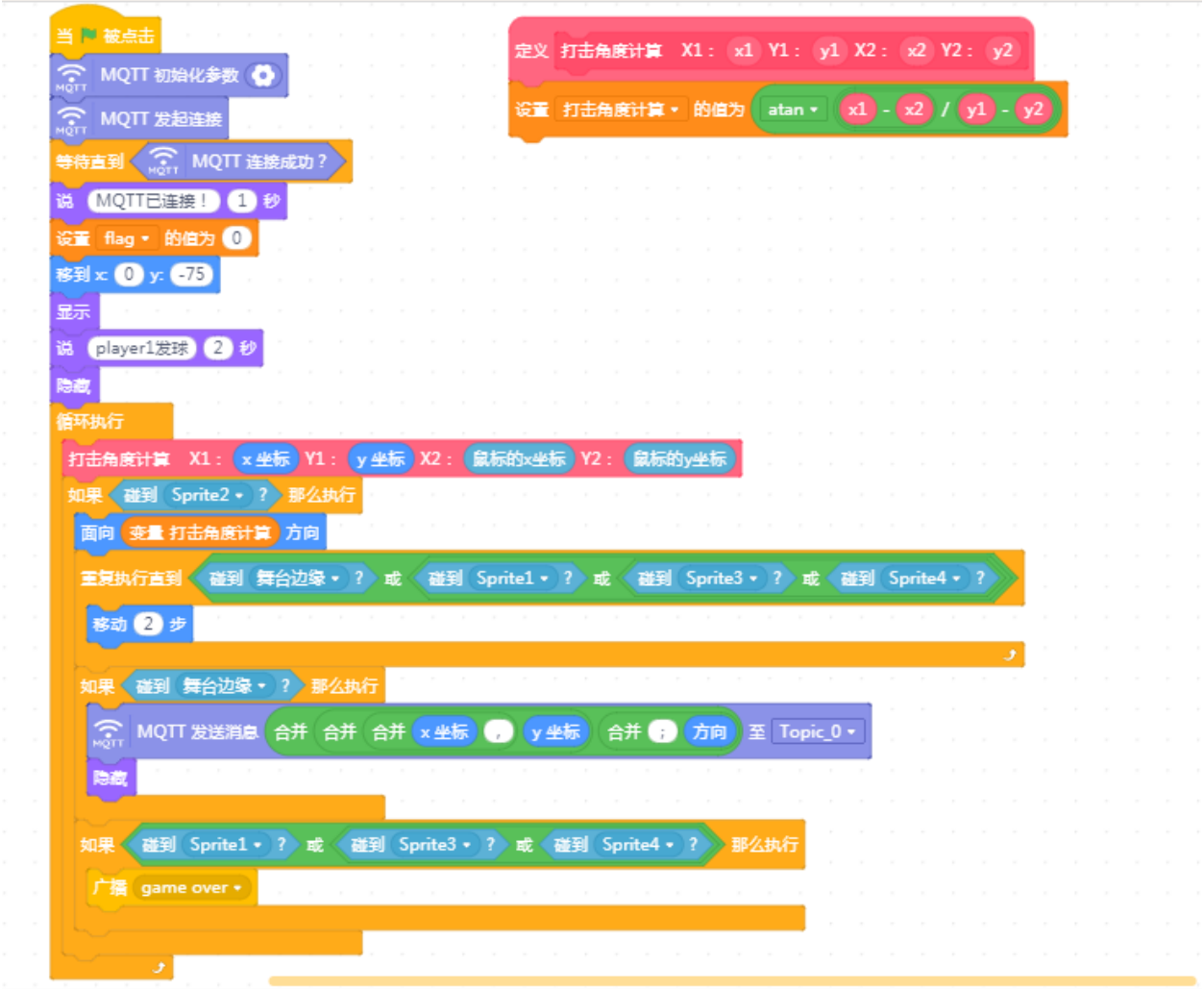
2、击球光标角色程序代码

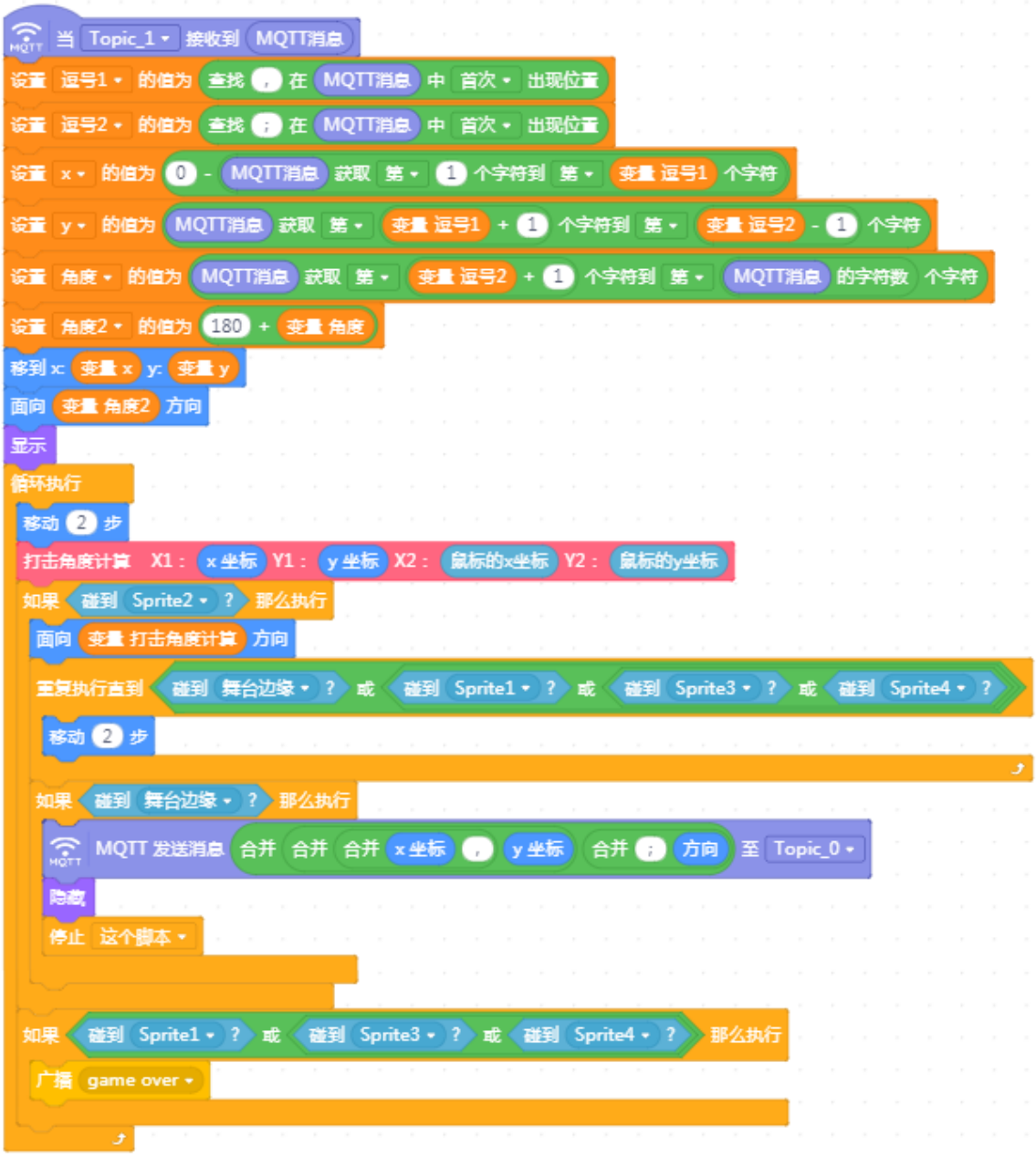
3、结束角色程序代码

player2 端

player2 端的程序代码与 player1 端的基本相同，就足球角色程序代码有点不一样。

1、足球角色程序代码





参考代码下载：

https://github.com/vvlink/SIoT/tree/master/examples/Mind%2B

4.7 互动媒体之智慧农场

光照是地球上生物赖以生存与繁衍的基础，作物的光合作用离不开光照，光照条件的好坏直接影响作物的产量和品质，现如今人工补光已经成为高效生产的重要手段，让农作物在光线不足的时候也能拥有需要的光照。

本项目我们将利用掌控板上的光线传感器和 LED 灯模拟实现智能农场中的补光功能，并结合 SIoT 和 Mind+

中的舞台模拟智能农场中的远程监控功能，让我们足不出户也可以在电脑上实时看见农场中的光线变化。针对此项目我们可以划分为两种应用场景，具体如下：

(1) 应用场景一：农场。利用掌控板实时采集光线值，并通过控制 LED 灯的亮灭实时对农作物补光。为了能够与远程端共享数据，可将采集到的数据上传至 SIoT 进行存储。

(2) 应用场景二：远程端。此时只需要一台电脑，不需要连接其他硬件。通过 mind+ 软件实时模式下 SIoT 控制获得农场中掌控板上传的数据，配合舞台设计，实现光线值的显示以及昼夜交替的模拟场景效果。

我们将通过两个应用场景进行实践解决智能农场补光的项目设计。

4.7.1 准备工作

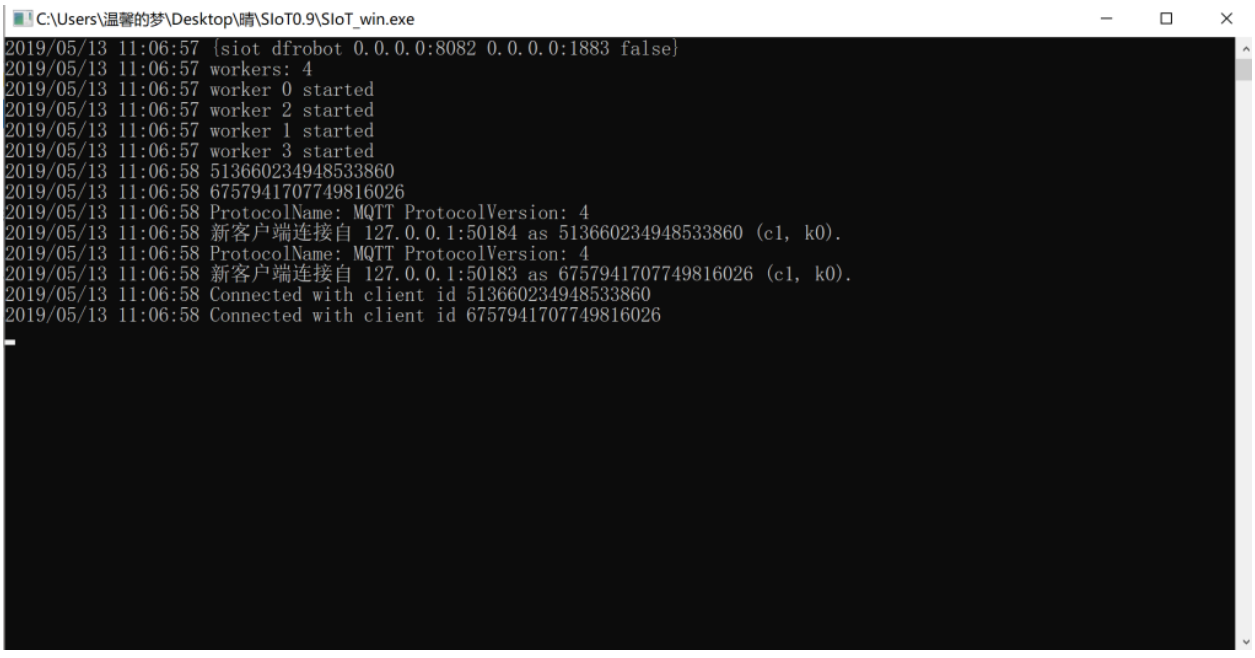
1. 硬件准备

掌控板及其连接线

2. 软件准备

(1) 搭建 SIoT 服务器

直接双击点击与系统匹配的 SIoT 运行文件，屏幕会弹出一个黑色的 CMD 窗口，在配置中请不要关闭它。



(2) 登录 SIoT 平台

打开浏览器，输入 url：http://localhost:8082。

4.7.2 步骤

一、应用场景一：农场（掌控板采集光线）

此时掌控板需要脱离电脑使用 SIoT，所以应使用 Mind+ 的上传模式。

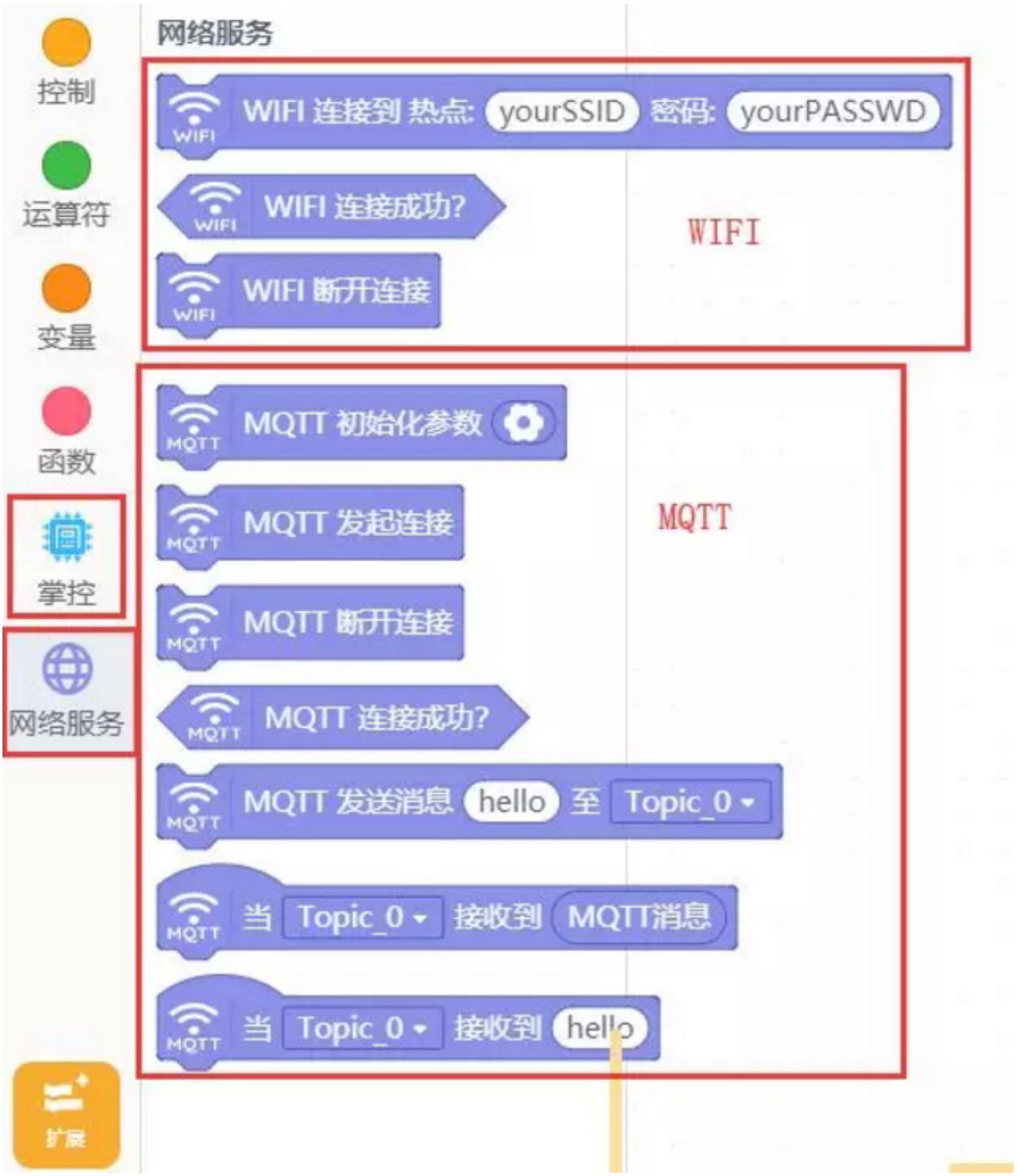
1.Mind+ 软件设置

打开 Mind+ 软件（1.5.5 及以上版本）：

（1）选择“上传模式”；

（2）点击“扩展”，在“主控板”下，点击选择“掌控板”；

（3）点击“扩展”，在“网络服务”下，点击选择“MQTT”和“WIFI”后点击“返回”。出现下图标记的内容证明选择成功！


网络服务
控制
运算符
变量
函数
掌控
网络服务
扩展
WIFI 连接到 热点: yourSSID 密码: yourPASSWD
WIFI 连接成功?
WIFI 断开连接
WIFI
MQTT 初始化参数
MQTT 发起连接
MQTT 断开连接
MQTT 连接成功?
MQTT 发送消息 hello 至 Topic_0
当 Topic_0 接收到 MQTT消息
当 Topic_0 接收到 hello
MQTT

2. 编写程序

(1) 参考程序:


ESP32 主程序
屏幕显示文字 掌控板 在坐标 X: 42 Y: 22 预览
WIFI 连接到 热点: DFRobot-guest 密码: dfrobot@2017
等待直到 WIFI 连接成功?
屏幕显示为 全黑
屏幕显示文字 WIFI 在坐标 X: 42 Y: 22 预览
MQTT 初始化参数
MQTT 发起连接
等待直到 MQTT 连接成功?
屏幕显示为 全黑
屏幕显示文字 MQTT 在坐标 X: 42 Y: 22 预览
MQTT 发送消息 hello 至 Topic_0
循环执行
SIoT详情请点击图片下方链接
物联网平台: SIOT
参数:
服务器地址: 192.168.9.191
账号: siot
密码: dfrobot
Topic_0: xx/aa

（2）点击上传到设备将程序下载到掌控板中。

3. 功能实现

上传成功后可在 SIoT 网页端查看消息以及实时光线折线图，如下图：

二、应用场景二：远程端（mind+ 实时模式）

为了能在远程端更直接的显示光线效果，在 Mind+ 实时模式下可以对舞台进行设计模拟光线变化时昼夜交替的效果以及绘制光线值折线图。

1.Mind+ 软件设置

打开 Mind+ 软件（1.5.5 及以上版本）：

1、选择“实时模式”；

2、选择“功能模块”，点击选择“画笔”；

3、选择“网络服务”，点击选择“MQTT”后点击“返回”。

画笔在本项目中主要用来进行折线图的绘制，“MQTT”主要是将实时模式与 SIoT 进行通信实现读取掌控板中的光线值。

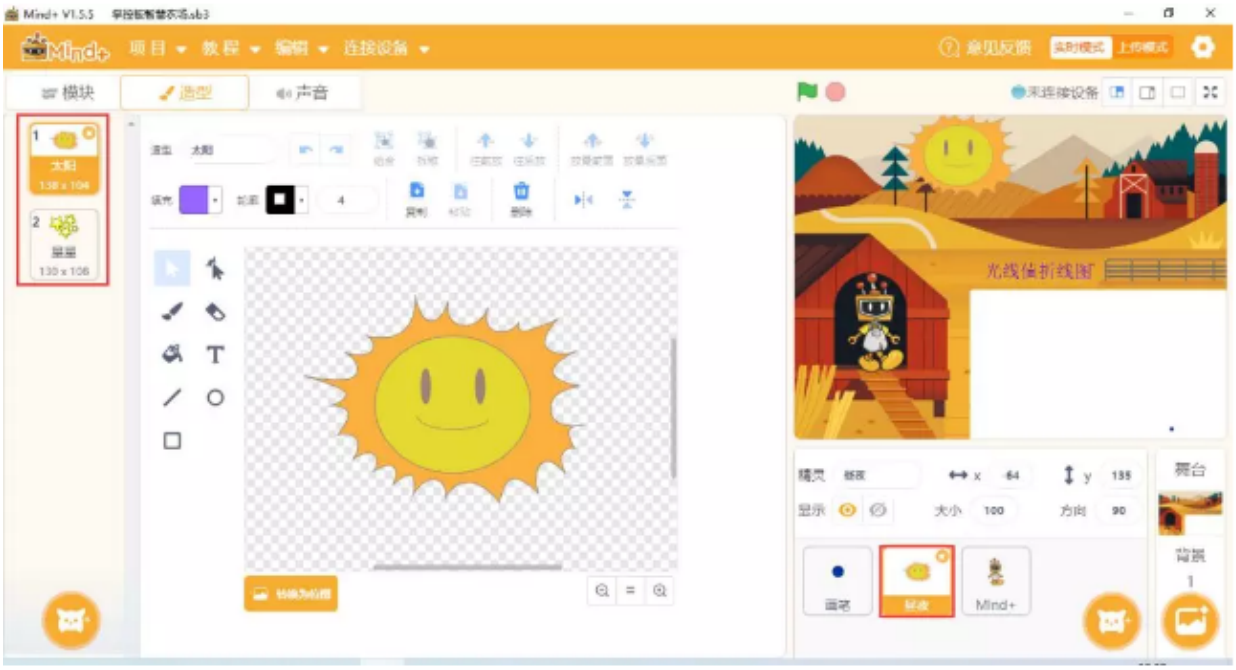
2. 编写程序

（1）参考程序：


当 被点击
抬笔
移到 x: -20 y: -170
全部擦除
设置 X轴 的值为 0
广播 开始连接...
MQTT
MQTT 初始化参数
需要与上传模式中的
参数设置保持一致
MQTT
MQTT 发起连接
等待直到
MQTT
MQTT 连接成功?
广播 连接成功...

下面的程序需要在实时模式舞台中新建角色才能实现更换造型的效果（新建角色成功如下图），更换造型太阳和星星来区分昼夜。在此项目因为在常规环境中，光线传感器的返回值一般为 0，在本项目中设置的分界点为 200，在项目实施中可通过手机手电筒照射在传感器上感受数值的变化并观察在舞台中的效果。

为了更清楚地展现昼夜交替的效果，新建的角色为太阳和星星，如下图：



对应程序如下：

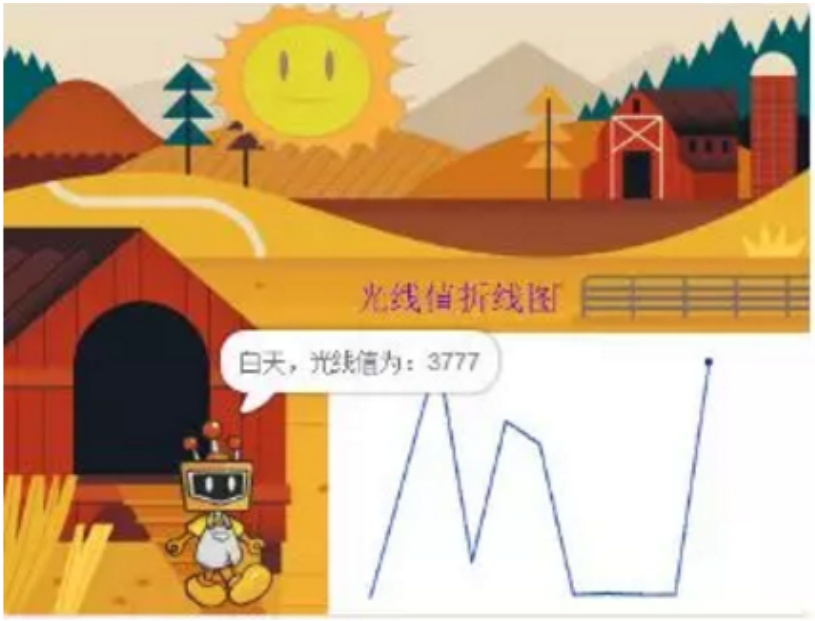
为了更明显的感受到昼夜的变化，mind+ 机器人将会在晚上和白天出现在不同的位置并讲出当前环境以及光线值，具体程序如下图：

（2）点击“绿旗标志”运行程序

程序和舞台设计结束后点击“绿旗标志”运行，则可在 mind+ 实时模式舞台中看见光线值的折线图以及是白天还是傍晚。

3. 功能实现

当光线值大于 200，舞台将切换为白天，掌控板的显示屏将会显示“白天，光线充足”效果如下图：



当光线值小于 200，舞台将切换为黑夜，掌控板上的 LED 灯将被点亮进行补光，显示屏将会显示“晚上，需要补光”，效果如下图：

4.7.3 参考代码

https://github.com/vvlink/SIoT/blob/master/examples/Mind+/智慧农场.sb3

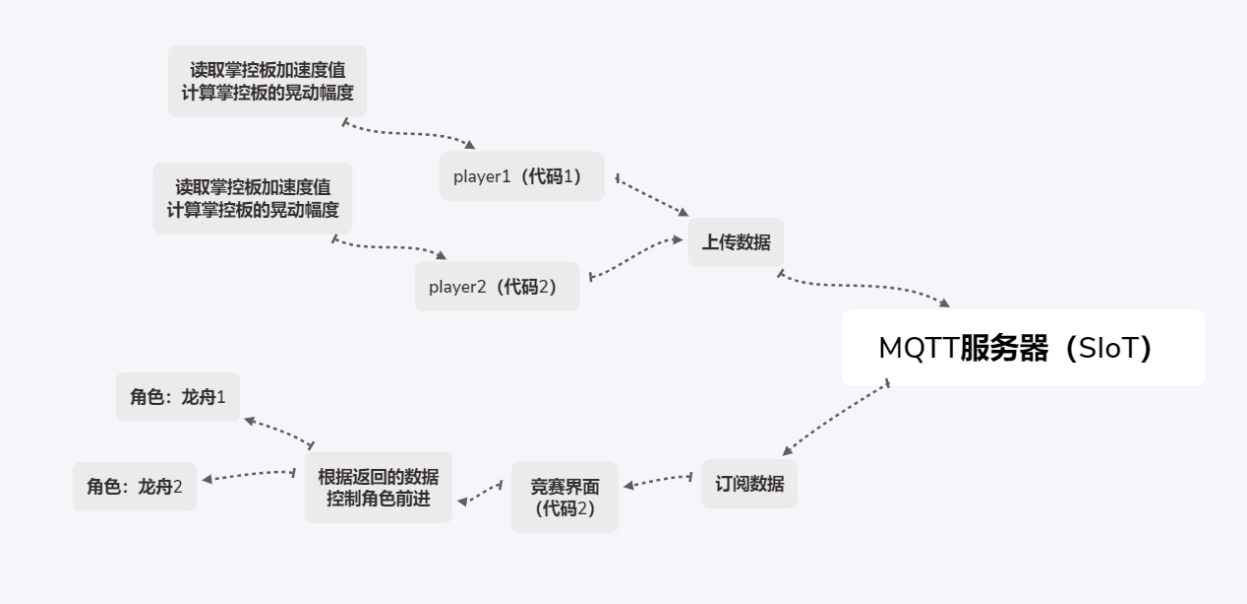
4.8 互动媒体之龙舟竞赛

基于 SIoT 和和掌控板，可以设计一些多人竞赛的游戏，然后借助 Mind+ 实时呈现出来。

案例作者：张喻，林淼焱

4.8.1 程序描述：

1. 有两个玩家：玩家 1 和玩家 2，每个玩家拥有一块掌控版用以摇晃
2. 游戏开始，摇晃掌控版，让舞台上的龙舟开始移动
3. 可以添加多支龙舟进行比拼

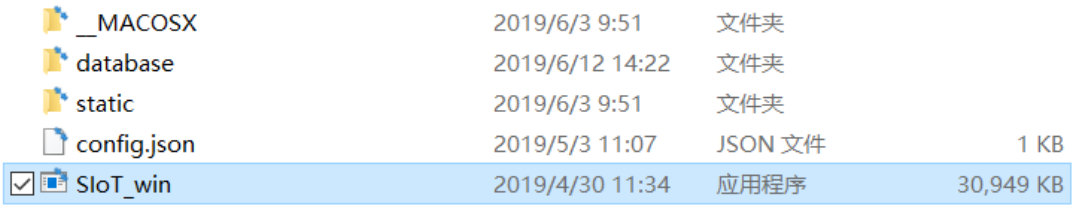


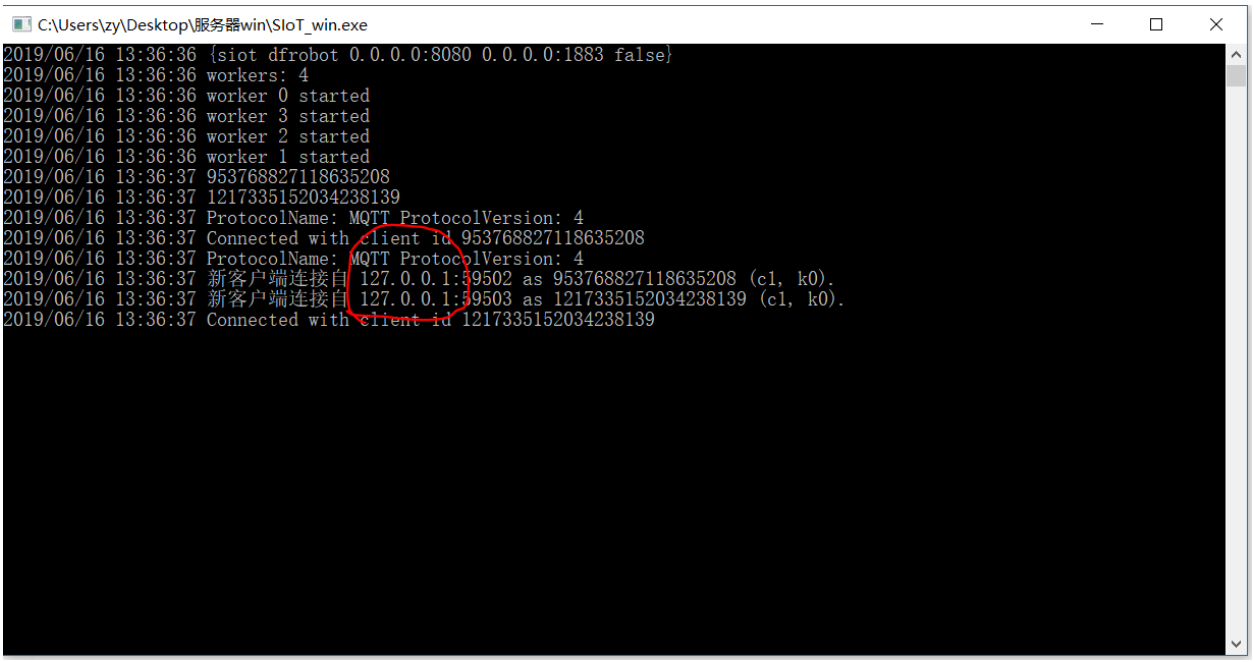
4.8.2 原理介绍

本案例分为两个终端，分别为划手 1（player1）和划手 2（player2）。player1 端和 player2 端通过连接同一物联网平台 MQTT（siot）进行数据的交换，从而实现联机比拼的功能。

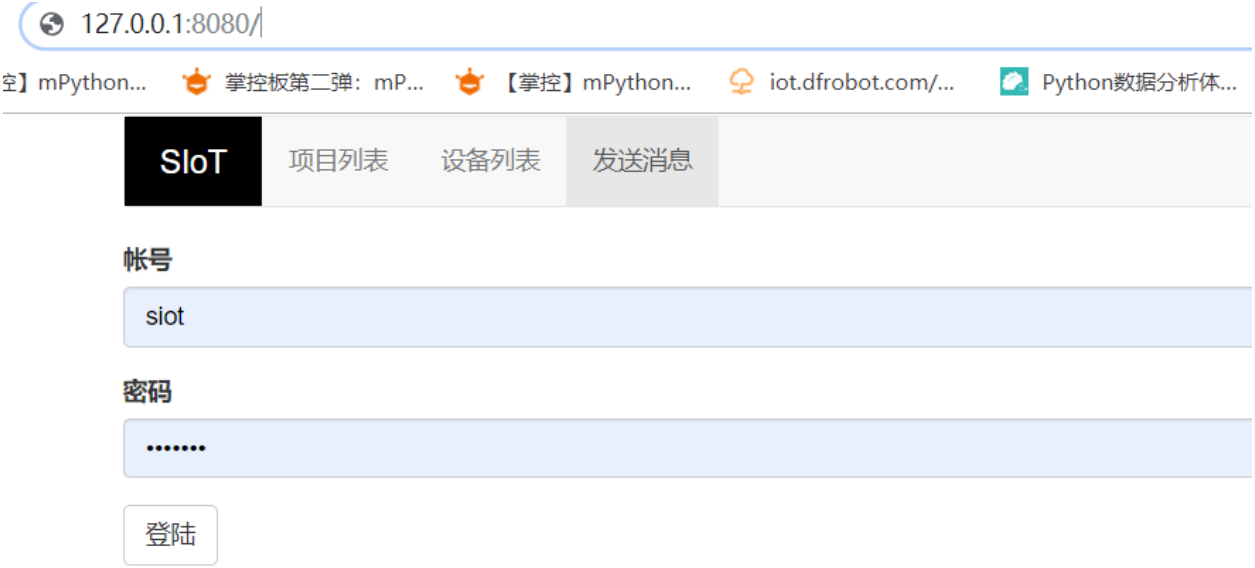
4.8.3 准备工作

1. 运行 SIoT





2. 将上图中的 IP 地址输入网页，如下图：



3. 运行 Mind+1.5.5 及以上的版本

mind+ 下载地址：http://mindplus.cc

说明：本教程使用的是 Mind+1.5.5 版本软件

4.8.4 操作步骤

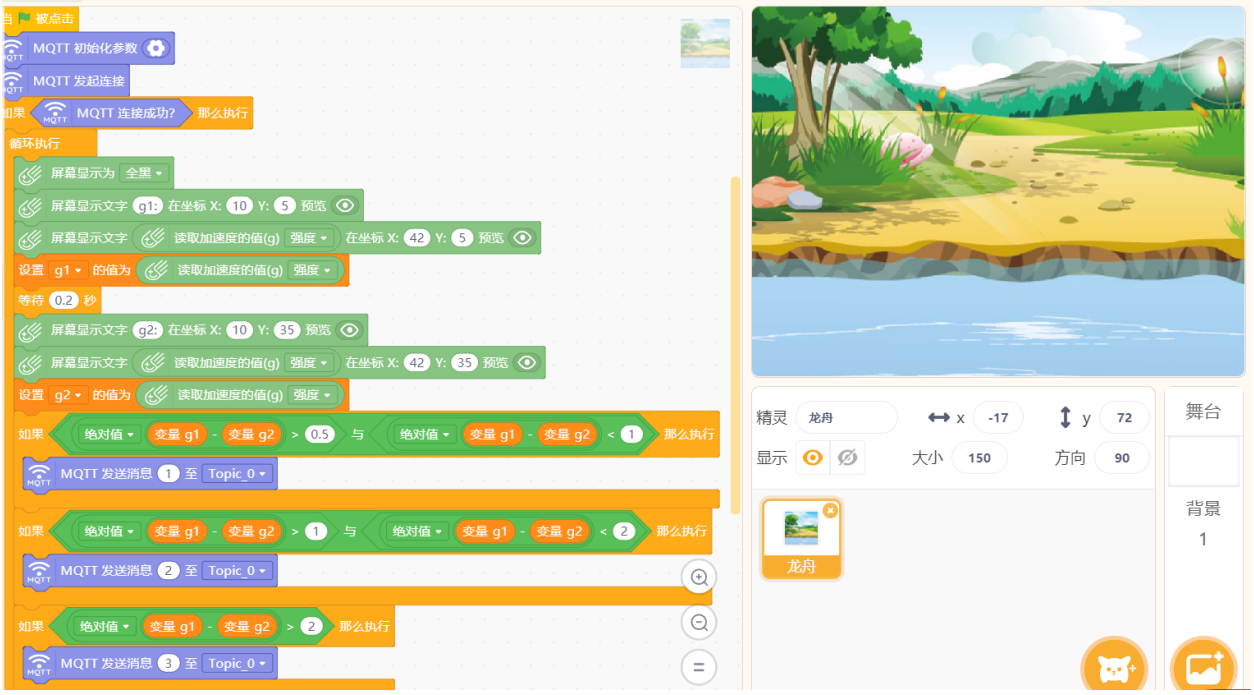
1. 运行 mind+

选择右上角“实时模式”，点击左下角“扩展”，添加”网络服务”中的“MQTT”；

2. 编写代码

1）在终端 1（代码 1）的“背景”中写如下程序；

说明：本操作是将掌控板设备与服务器建立连接，在服务器上能够实时反馈数据。



并修改 MQTT 服务器相关的参数；

说明：Topic 设置为“xzr/001”（项目 ID/名称）

2）在终端 2（代码 2）中创建两个角色，分别为龙舟 1 和龙舟 2；

3）点击角色 1（龙舟 1），写如下代码；

4）点击角色 2（龙舟 2），写如下代码；

4.8.5 参考代码

代码下载地址：https://github.com/vvlink/SIoT/tree/master/examples/Mind%2B

演示视频地址：https://github.com/linmiaoyan/Lins-video/blob/master/Mind%2B%E9%BE%99%E8%88%9F%E7%AB%9E%E8%B5%9B.mp4

4.8.6 拓展思考

利用这一作品原理，可以制作一些集体互动的大型游戏。

4.9 互动媒体之弹幕效果演示

弹幕，是现在最常见的一种视频交互应用了。用 Mind+ 也可以做一个弹幕应用。

**案例作者：谢作如，林淼焱 **

4.9.1 案例描述：

1. 当收到 MQTT 的消息，一个小动物会带着对话气泡（即说话），从屏幕的右边跑到左边，然后消失。

2. 带着对话气泡出来小动物有多个形象，如小猫、兔子、小狗等。

3. 主程序用 Mind+ 制作，发送弹幕消息的客户端用各种工具，可以是 Python、手机 App、MQTT 客户端、Web 管理端，当然，也可以是 Mind+。

4.9.2 准备工作

1. 运行 SIoT。

2. 安装 Mind+。

4.9.3 操作步骤

1. 打开 Mind+，添加”网络服务”中的“MQTT”。

2. 编写代码。

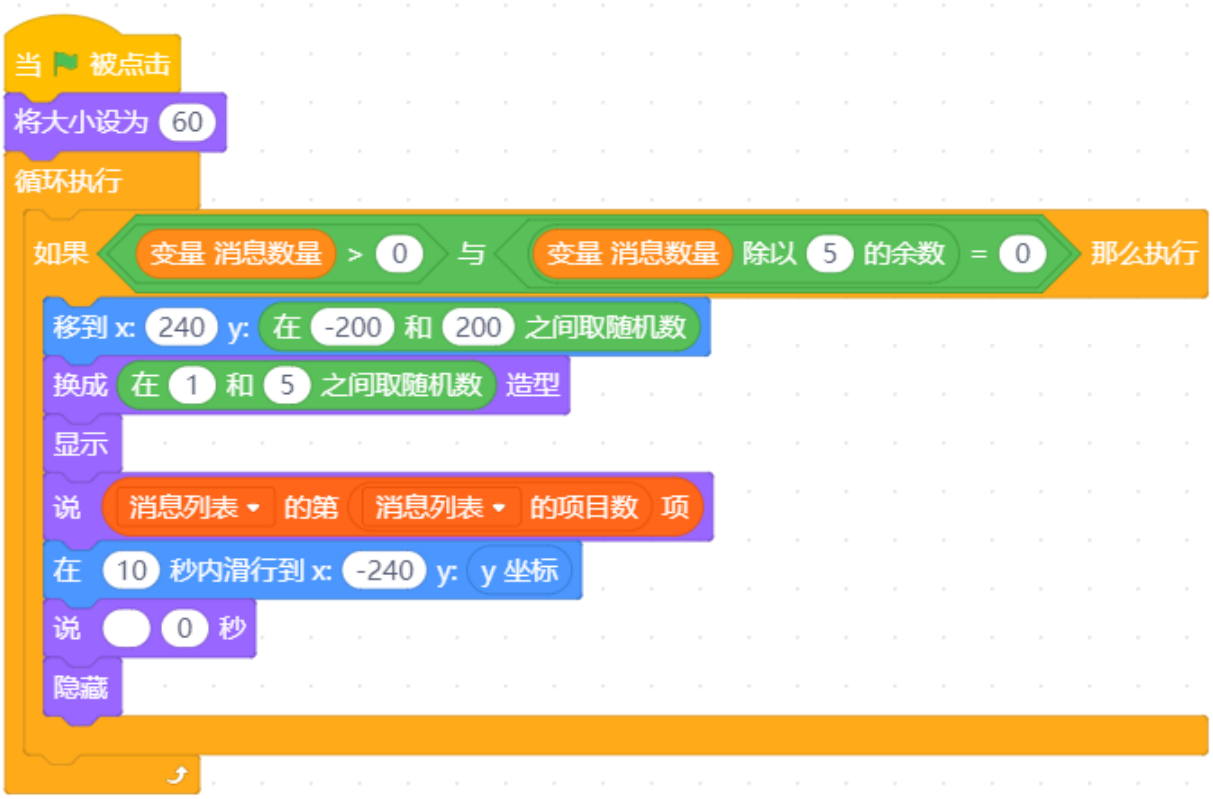
1）在“背景”中写如下程序。

并修改 MQTT 服务器相关的参数

说明：Topic 设置为“xzr/001”（项目 ID/名称）

2）给角色 1 选择多个造型。

3）在角色 1 中写如下代码。



4）复制出 4 个角色，修改代码。

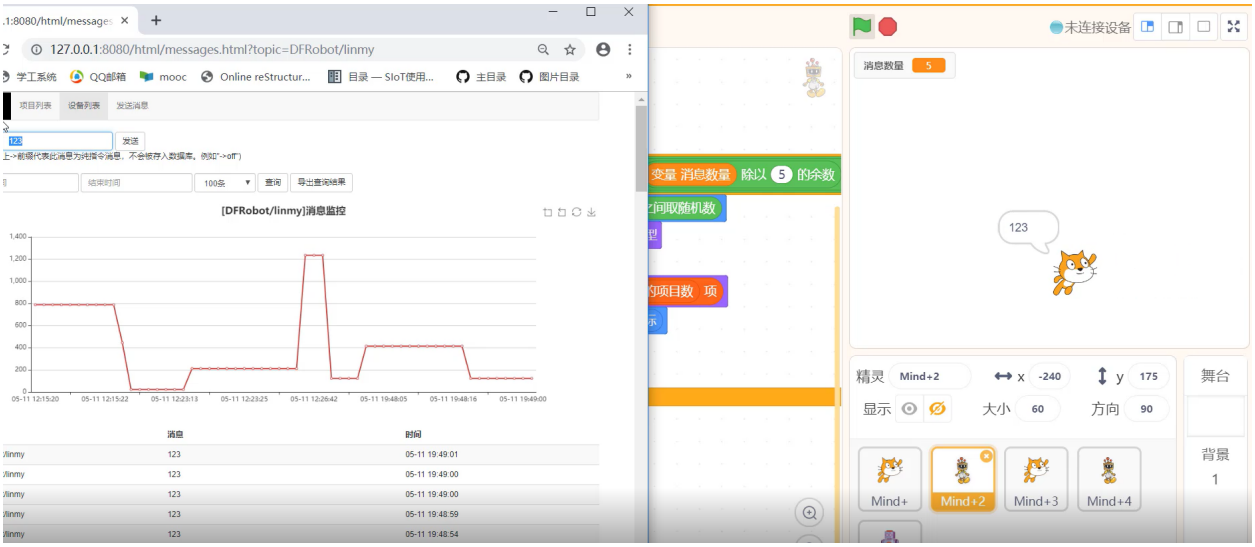
5）如果想要确认客户端与服务器间的通讯是否正常，可以在变量中为“消息数量”打勾，实时查看数据的传输情况。

4.9.4 参考代码

代码下载地址：https://github.com/vvlink/SIoT/tree/master/examples/Mind%2B

> 通过网页或者另外的程序，给服务器“xzr/010”Topic 发送消息，多个小动物就跑出来显示内容了，支持中文弹幕。

运行界面：



视频演示：

https://github.com/linmiaoyan/Lins-video/blob/master/%E4%BA%92%E5%8A%A8%E5%BC%B9%E5%B9%95%E6%95%99%E7%A8%8B.mp4

4.9.5 拓展思考

利用这一作品原理，可以制作一些集体互动的大型游戏。

4.10 科学探究："观察"食盐在水中的溶解

溶解是物质科学领域涉及的内容，小学科学课程标准给出的相关活动建议包含"观察常见物质在水中的溶解过程"。食盐是最常见、安全、廉价的实验对象，但传统实验方法却因其扩散过程透明不可见，往往安排学生采用生活中不常见的高锰酸钾代替食盐做溶解实验，学生也只能在高锰酸钾溶解实验的基础上通过想象进行认知迁移。有没有一种方法让食盐的溶解过程可被检测和观察到呢？其实借助虚谷号搭建的 SIoT 服务器和 TDS 传感器制作的实验装置，我们可以用数据可视化的方法变通让食盐溶解过程"可见"。

案例作者：宁波市广济中心小学狄勇

4.10.1 案例描述：

参考教材针对高锰酸钾的实验设计，实验装置主要被设计用于观察搅拌前的食盐溶解情况。我们需要寻找一种可以检测到盐分的传感器以量化该扩散过程。大量程的电导率传感器是最适合的选择，但相对廉价许多的 TDS 传感器也可用于该部分实验。TDS 中文名称为总溶解固体（英文：Total dissolved solids，缩写 TDS），又称溶解性固体总量，测量单位为毫克/升（mg/L），它表明 1 升水中溶有多少毫克溶解性固体。TDS 值越高，表示水中含有的溶解物越多。总溶解固体指水中全部溶质的总量，包括无机物和有机物两者的含量。一般可用电导率值大概了解溶液中的盐份，一般情况下，电导率越高，盐份越高，TDS 越高。由于天然水中所含的有机物以及呈分子状的无机物一般可以不考虑，所以一般也把含盐量称为总溶解固体。

开始实验前，应在教室内部署一台路由器，该路由器无需连接外网，仅负责局域网内数据中转。虚谷号、教室电脑均连至该路由器以实现数据互通。教室电脑提前打开浏览器访问虚谷号的 SIoT 后台，勾选自动刷新消息，做好数据呈现的准备。

实验时将一勺食盐轻轻放入烧杯中，静观食盐的变化。短时间内，沉入水底的食盐颗粒并不会有太大肉眼可见的变化发生，但扩散过程从食盐入水的一刻已经开始。此时可将 TDS 传感器插入烧杯中，并缓缓上下移动探头，检测不同水位的 TDS 值，检测到的数据将由虚谷号上的 SIoT 服务器进行记录，并自动生成图表实现数据可视化。学生将从教室大屏幕呈现的折线图实时看到不同水位的 TDS 值，帮助理解和想象食盐在水中的扩散过程。

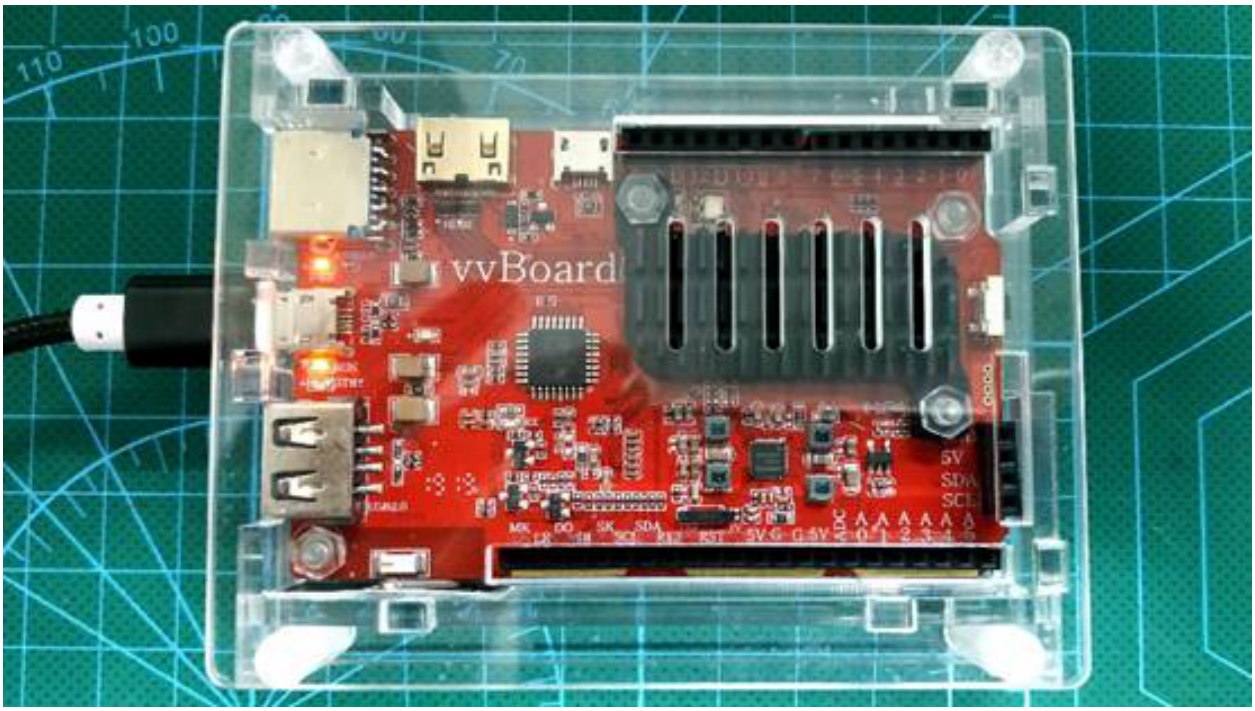
4.10.2 准备工作

4.10.3 在虚谷号上搭建 SIoT 服务器

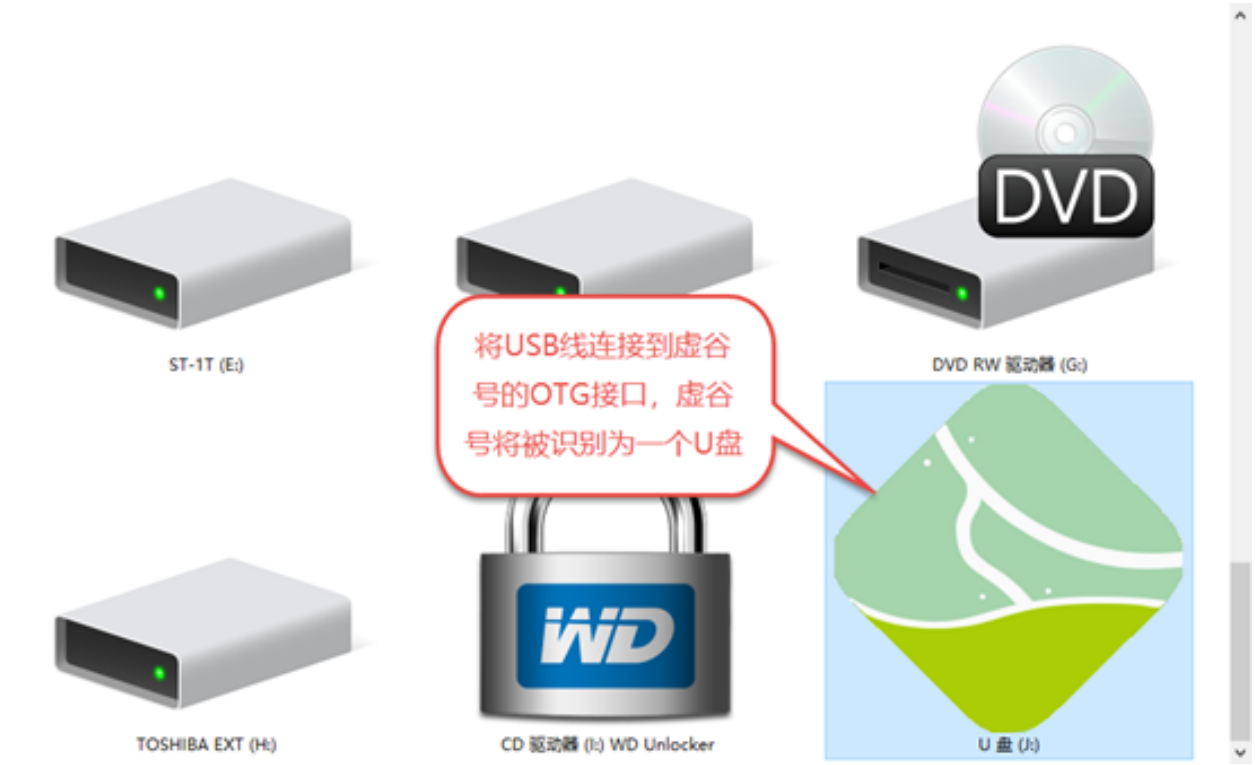
SIoT 已在虚谷号出厂预装。作为一个开源物联网项目，若被删除，可至 GitHub 下载针对 vvboard 的版本安装。(网址：https://SIoT.readthedocs.io/zh_CN/latest/)

虚谷号部署 SIoT 的步骤如下：

1. 将 USB 线连至虚谷号的 OTG 口

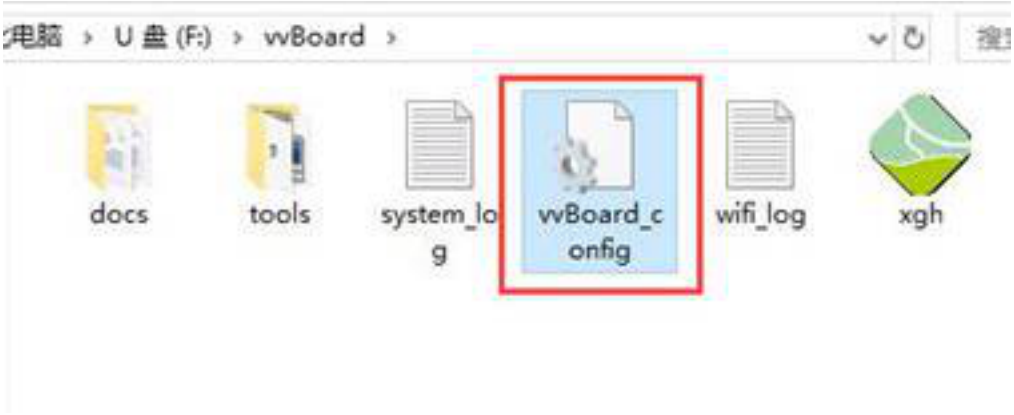


2. 稍后系统会将虚谷号识别为一个 U 盘



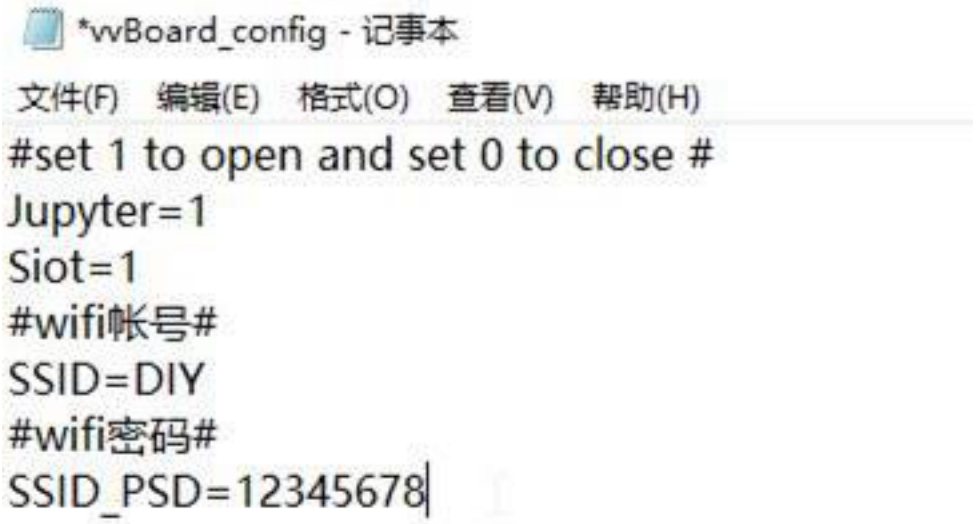
3. 打开 vvBoard 的文件夹

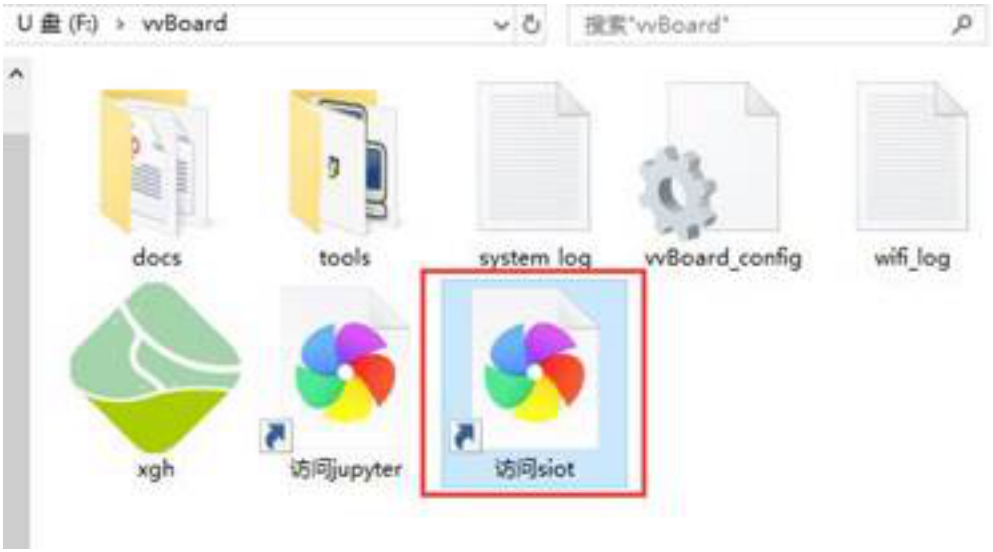
4. 用记事本编辑 vvBoard_config 文件



将 SSID 和 SSID_PSD 改为局域网的 WIFI 账号密码，保存配置文件。

- 注：目前虚谷号和掌控板仅支持 2.4GHz 的 WiFi，且不适用于像校园网之类需要二次认证的登录模式。



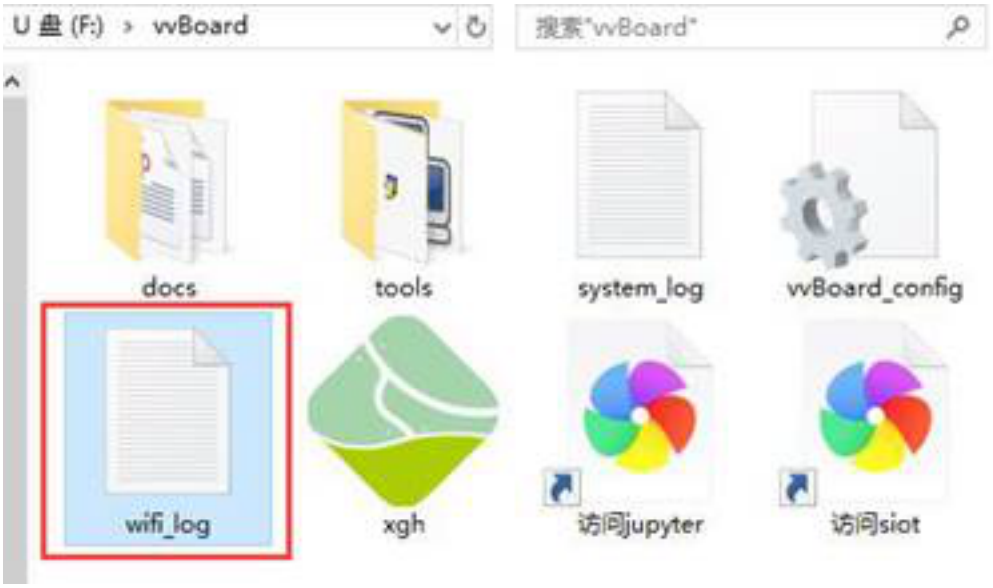


5. 重启虚谷号（断电或长按 RST 键 5 秒），双击“访问 siot”

正常情况下，此时浏览器应呈现 SIOT 的后台登录页面

如果未自动生成包含虚谷号 IP 地址的快捷方式，可再次进入 vvBoard 文件夹，打开其中的 wifi_log 日志文件

如果还是不成功，可以尝试再次重启虚谷号



如下图，虚谷号的 IP 地址为 192.168.0.1，访问 http://192.168.0.101:8080/html/ 即可打开后台页面。

完成上述步操作后，虚谷号便可替代 PC 作为 SIoT 服务器应用于各种实验场景

以下步骤，为虚谷作为服务器，控制板载 Arduino 的案例

4.10.4 操作步骤

一. 实验装置的硬件搭建

材料清单

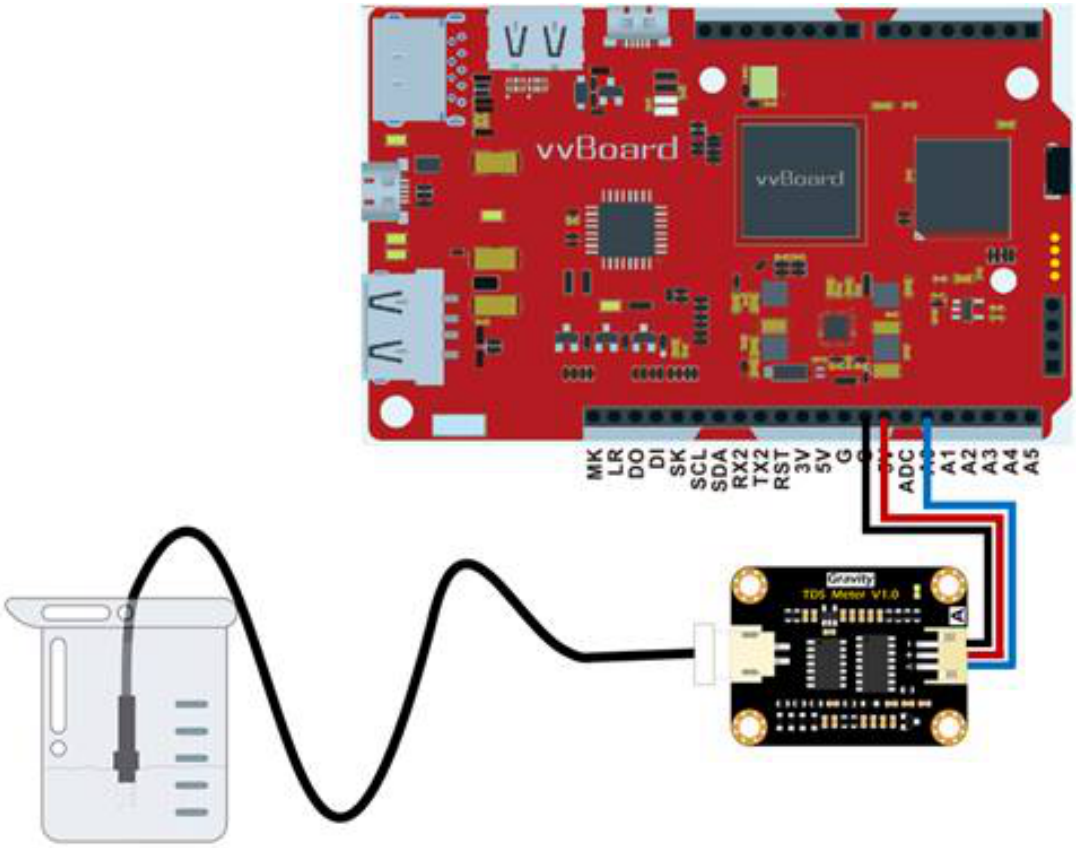
虚谷号 ×1

厚物—虚谷号扩展板 ×1（可选）

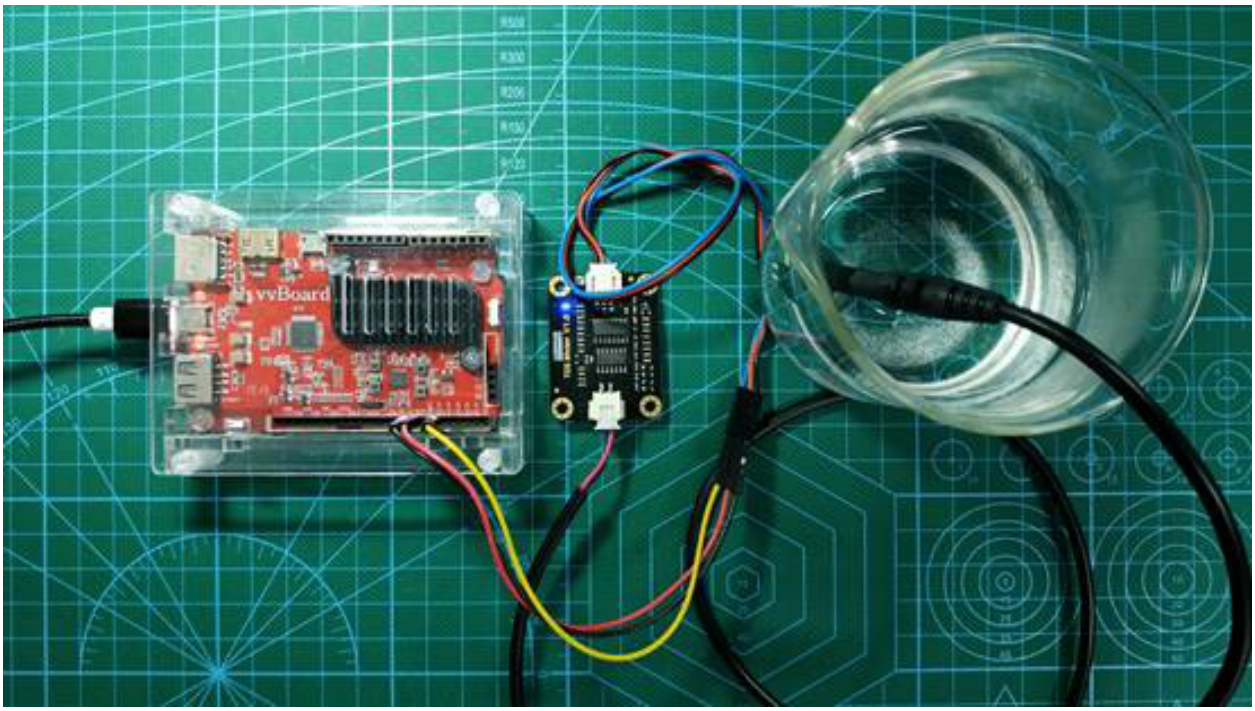
Gravity: 模拟 TDS 传感器 ×1

虚谷号的接口板型和 Arduino UNO 相似，我们可以迁移经验连接设备，将 TDS 传感器连接至虚谷号的 A0 口。引脚对应关系如下

TDS 传感器引脚	虚谷号引脚
A	A0
+	5V
-	G

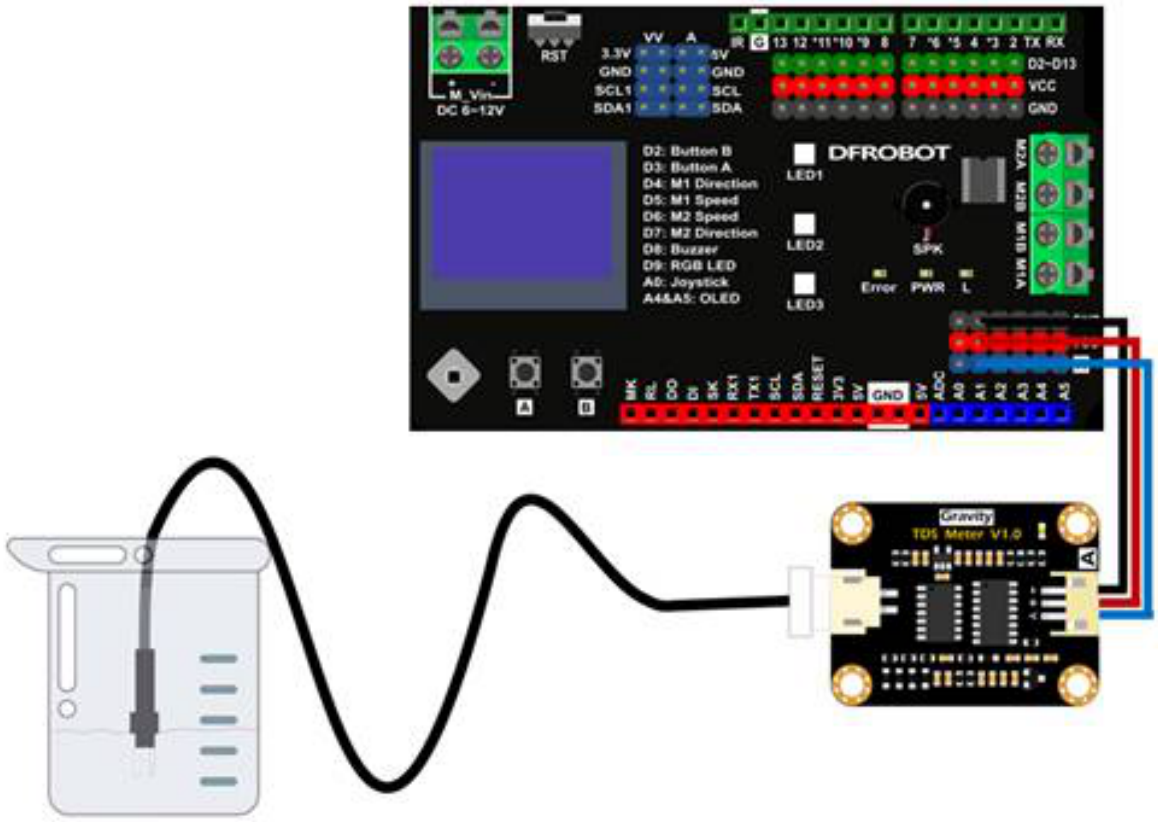


虚谷号直连传感器示意图

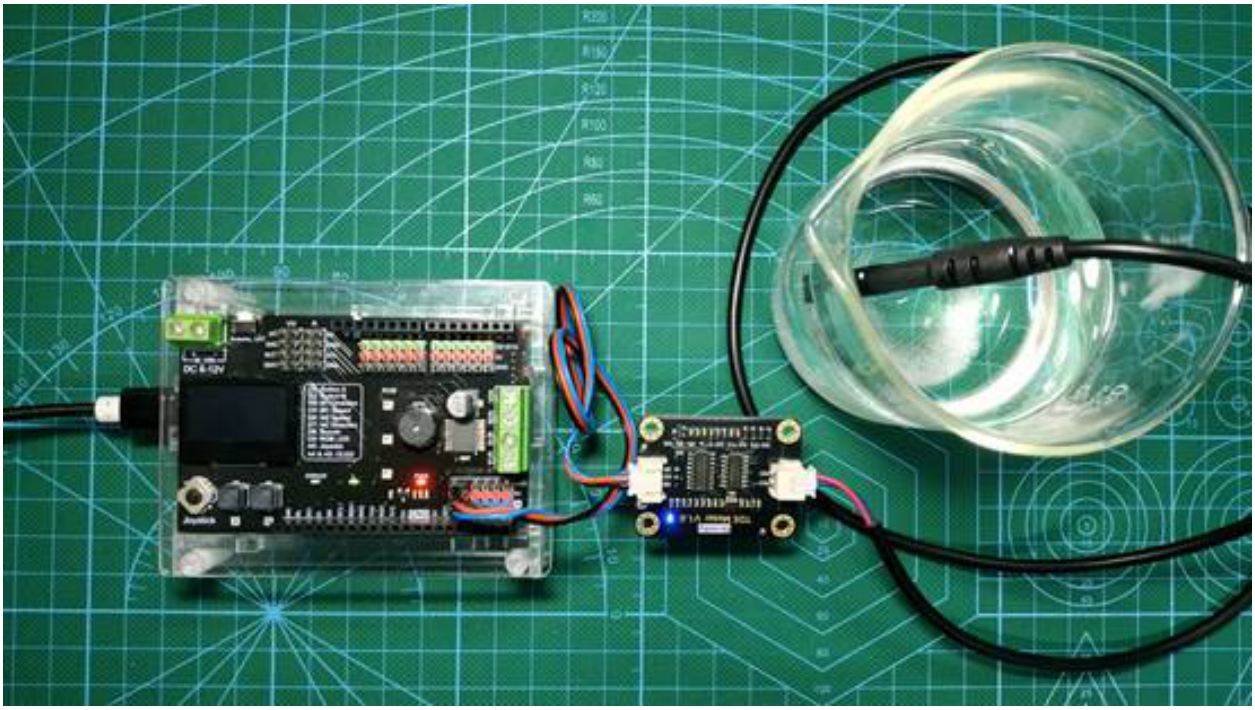


虚谷号直连传感器实物图

如果使用虚谷号专用扩展板——厚物，那么连接 Gravity 系列传感器将更加简单，直接插到扩展板的 3PIN 模拟口上即可。注意由于厚物的 A0 口被板载摇杆占用，我们选择将传感器接入到 A1 口。



使用厚物扩展板转接的连线示意图



使用厚物扩展板转接的实物图

二. 实验装置程序设计

1. 参考 SIoT 官方使用手册中的 Python 章节样例代码（https://siot.readthedocs.io/zh_CN/latest/demo/08_Python.html），编写以下程序，保存为 TDS.py

4.10.5 参考代码

```
import siot
import time
from xugu import Pin # 从 xugu 库中导入 Pin 类

p = Pin("A0", Pin.ANALOG) # 初始化 A0 引脚，设置为输入模式

SERVER = "192.168.0.101"         #MQTT 服务器 IP
CLIENT_ID = ""               # 在 SIoT 上，CLIENT_ID 可以留空
IOT_pubTopic  = 'DIY/TEST01'    #“topic”为“项目名称/设备名称”
IOT_UserName ='scope'          # 用户名
IOT_PassWord ='scope'      # 密码

siot.init(CLIENT_ID, SERVER, user=IOT_UserName, password=IOT_PassWord)
```

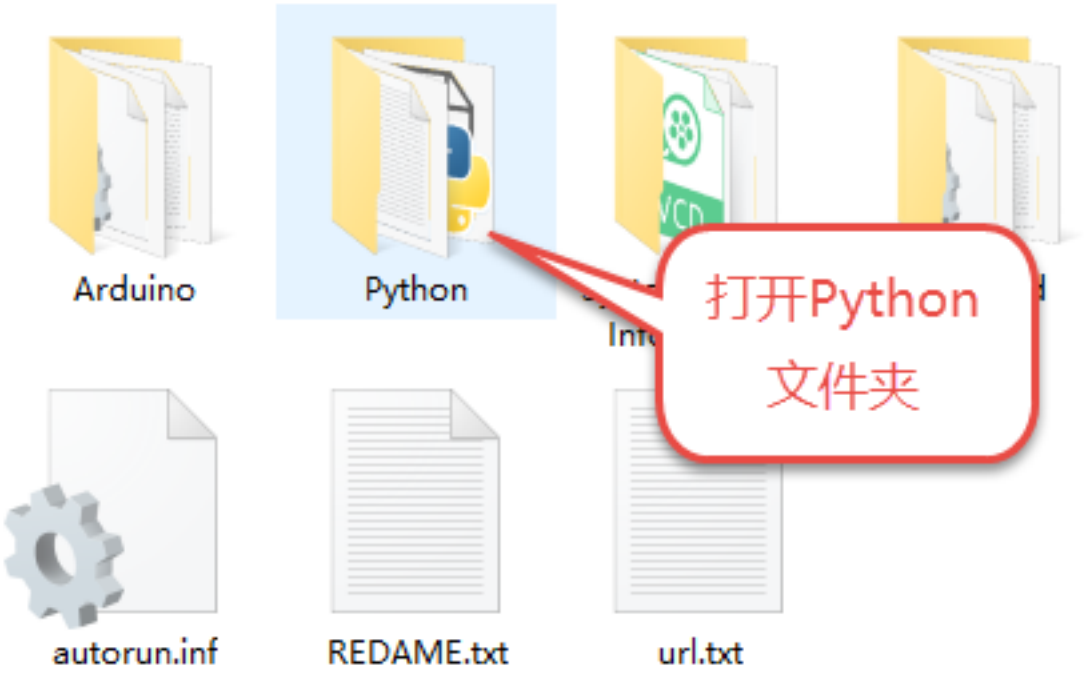
(下页继续)

(续上页)

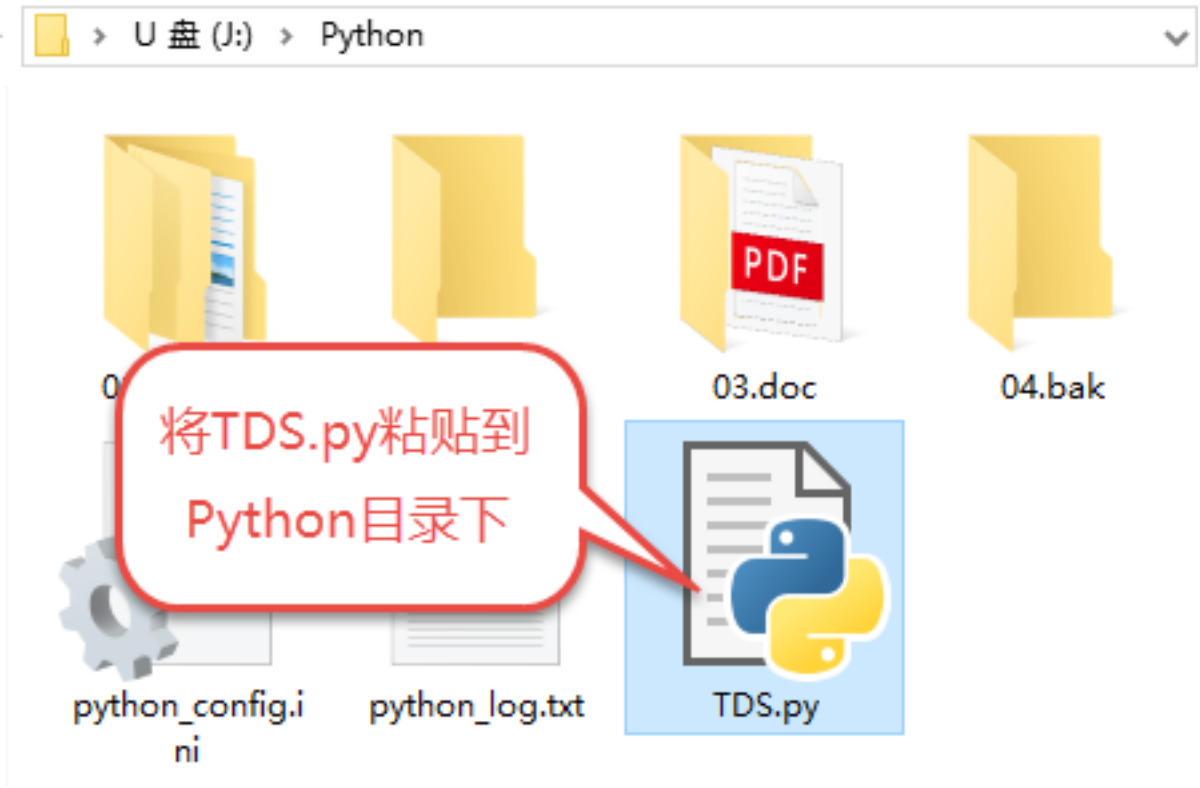
```
def sub_cb(client, userdata, msg):
    print("\nTopic:" + str(msg.topic) + " Message:" + str(msg.payload))

siot.connect()
siot.set_callback(sub_cb)
siot.getsubscribe(IOT_pubTopic)
siot.loop()
while True:
    TDS = p.read_analog() # 读取 A0 引脚的模拟量
    siot.publish(IOT_pubTopic, "%d"%TDS)
    time.sleep(1)
```

代码下载地址：https://github.com/vvlink/SIoT/blob/master/examples/Python/10_vvboard_TDS.py

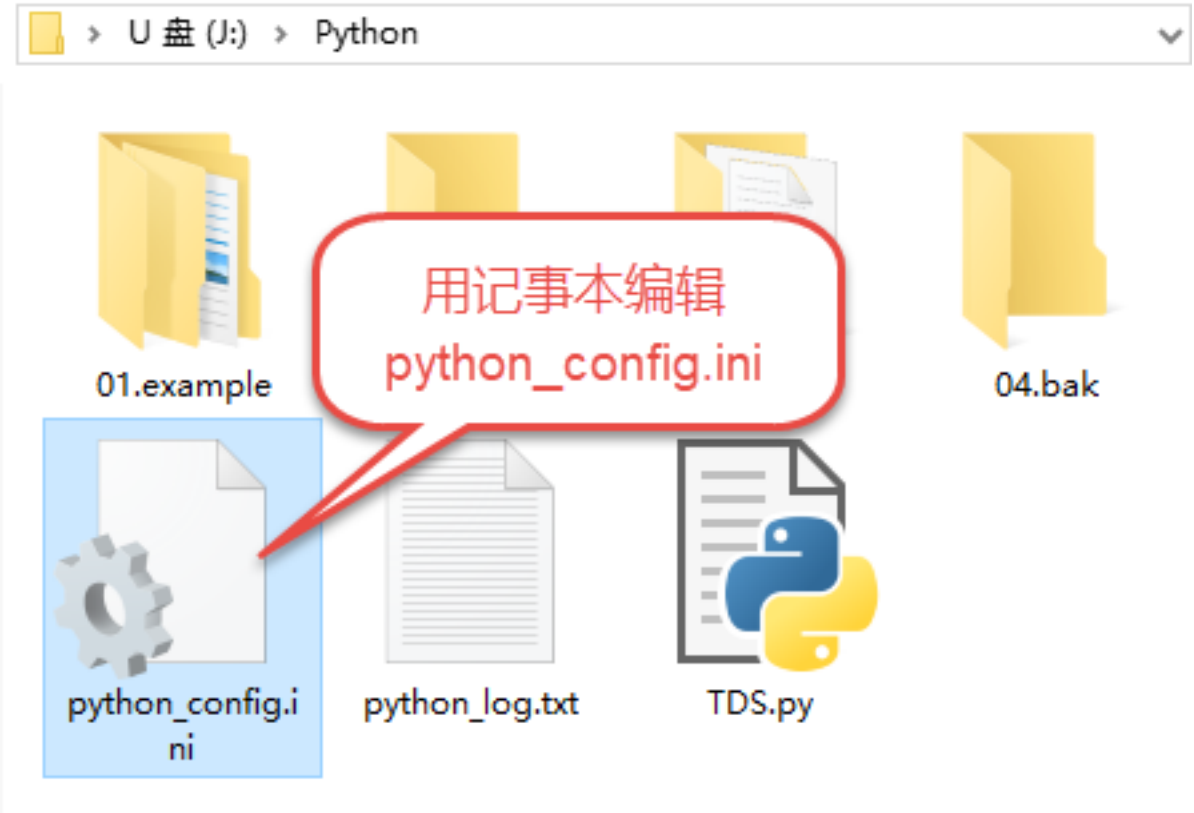
2. 打开 U 盘模式的虚谷号，进入 Python 目录



3. 将 TDS.py 粘贴到 Python 目录下

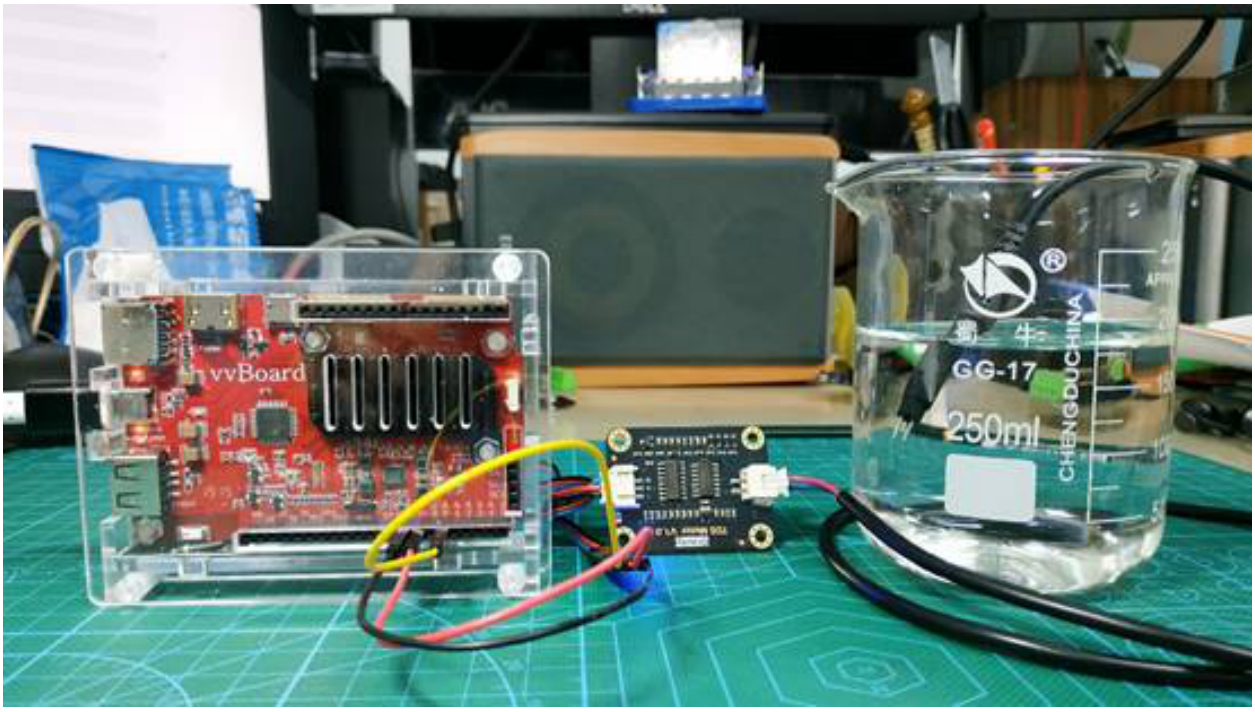


4. 编辑同目录下的 python_config.ini，将首行改为 Python=TDS.py。这样虚谷号开机后将自动运行 TDS.py。



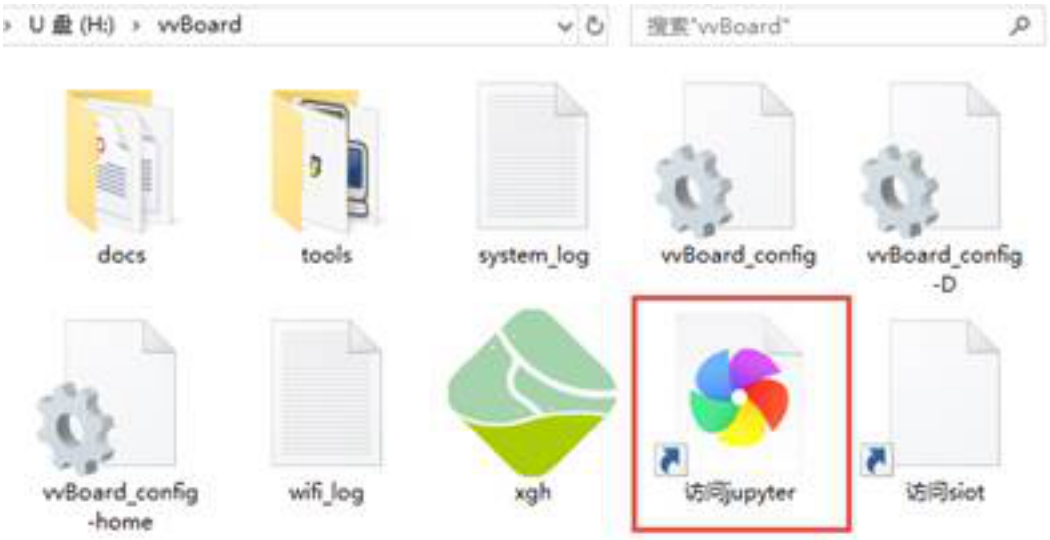
三. 系统测试

根据前述实验原理，我们可通过提升和下放传感器探头，用 SIoT 记录不同水位的 TDS 值，让盐分在水中的分布数据可视化。



测试场景

1. 借助 Jupyter 测试和运行程序。

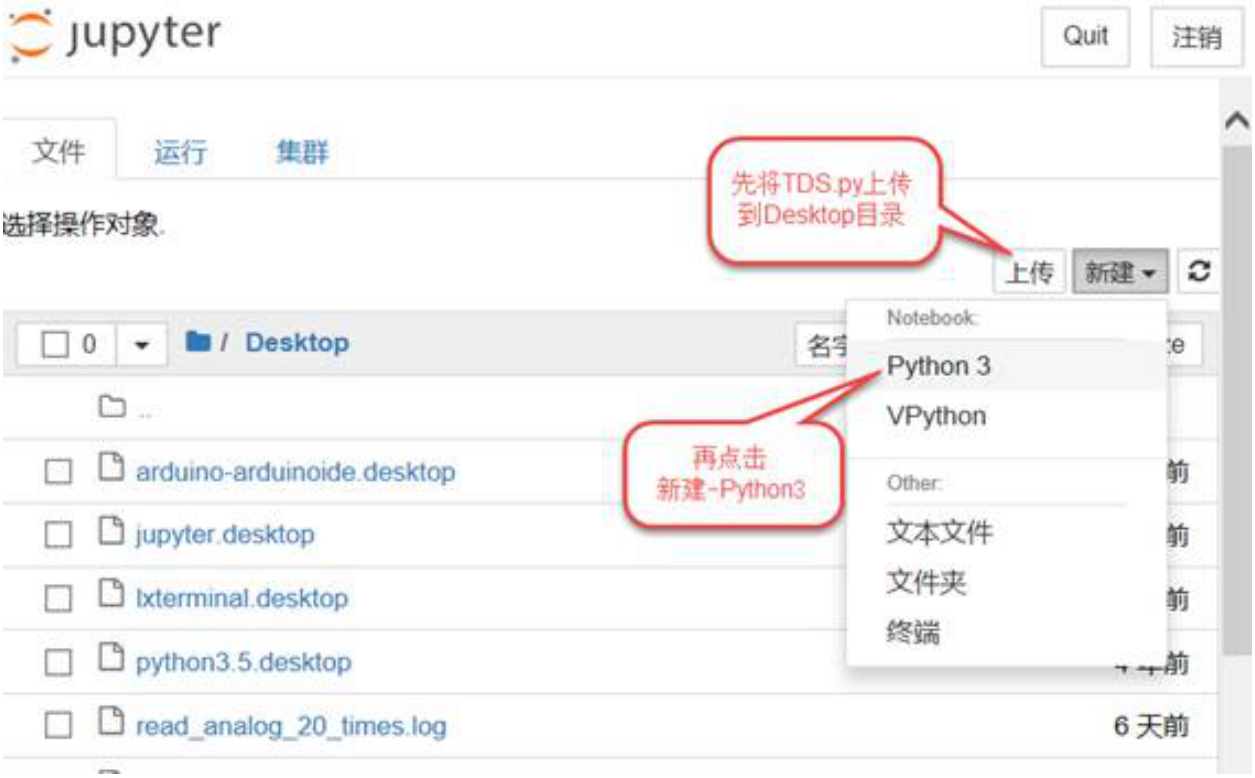


Jupyter 是一个交互式笔记本，支持运行 40 多种编程语言。虚谷号预装了 Jupyter，并且可以通过 U 盘模式下的快捷方式直接在浏览器打开。

Jupyter 默认登录密码为 scope

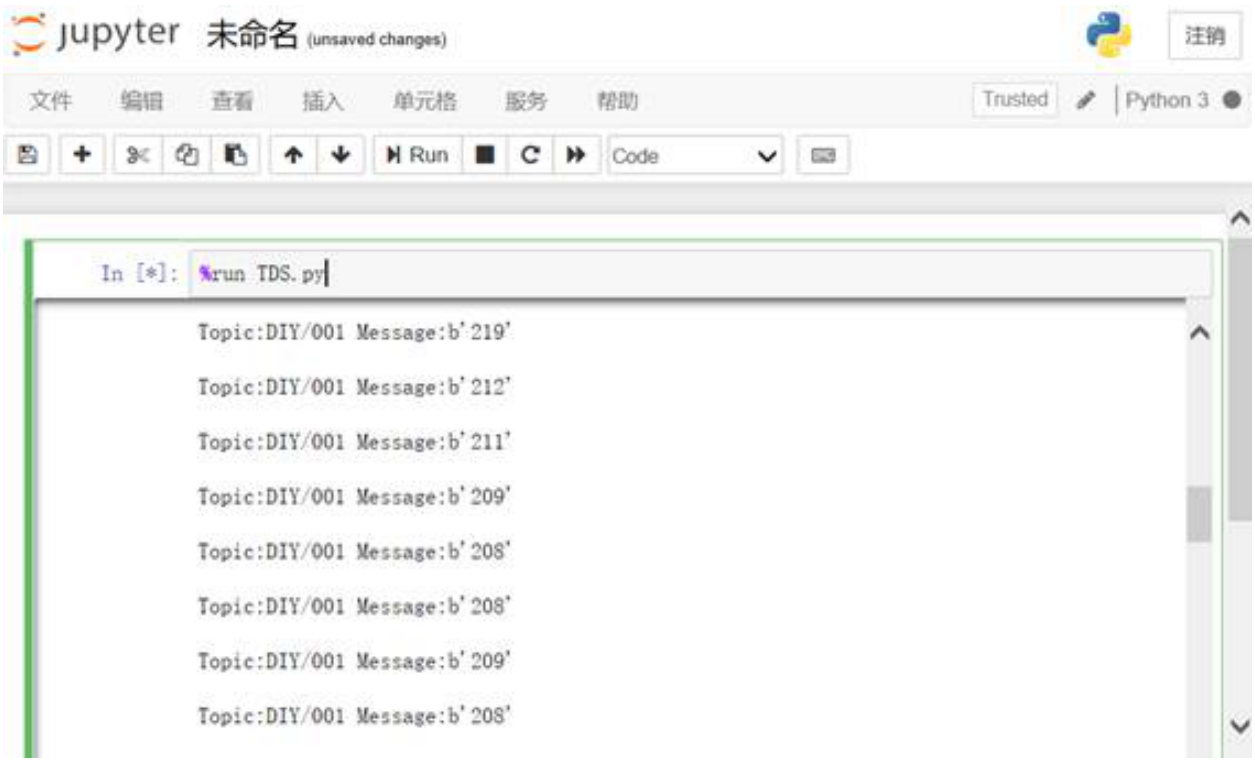
登录后 web 页面会列出虚谷号的文件目录

先将之前编写的 TDS.py 上传到/Desktop 目录下，然后点击新建——Python3



在代码单元格中输入命令%run TDS.py

TDS.py 被执行后，开始加载相关模块并初始化，完成后就可以看到虚谷号上传和返回的数据了。



2. 通过 web 页面测试

由于通过 python_config.ini 设置了 TDS.py 开机运行，通电后我们也可以用浏览器访问虚谷号开启的 SIoT 服务查看装置工作情况。根据前述方法，双击“访问 siot”快捷方式登录后台，可以看到已有数据被记录。实验证明 TDS 传感器对盐分非常敏感，可以恰当反馈盐分浓度的变化趋势。

SIoT 自动生成的折线图

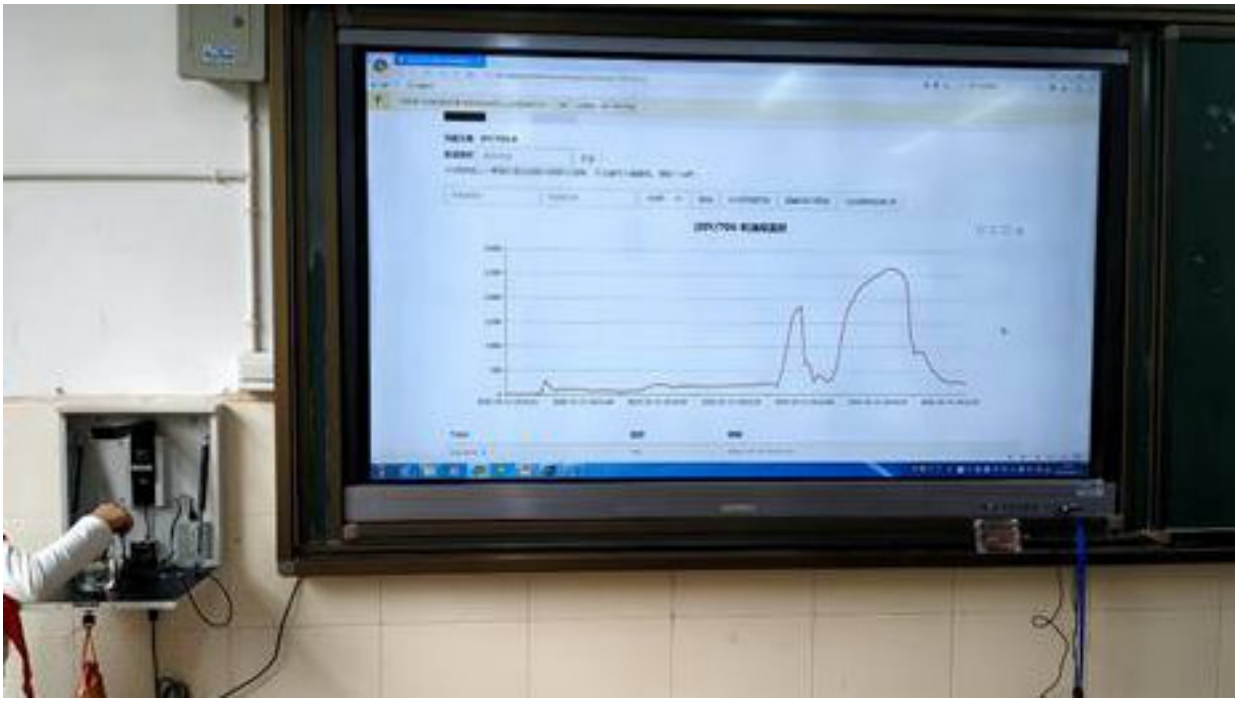
	A	B	C	D
1	ID	主题	消息	时间
2	15559	DIY/TEST01	13	2019/10/21 16:29
3	15558	DIY/TEST01	15	2019/10/21 16:29
4	15557	DIY/TEST01	14	2019/10/21 16:29
5	15556	DIY/TEST01	15	2019/10/21 16:29
6	15555	DIY/TEST01	12	2019/10/21 16:29
7	15554	DIY/TEST01	114	2019/10/21 16:29
8	15553	DIY/TEST01	302	2019/10/21 16:29
9	15552	DIY/TEST01	308	2019/10/21 16:29
10	15551	DIY/TEST01	309	2019/10/21 16:29
11	15550	DIY/TEST01	313	2019/10/21 16:29
12	15549	DIY/TEST01	313	2019/10/21 16:29
13	15548	DIY/TEST01	323	2019/10/21 16:29
14	15547	DIY/TEST01	321	2019/10/21 16:29
15	15546	DIY/TEST01	327	2019/10/21 16:29

以 EXCEL 格式导出的数据

4.10.6 拓展思考

经过教学实践，我们发现学生对基于虚谷物联的实验接受度良好。改进的实验方法以数据和图表的形式，让食盐的扩散过程明晰可见；相较于传统实验手段面对食盐的无能为力，实现了从无到有的进阶，教学上可以作为高锰酸钾溶解实验的有效补充。欲培养学生的科学精神，须让学生懂得“用数据说话”，而数字化科学探究是培养学生这种意识和能力的重要途径。现阶段我们的中小学教育亟需一款开放、实用、廉价、甚至能“跋山涉水”的数字化实验平台。我们认为成本不过几百块，能兼任实验终端和服务器，只需用充电宝供电的虚谷号极致降低了数字化实验室的建设成本，且具备极大的便捷性和灵活性。事实上我们还在课堂上尝试了用掌控板作为终端访问虚谷号 SIoT 的策略，验证了基于虚谷物联对分组实验进行数据回收的方法，可满足不同类型的实验教学需求。最让师生兴奋的是，这种可由学生自己 DIY 的数字化实验装置，其完善过程本身，就是充满乐趣与成就感的货真价实的 STEM 教育。

课堂上由一体机供电的虚谷号



课堂上使用掌控板作为终端访问虚谷号 SIoT

4.11 更多案例（网络收集）

自“虚谷物联”项目发布之后，引发很多教育创客的关注，涌现出一批精彩的案例。

4.11.1 案例：基于掌控板测加速值的科学探究

1. **描述**。 虽然可以通过坐电梯的实验来理解超重、失重的知识，但是看不到数据总觉得很抽象。今天，小麦化身一名物理教师，通过物联网 SIOT，分享一节科学探究物理课。

2. 作者 rzegkly（汝州二高）

3. 链接。http://mc.dfrobot.com.cn/thread-289931-1-1.html

4.11.2 案例：掌控板结合 SIOT 测量光线值

1. **描述**。 掌控板板载光线传感器负责收集光线值，转发 MQTT 服务器，通过 Web 查看消息然后导出数据。

2. 作者 rzegkly（汝州二高）

3. 链接。http://mc.dfrobot.com.cn/thread-288715-1-1.html

4.11.3 案例：远程控制掌控板显示城市气象信息

1. **描述**。 在 Web 页面输入城市名称，掌控板从心知气象网站读取该城市气象信息，然后显示在 OLED 屏幕上。

2. 作者北山脚（浦江中学，方春林）

3. 链接。http://mc.dfrobot.com.cn/thread-289850-1-1.html

4.11.4 案例：掌控板 +SIOT 实现远程开关灯

1. **描述**。 在 Web 页面发送 on 或 off 命令，远程控制掌控板的 LED。

2. 作者 rzegkly（汝州二高）

3. 链接。http://mc.dfrobot.com.cn/thread-281136-1-1.html

4.11.5 案例：micro:bit+SIoT 做热辐射实验

1. **描述**。 教育科学出版社小学《科学》教材五年级上有个章节，内容是《怎样得到更多的光和热》，利用 micro:bit+SIoT，以物联网采集的方式来做这个实验。

2. 作者 digi_cow（宁波，狄勇）

3. 链接。http://mc.dfrobot.com.cn/thread-289794-1-1.html

4.11.6 案例：心率监测–科学助跑

1. **描述**。 当看到自己的心率值出现在屏幕上，顿时感到很兴奋。现在很多智能手环/手表都可以用来监测人体的生理信息，如心率，血压，步数，睡眠质量等，帮助大家关注并了解自己的身体状况。

2. 作者 youjingisland

3. 链接。http://mc.dfrobot.com.cn/thread-297794-1-1.html

CHAPTER 5

高级操作技巧

这一部分主要介绍 SIoT 的一些高级操作。

5.1 安全设置

为了方便中小学的老师们在课堂上教学和应用物联网技术，SIoT 采取了一种最简单的方式，即采用了统一的用户名和密码，加自定义 Topic 的方式使用物联网平台。设置简单并不等于不安全，哪怕在多人使用的情况下，SIoT 也有一定的机制确保数据的安全。

5.1.1 密码设置

修改 config.json 中的，可以设置用户名和密码。

```
{
"User": "siot",
"Password": "dfrobot",
"WebServerAdrr": "0.0.0.0:8080",
"MqttAdrr": "0.0.0.0:1883",
"OnlyLocalURD": false
}
```

- config.json 可以通过记事本来修改，打开时可以右键选择打开方式为记事本，编辑结束后点击保存即可。

5.1.2 Web 管理端口修改

通过修改”WebServerAdrr”，可以修改 Web 管理的地址和端口。

- “0.0.0.0:8080”，表示所有可用的 IP 地址。
- “127.0.0.1:8080”，表示只有本机访问才能登录 Web 管理页面，可以确保数据安全。

5.1.3 权限隔离

OnlyLocalURD 值默认是 false，表示对项目和设备的管理不需要限制在“本地访问”。在多人使用的情况下，如果担心数据被恶意篡改或者设备被删除，可以将这个值设置为 true。当 OnlyLocalURD 值为 true，如果不是通过 127.0.0.1 的地址登录 Web 管理页面，将不能看到任何项目和设备，只能通过网页给 Topic 发送消息。

5.2 数据导出

5.3 WebAPI

SIoT 提供了一系列的 WebAPI，供用户调用。一些不支持 MQTT 的编程语言，如 VB、S4A 等，可以通过 HTTP 的形式和 SIoT 进行互动。

需要强调的是，HTTP 的请求/应答方式的会话都是客户端发起的，缺乏服务器通知客户端的机制，如果要获取服务器信息，客户端应用需要不断地轮询服务器。这也就是在物联网方面，MQTT 比 HTTP 更受欢迎的原因。

5.3.1 API 列表：

这里的账号密码为默认的 siot 和 dfrobot，如果您曾修改过账号或密码，这里的 iname 或 ipwd 也需要相应的修改

- 发布消息

Http://[SIoT 的 IP]:8080/publish?topic=xzr/001&msg=on&iname=siot&ipwd=dfrobot

说明：向 topicid（主题）“xzr/001”发送内容为“on”的消息，其中“xzr”是项目 id，“001”是设备 id。

返回数据：{ “code” :1,” msg” :” 数据已发送” }

- 获取最新数据

Http://[SIoT 的 IP]:8080/lastmessage?topic=xzr/001&iname=siot&ipwd=dfrobot

说明：获取 topicid（主题）“xzr/001”的最新一条消息

返回数据：{ “code” :1,” data” :[{ “ID” :27,” Topic” :” xzr/001” ,” Content” :” 11” ,” Created” :” 2019-06-06 11:55:39” }],” msg” :” 成功” }

- 获取消息列表

Http://[SIoT 的 IP]:8080/messages?topic=xzr/001&iname=siot&ipwd=dfrobot&pnum=1&psize=10&begin=04-04 00:00:00&end=2019-07-01 00:00:00

Http://[SIoT 的 IP]:8080/messages?topic=xzr/001&iname=siot&ipwd=dfrobot&pnum=1&psize=10&begin=04-04&end=2019-07-01

说明：获取 topicid（主题）“xzr/001” 从 “2019-04-04” 到 “2019-07-01” 的数据。其中时间可以省略。

返回数据：{ “code” :1,” data” :[{ “ID” :26,” Topic” :” xzr/001” ,” Content” :” 10” ,” Created” :” 2019-06-06 11:55:33” },{ “ID” :27,” Topic” :” xzr/001” ,” Content” :” 11” ,” Created” :” 2019-06-06 11:55:39” }],” msg” :” 成功” }

- 清除消息

Http://[SIoT 的 IP]:8080/clearmsg?topic=xzr/001&iname=siot&ipwd=dfrobot

说明：删除 topicid（主题）“xzr/001” 的所有消息

返回数据：{ “code” :1,” data” :4,” msg” :” 成功清空消息” }

- 获取项目列表

Http://[SIoT 的 IP]:8080/projects?iname=siot&ipwd=dfrobot

说明：

返回数据:” code” :1,” data” :[{ “ID” :” DFRobot” ,” Description” :” 一个项目中包含多个设备 t” ,” Created” :1556265411},{ “ID” :” abc” ,” Description” :”” ,” Created” :1556939389},{ “ID” :” xzr” ,” Description” :”” ,” Created” :1556941184}],” msg” :” 成功” }

- 更新项目

Http://[SIoT 的 IP]:8080/updateprj?pid=xzr&iname=siot&ipwd=dfrobot&desc=科学测量

说明：将名称为 “xzr” 的项目的备注修改为 “科学测量”

返回数据：{ “code” :1,” msg” :” 成功” }

- 获取设备列表

Http://[SIoT 的 IP]:8080/devices?pid=xzr&iname=siot&ipwd=dfrobot

说明：返回项目名称为 “xzr” 的设备列表。

- 更新设备

Http://[SIoT 的 IP]:8080/updatedev?pid=xzr&dname=001&iname=siot&ipwd=dfrobot&desc=台灯控制

说明：将项目名称为 “xzr”，设备名称为 “001” 的设备，备注修改为 “台灯控制”（发现 BUG）

{ “code” :1,” msg” :” 成功” }

- 删除设备

Http://[SIoT 的 IP]:8080/deldev?topic=xzr/001&iname=siot&ipwd=dfrobot

说明：

返回数据：{ “code” :1,” msg” :” 删除成功” }

5.3.2 返回数据的通用格式说明：

{ code = 1, message = “”, data = {}

} 说明：code 等于 1，表示成功。data 中返回列表信息。

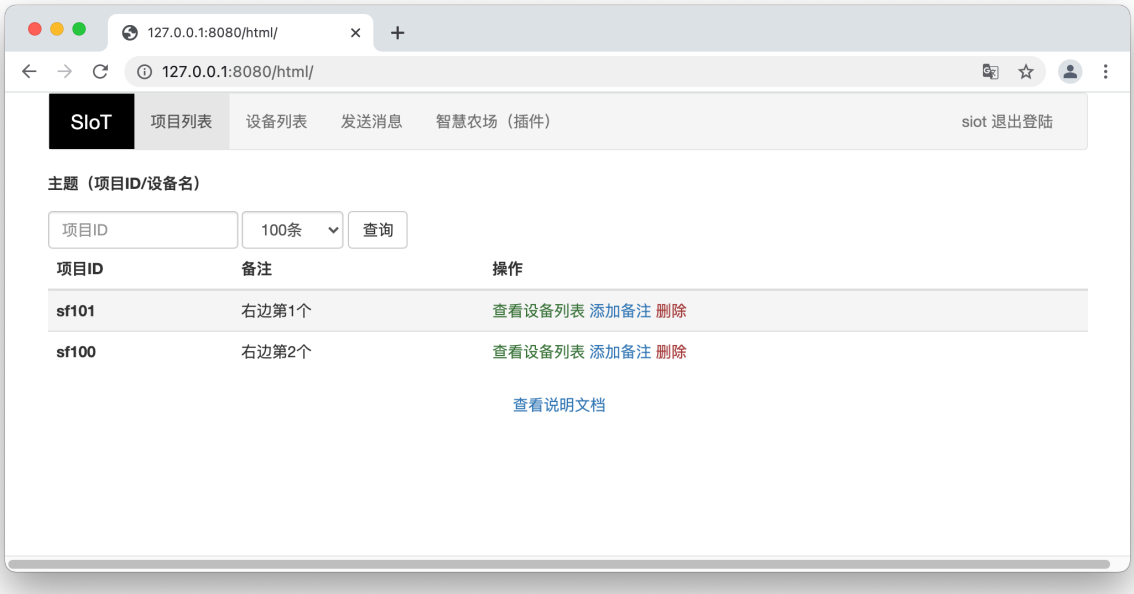
5.4 扩展插件

SIoT 采用了前端和后端数据分离的方式开发 Web 管理界面，因而支持插件编写。

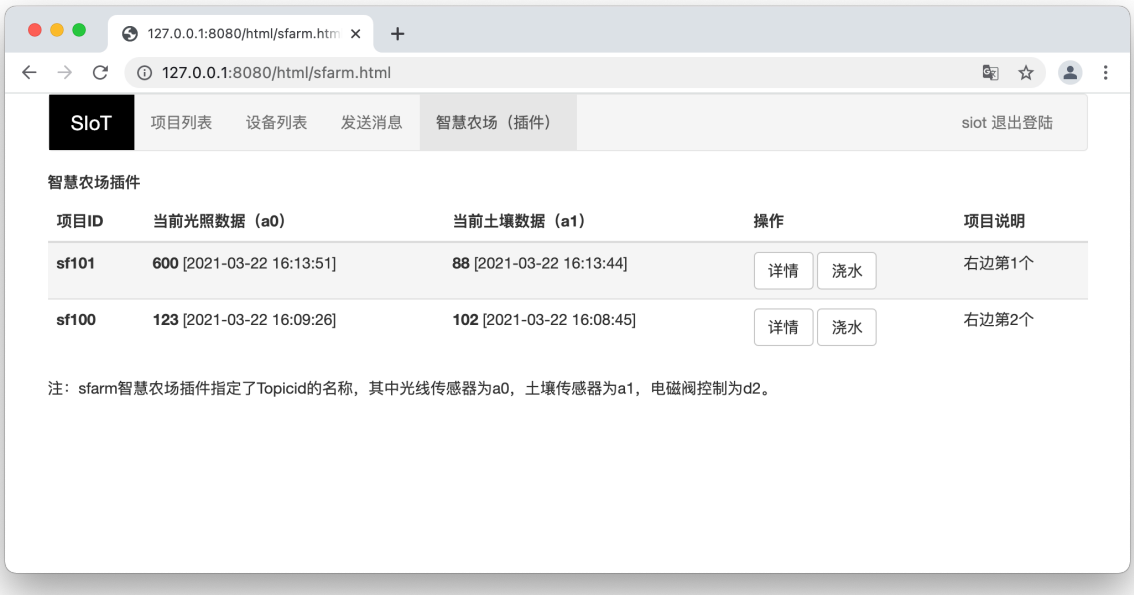
Html 文件夹中，存放 Web 管理页面，基于 Vue.js 编写。可以直接修改这些 Html 文件，加入新的功能。

虚谷物联团队提供了开源智慧农场（sfarm）的插件，在页面中点击按钮即可发送 “浇水” 指令。

1. 默认主页效果



2. 插件页面效果



通过开源智慧农场（sfarm）的插件来管理特定的项目，会更加直观、便捷。

5.5 常见问题解答

5.5.1 1. 为什么掌控板无法连接 SIoT？

造成掌控板无法连接 SIoT 的原因很多，一般要从这几个方面进行检查：

- 运行 SIoT 的电脑，是否正常启用了 1883 端口。

  建议用 MQTT 客户端程序来访问 SIoT，如手机端的 MQTTool（iPhone）或 MQTT Client（安卓）、电脑端的 MQTTbox（一般是除运行 SIOT 的电脑外的另一台电脑），确认 MQTT 服务是否正常。如果不正常，建议关闭各种防火墙软件，尤其是 360。

  测试方法可以参考：客户端连接范例 https://siot.readthedocs.io/zh_CN/latest/demo/index.html

- 掌控板和运行 SIoT 的电脑（服务器）是否可以相互访问。

  （1）使用 Ping 命令，在运行 SIoT 的电脑 Ping 掌控板的 IP 地址。

  （2）将掌控板和运行 SIoT 的电脑接入同一个局域网，如连接同一个无线路由器。

  （3）用手机热点提供的局域网常常不稳定，导致连接失败，可以尝试其他方法搭建局域网。

- 找到问题之后，一般重启即可解决问题。重启一般按照以下顺序进行：路由器-SIOT 服务器-掌控板。
- 如果仍然没有解决，您可以在 github 上发起 issue 或者直接联系开发人员，我们将及时回复您的问题。

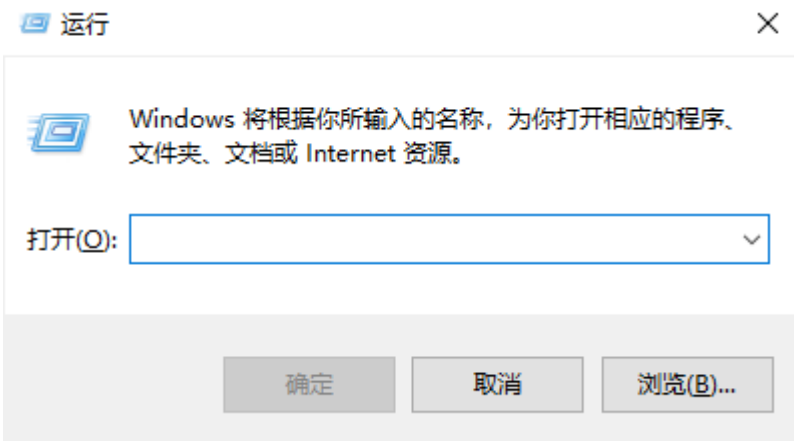
5.5.2 2. 如何获取电脑的 IP 地址？

电脑每次连接 WIFI，都会生成一个 IP 地址，每个 IP 地址所对应的电脑是唯一的。运行 SIOT 程序后会在电脑上建立一个 SIOT 服务器，其他设备要访问这个服务器，需要知道这个 SIOT 服务器所在电脑的 IP 地址。

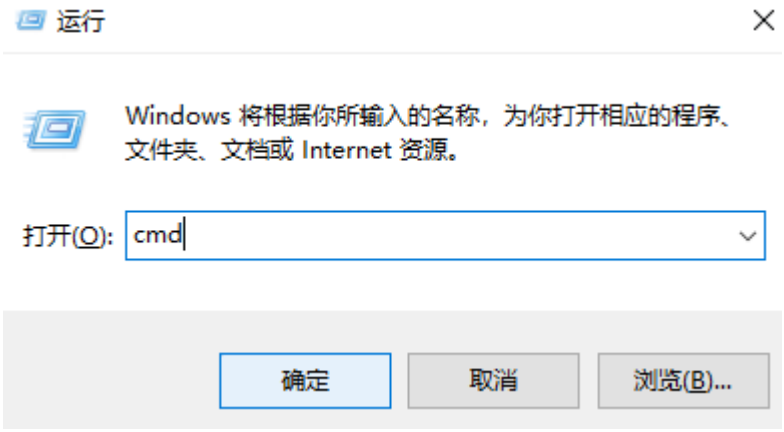
SIoTv1.2 及以上版本在启动时提供参考的 IP 地址，可以逐个尝试。如果都不行，可以尝试以下方法。

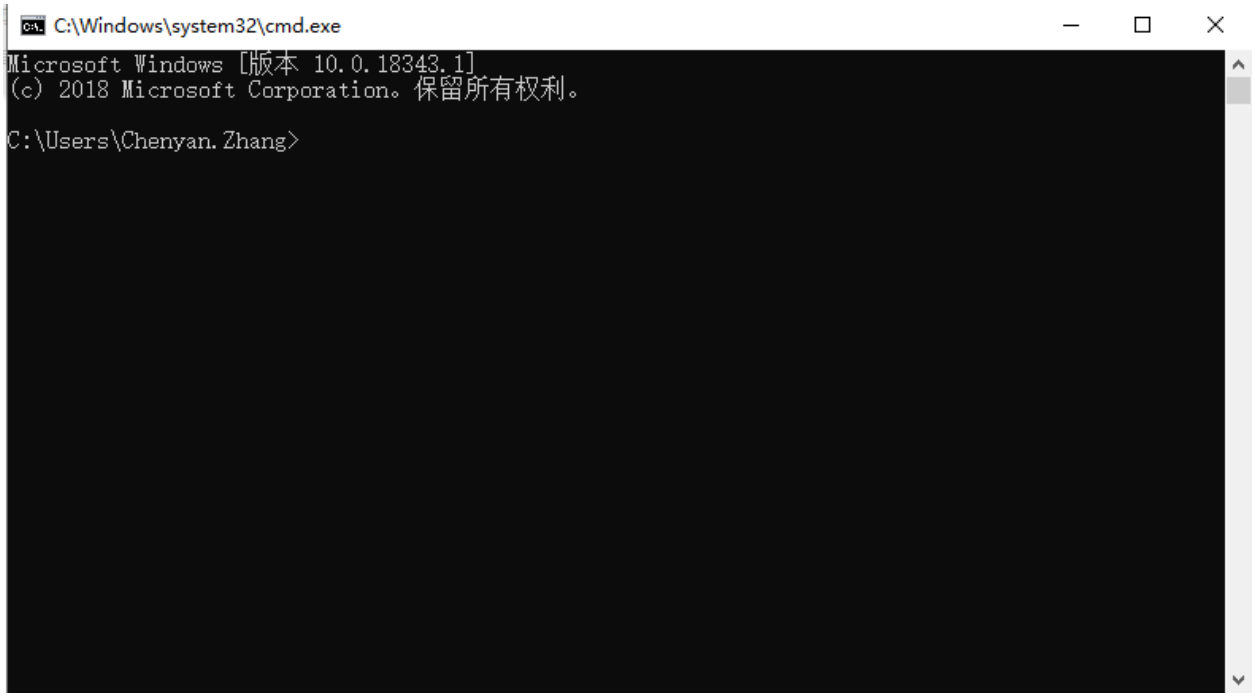
获取电脑 IP 的方法有很多，可在网页上搜索到，下面我们来介绍其中一种简易操作方法，通过以下 3 步获取电脑 IP 地址。

1）同时按下键盘上“WIN”＋“R”，弹出如下运行窗口。

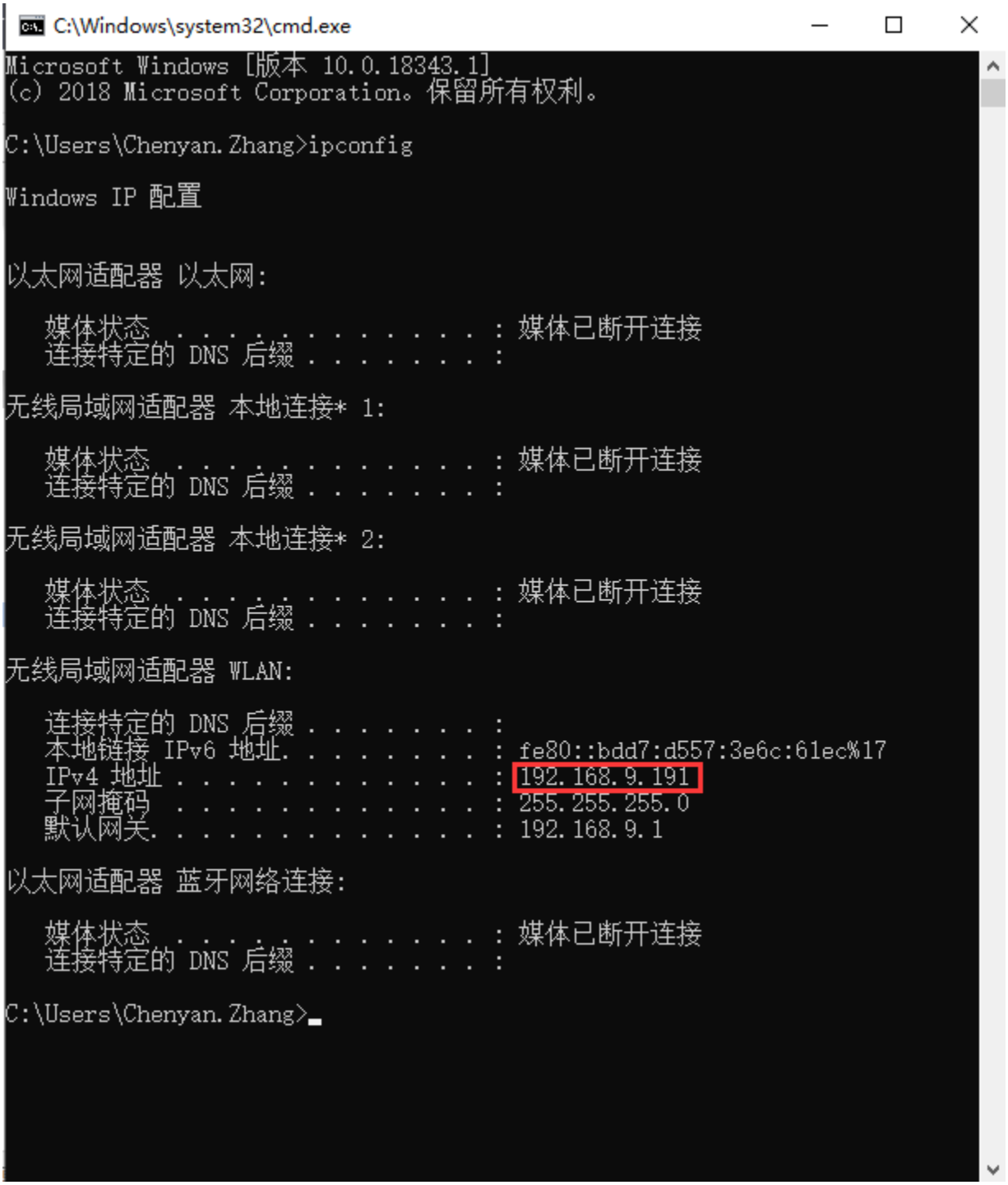


2）输入“cmd”，点击确定，弹出小黑框。





3）在小黑框中输入“ipconfig”，点击键盘“enter”，在小黑框中可以看到 IP 地址，如下图 IP 为 192.168.9.191。

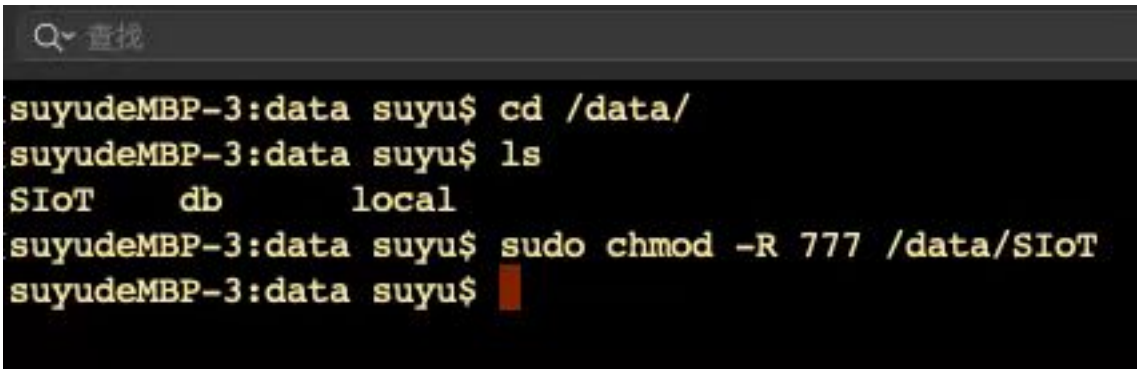


注意：每次连接不同的 WIFI，电脑的 IP 地址都可能会发生变化，需要通过上述方法重新获取。

5.5.3 3. 如何在虚谷号中运行 SIoT

将 SIoT（linux 版本）拷贝到虚谷号中, 例如/data/SIoT 目录。然后，将该目录的权限设定为最高权限。方法如下: 打开 terminal(终端) 输入: chmod -R 777 /data/SIoT 双击运行虚谷号的可执行文件即可。如果希

望这个程序能够在后台保持运行，请使用 nohup 命令，如：nohup ./SIoT_linux & 其中“SIoT_linux”为程序的路径。



CHAPTER 6

附录

这一部分主要介绍与物联网技术、掌控板相关的资源，如掌控板的无线广播、物联网平台的触发器设计等。

6.1 siot 的 Python 库

SIoT 是一款开源的 MQTT 服务器，siot 则是 Python 和 MicroPython 的一个库。

6.1.1 简介

MQTT 的库明显不好用，前面要定义一个类，代码还很长，让初学者不知所措。

siot 是虚谷物联团队写的一个 Python 库。为了让初学者能够写出更加简洁、优雅的 Python 代码。

siot 库同时支持 MicroPython，语法完全一致。

6.1.2 安装

使用 pip 命令：pip install siot

注：虚谷号中已经内置了 siot 库。

6.1.3 官方地址

- GitHub 地址：https://github.com/vvlink/SIoT/tree/master/siot-lib

6.1.4 使用说明

1. 导入库

import siot

2. 连接 MQTT 服务器

siot.init(CLIENT_ID, SERVER, user=IOT_UserName, password=IOT_PassWord)

siot.connect()

siot.loop()

3. 发送消息

siot.publish(IOT_pubTopic, "value %d" %tick)

4. 订阅消息

siot.subscribe(IOT_pubTopic, sub_cb)

"sub_cb" 为回调函数的名称，需要写一个名称为 "sub_cb" 的函数，带 3 个参数，其中 msg 为订阅的消息，为一个元组，属性有 topic 和 payload。如下面的代码。

::

def sub_cb(client, userdata, msg): print("nTopic:" + str(msg.topic) + " Message:" + str(msg.payload))

6.1.5 代码范例

1. 发送消息：

```
import siot
import time

SERVER = "127.0.0.1"            #MQTT 服务器 IP
CLIENT_ID = ""                  # 在 SIoT 上，CLIENT_ID 可以留空
IOT_pubTopic  = 'xzr/001'       # "topic" 为 "项目名称/设备名称"
IOT_UserName ='siot'            # 用户名
IOT_PassWord ='dfrobot'         # 密码

siot.init(CLIENT_ID, SERVER, user=IOT_UserName, password=IOT_PassWord)
siot.connect()
siot.loop()

tick = 0
```

(下页继续)

(续上页)

```
try:
    while True:
        siot.publish(IOT_pubTopic, "value %d"%tick)
        time.sleep(1)            # 隔 1 秒发送一次
        tick = tick+1
except:
   siot.stop()
   print("disconnect seccused")
```

- 订阅消息：

```
import siot
import time

SERVER = "127.0.0.1"         #MQTT 服务器 IP
CLIENT_ID = ""               # 在 SIoT 上，CLIENT_ID 可以留空
IOT_pubTopic  = 'xzr/001'    # “topic” 为 “项目名称/设备名称”
IOT_UserName ='siot'         # 用户名
IOT_PassWord ='dfrobot'      # 密码

def sub_cb(client, userdata, msg):
  print("\nTopic:" + str(msg.topic) + " Message:" + str(msg.payload))

siot.init(CLIENT_ID, SERVER, user=IOT_UserName, password=IOT_PassWord)
siot.connect()
siot.subscribe(IOT_pubTopic, sub_cb)
siot.loop()

try:
  while True:
    pass
except:
  siot.stop()
  print("disconnect seccused")
```

6.2 OneNET 的触发器设计

掌控板可以将数据传递至 OneNET 物联网平台，我们可以在 OneNET 物联网平台根据一定的条件设置触发器。

6.2.1 具体步骤

1. 准备好掌控板及数据连接线，打开 mPythonX 0.3.2，连接串口。

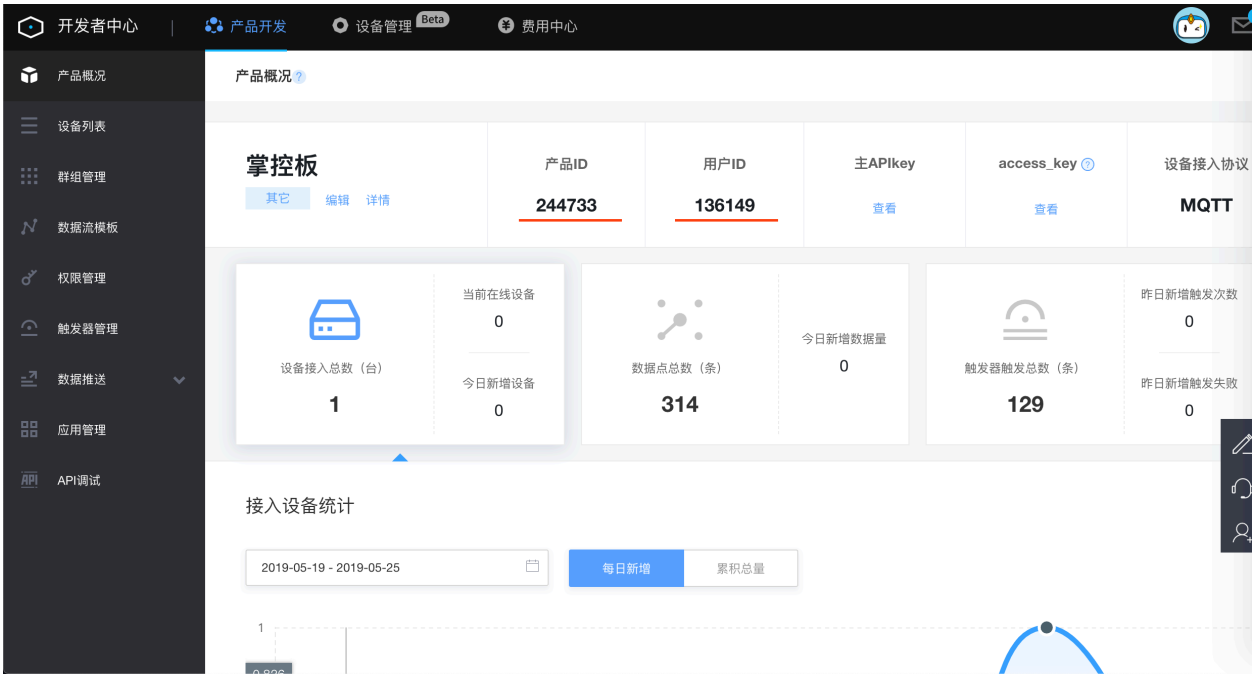
2. 在 OneNET 物联网平台使用手机号进行注册、登陆。OneNET 地址 https://open.iot.10086.cn/

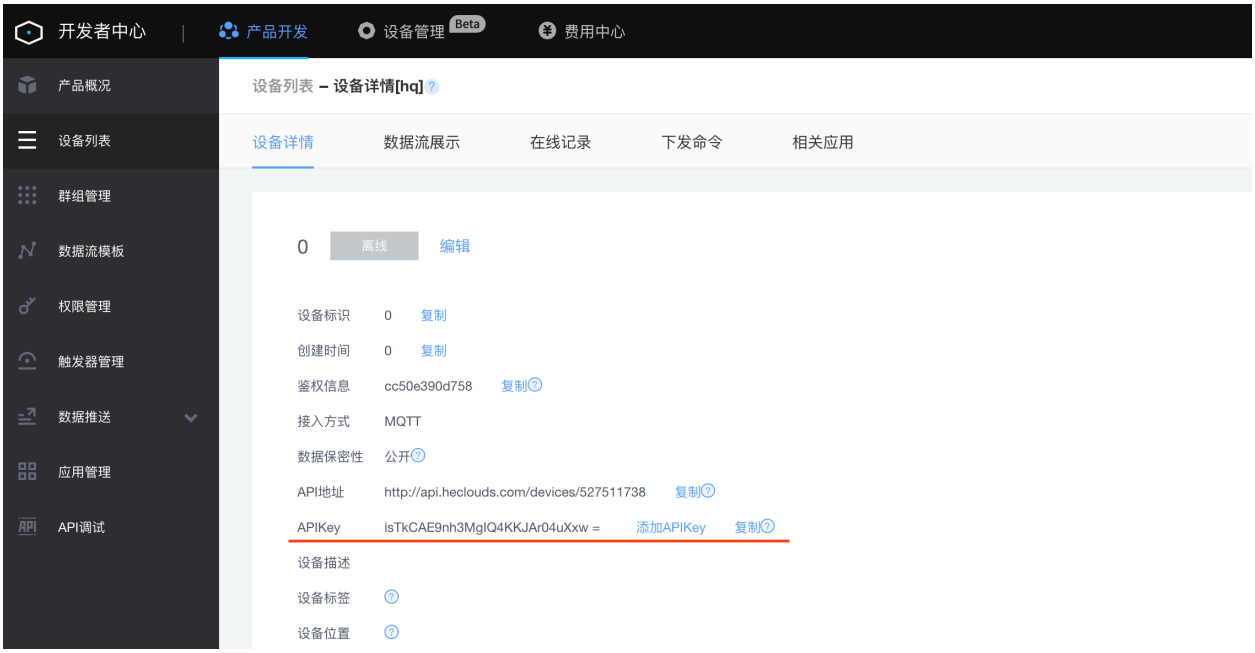
3. 在 OneNET “开发者中心”中添加产品与设备，可以通过开发者文档了解具体步骤。

开发者文档地址为 https://open.iot.10086.cn/doc/book/easy-manual/product&device/product-create.html

提示：协议选择 MQTT。

记录所需信息。





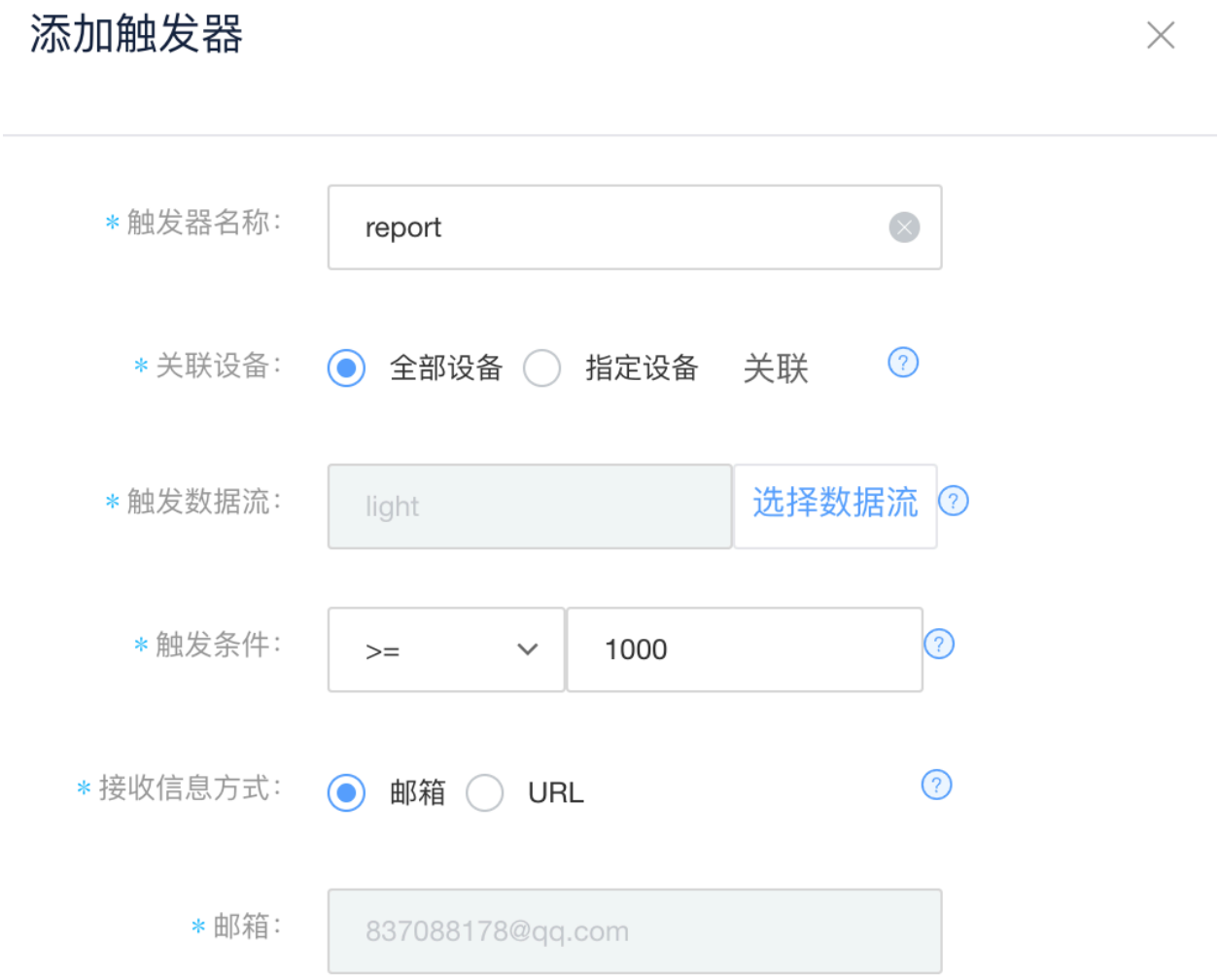
添加数据流模板，其名称需与图形代码中传递的数据名称相同。

4. 完成程序代码。图形代码如下。

提示：服务器地址为默认值，无需更改。

5. 在 OneNET 中查看设备状态及数据，设备状态应为“在线”，可以清晰地看到读取的数值了。

6. 在 OneNET 中点击触发器管理，按照操作提示添加一个触发器。



效果如图。


中国移动
下午9:24
收件箱
【OneNET】您的hq设备的report触发器在2019-05-26 21:20:08被触发。详情：light数据流、>=类型、触发值1070
OneNET
详情
触发器信息
触发器id: 1450524
触发器名: report
类型: >=
阈值: 1000
触发数据
设备id: 527511738
设备名: hq
数据流: light
触发时间: 2019-05-26T21:20:08.777
触发值: 1070

6.2.2 示例代码

https://github.com/vvlink/SIoT/blob/master/examples/Python/trigger.xml

6.3 MQTT 信息的发送和订阅（mPythonX）

基于 mPythonX，我们可利用掌控板、MQTT 服务器实现消息的发送和订阅。

6.3.1 准备工作

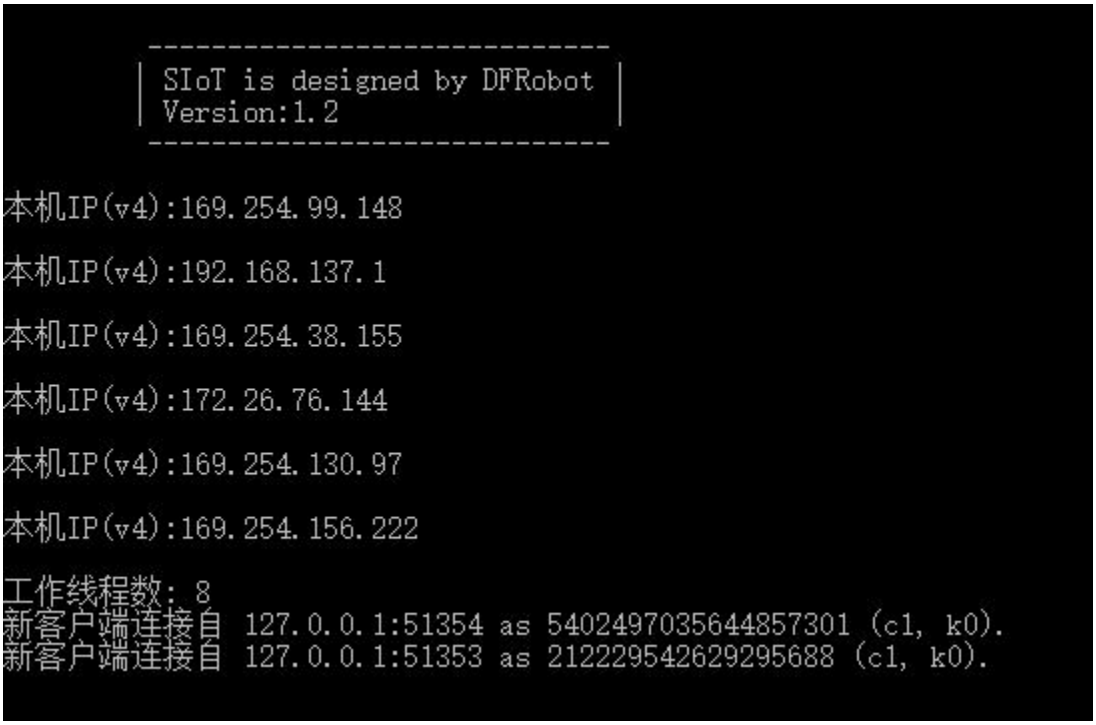
（一）硬件准备

掌控板及其连接线

（二）软件准备

1. 搭建 SIoT 服务器

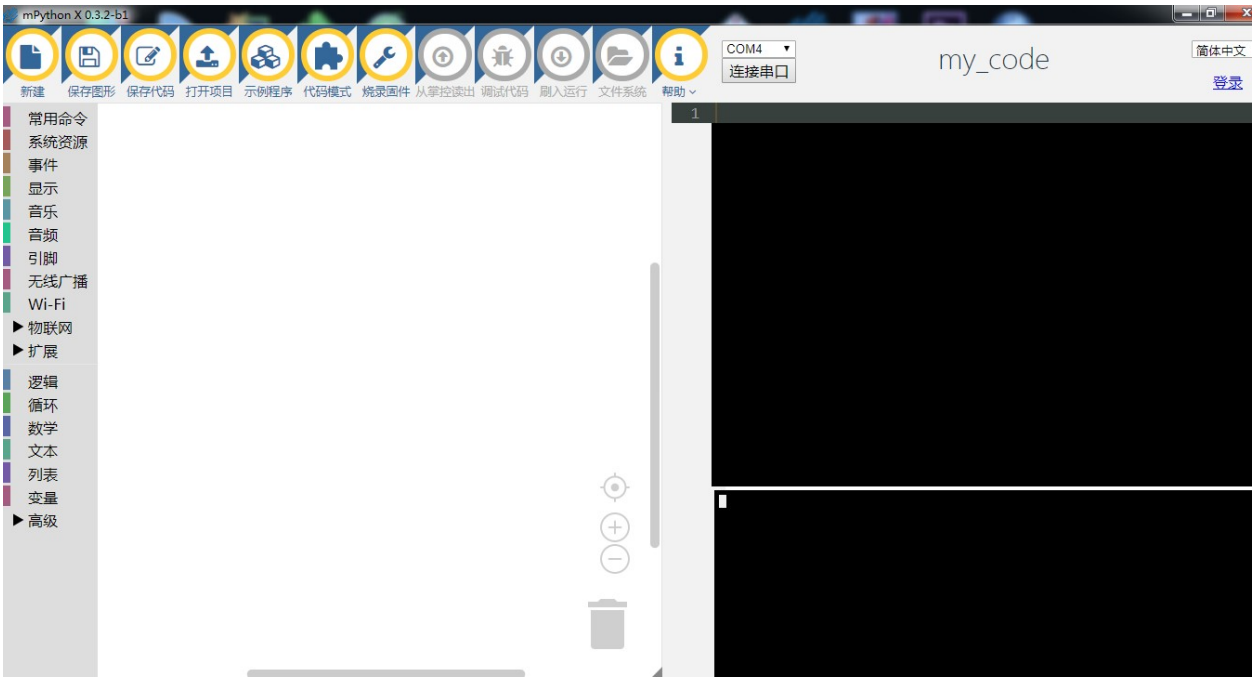
直接双击点击与系统匹配的 SIoT 运行文件，屏幕会弹出一个黑色的 CMD 窗口，在配置中请不要关闭它。



2. 登录 SIoT 平台

打开浏览器，输入 url：http://localhost:8080 （超链接）

3. 打开 mPythonX 0.3.2 编写程序



6.3.2 步骤

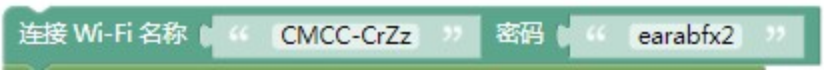
(一) 参考程序

注：为确保数据持续成功发送，注意定时器发送消息的用法。

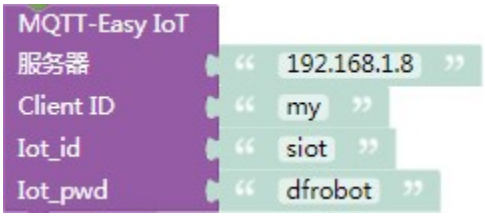
（二）具体操作

1. 打开 mPythonX 0.3.2，连接串口。

2. 手动修改可连接的 WiFi 名称与密码。

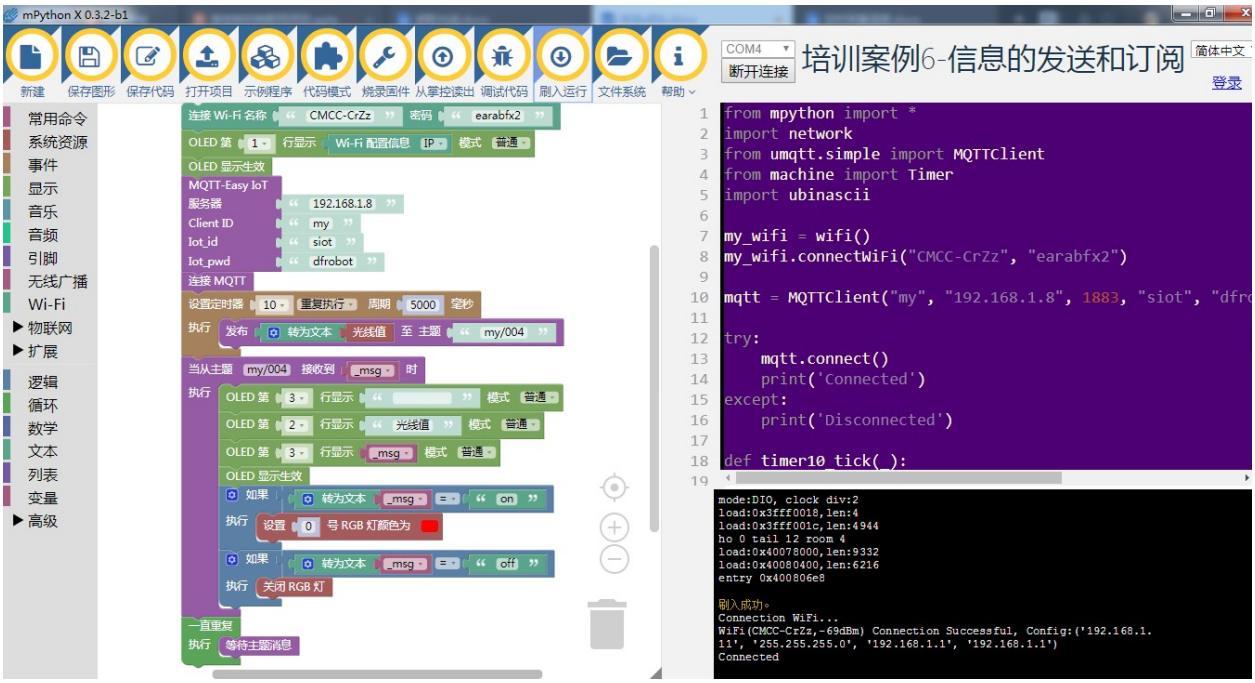


3. 设置 MQTT 初始化参数。服务器地址为本地 IP 地址，Client_ID 为项目 ID，Iot_id 和 Iot_pwd 即 SIoT 使用的账号密码。



4. 手动修改主题为“项目 ID/名称”。

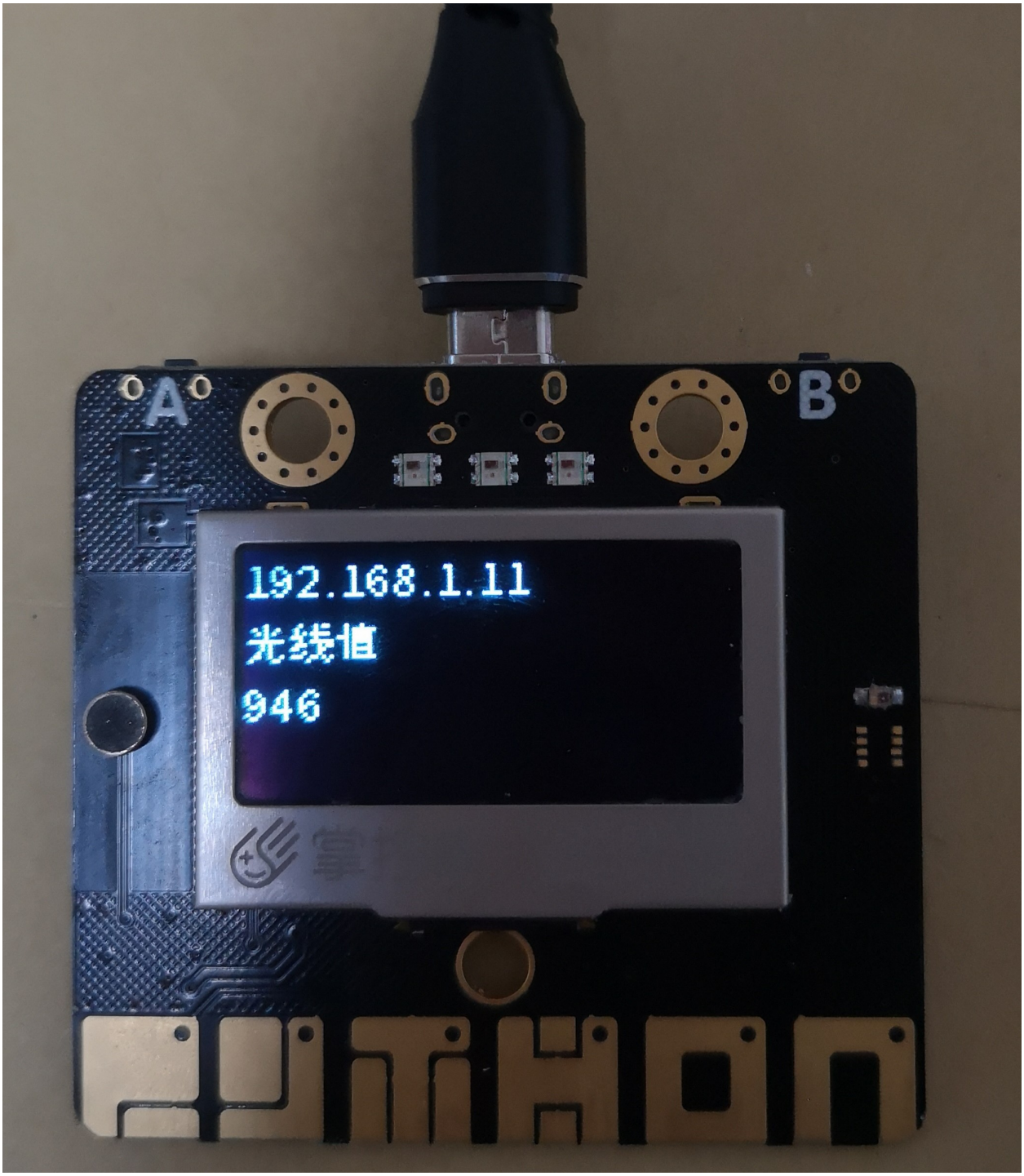
5. 将程序“刷入运行”进行测试，界面右下角显示当前程序的运行进程。



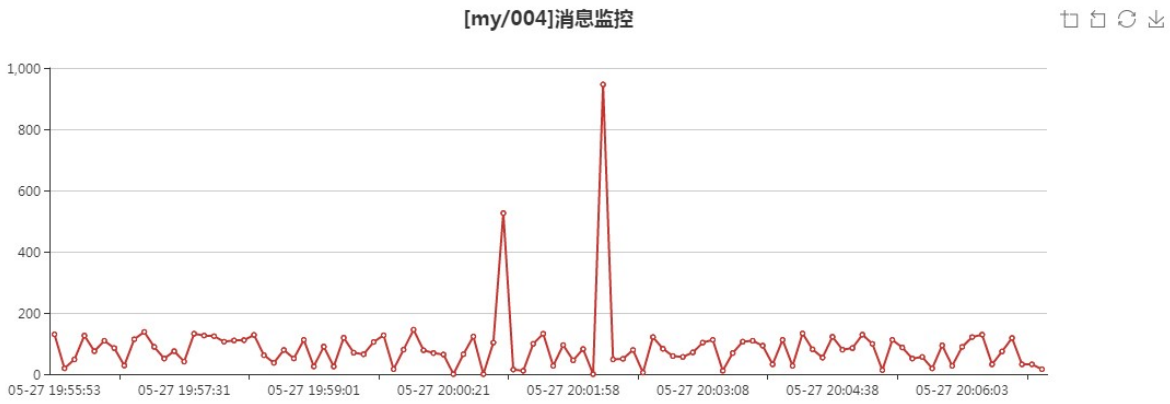
（三）运行结果

1. 信息的发送

（1）掌控版屏幕显示当前光线值



(2) SIoT 平台设备成功接收光线值



2. 信息的订阅

在 SIoT 平台给掌控板发送消息“on”，掌控板最左侧灯变成红色。相同操作，发送消息“off”，灯灭。

6.3.3 代码

代码下载地址：https://github.com/vvlink/SIoT/blob/master/examples/Python/sendingandsubscribe.xml