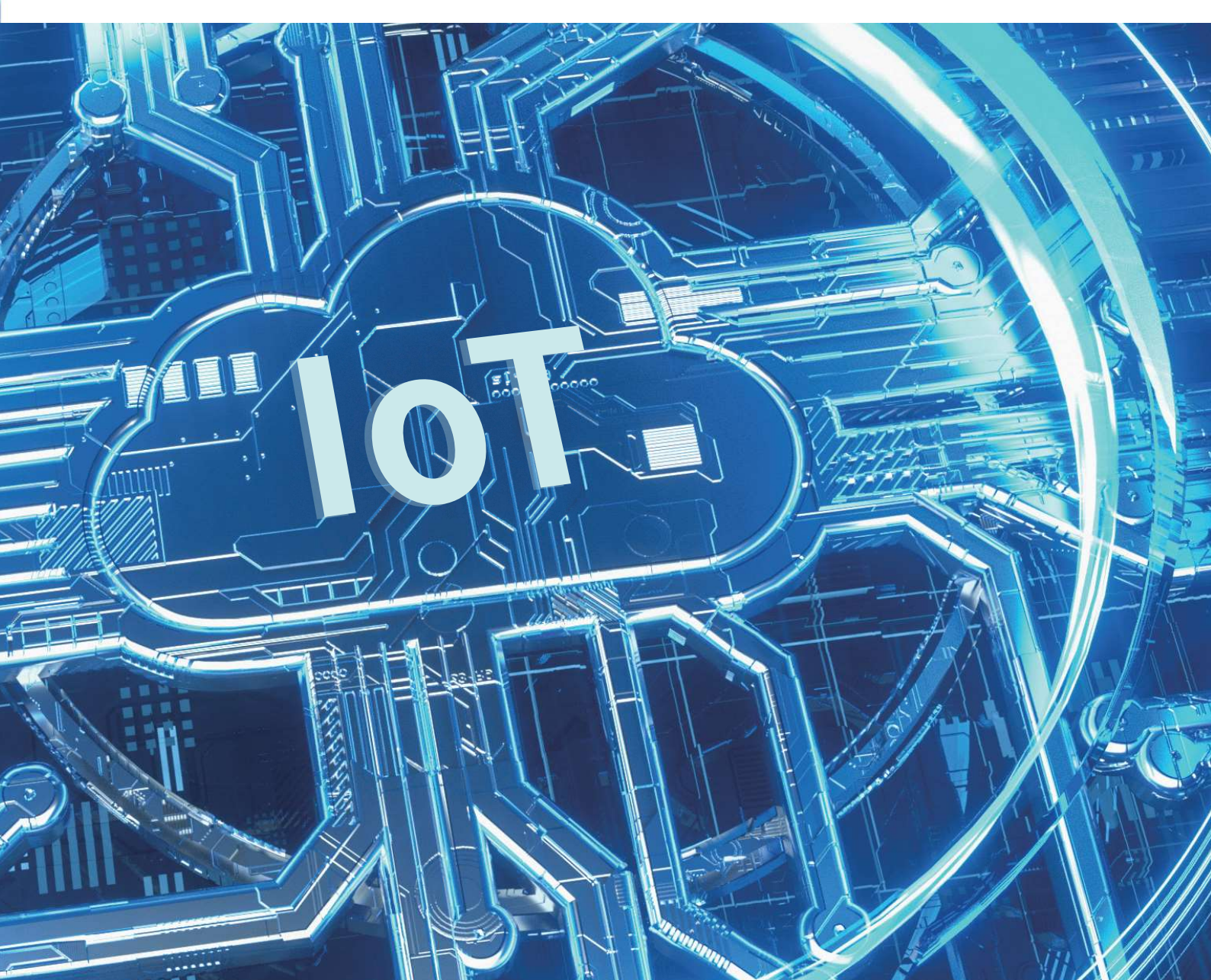
DFROBOT®
DRIVE THE FUTURE

蘑菇云

初中
信息技术教材

实践项目指南



扫码关注
蘑菇云创造
了解更多信息

DFROBOT®
DRIVE THE FUTURE

蘑菇云

DFROBOT
初中信息技术教材实践项目指南
BOK0093

目录

扫描二维码可下载项目程序包

序章

教程使用说明

一、教程简介

本套教程配套于初中信息技术教材包使用，是 Python 编程入门教程的第二部分。教程中的项目来源于八年级下信息技术教材，结合硬件，能够让学生加深对 Python 语言的理解与应用。

教程中，以 Mind+软件为编程工具，使用 Python 编程语言，结合 micro:bit 控制器及其余诸多硬件，实现智能厨房安防、植物监测、鱼缸水位控制等各类场景的应用项目，不仅能够让学生加深对 Python 语言的理解与应用，还能补充学习各类硬件知识并巩固知识点在实际运用中的作用。

二、教程大纲

三、单节教程结构

一、实践情境
二、实践目标
三、知识目标
四、实践准备
五、实践过程 任务 1 分析设计 硬件搭建 软件编写 软件设置 程序编写、运行及回顾 调试修改 任务 2 分析设计 硬件搭建 软件编写 软件设置 程序编写、运行及回顾 调试修改
六、巩固提高 1、项目小结 2、项目拓展
附录

软硬件概述

本系列教程以 **Mind+软件**的“**Python 模式**”为开发工具（自带基于 Python3.6.5 的开发环境），结合 Python 基础语法以及 pinpong 库，编写 Python 代码，控制 **micro:bit 开发板**等硬件器材实现项目的探究设计。

其中，我们的硬件器材可以分为**控制器**、**执行器**、**传感器**和配件四大部分。

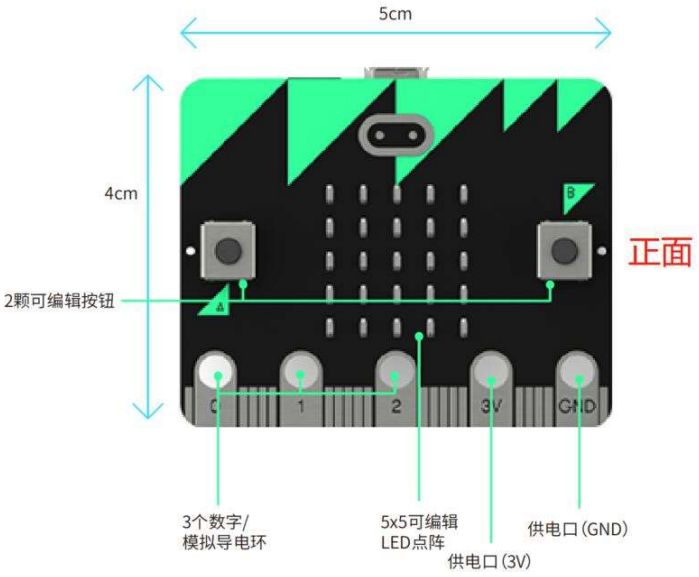
这里的“控制器”，也叫“智能终端”，我们可以简单理解为具有存储、控制等多种功能的智能设备，好比人的大脑，可以存储信息，控制人的行为。“执行器”也叫“输出单元”，就像人的四肢一样，在大脑的控制下可以行走、跳舞，做各种不同的动作。所谓的“传感器”，也可称为“输入单元”，它是一种检测装置，能够感知物体的信息，就像人的五官，可以感受周围的声音、光线等环境信息，并将感受到的信息告诉我们的大脑。

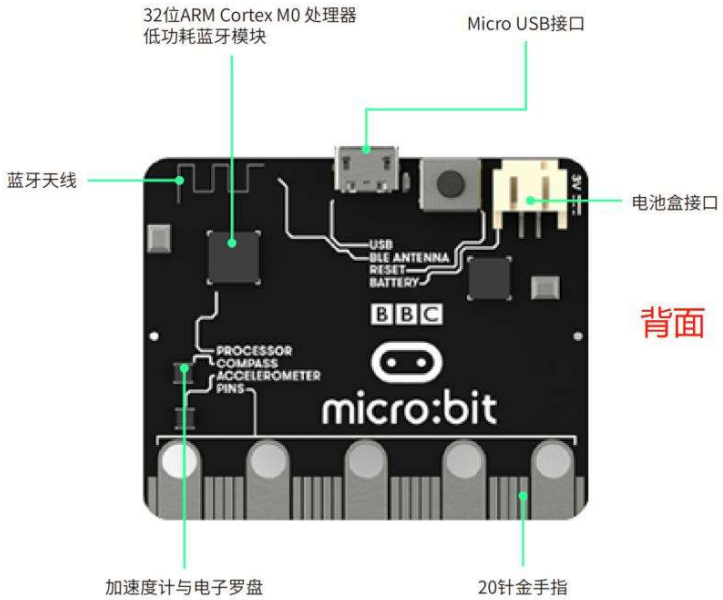
接下来我们将详细了解一下以下内容：

一、什么是 micro:bit

Micro:bit 是一款专为青少年编程教育设计的微型电脑，由英国的非盈利组织 micro:bit 基金会在全球范围内运营推广。它能够轻松胜任各种编程相关的教学与开发场景，如编写电子游戏、声光互动、机器人控制、科学实验、可穿戴装置开发等。

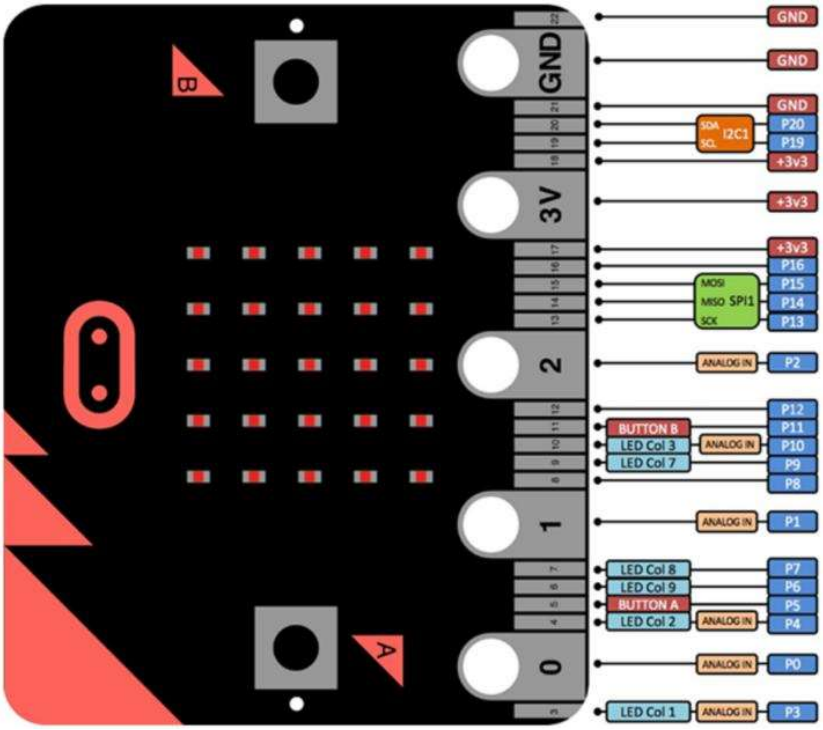
Micro:bit 作为智能终端设备，其板子本身内置了诸多元器件，如点阵屏、按钮、加速度计、电子罗盘、金手指等，可用于实现相应的功能，具体如下图，



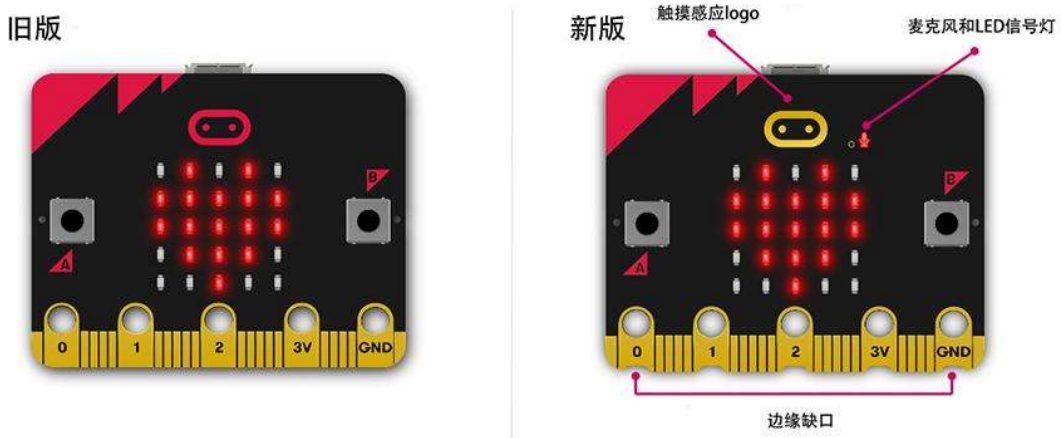


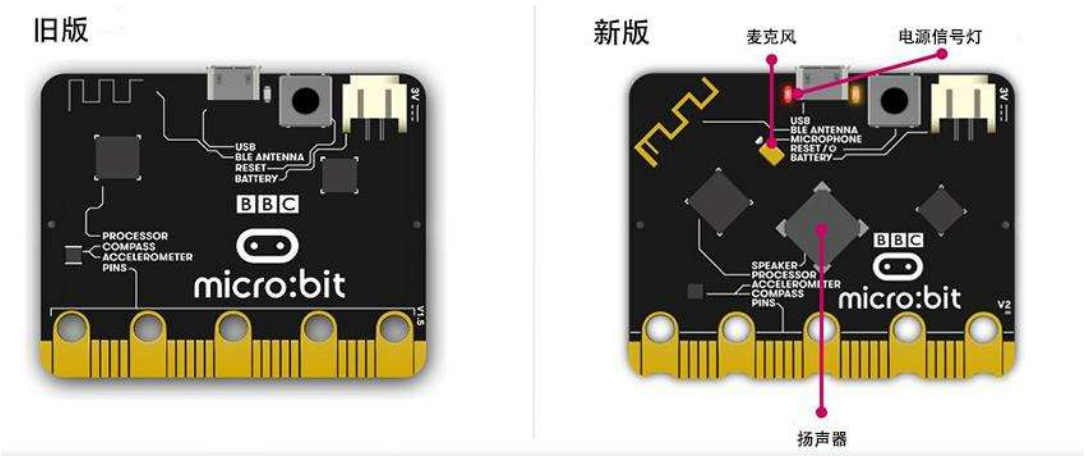
上两幅图分别是 micro:bit 的正反面，图中有标识的部分为常用的板载元器件。而要想控制 micro:bit 实现板载元器件的各种功能则首先需要给其供电。micro:bit 可以通过两种方式供电，一种通过 USB 供电，另一种通过外接 3V DC 电源供电。通电后，“Micro USB 接口”左侧的电源指示灯和通信指示灯就会亮起，而在下载程序的过程中，其中的通信指示灯会不停闪烁。

同时，micro:bit 还有很多扩展引脚，可通过外接扩展板进而与更多硬件设备相连接，实现读取传感器数据，控制执行器运行等功能。具体引脚如下，



注：micro:bit 有 V1 旧版、V2 新版两个版本，差别如下图。本套教程基于 V1 版编写，但所有程序同时兼容两个版本！





二、什么是 Mind+

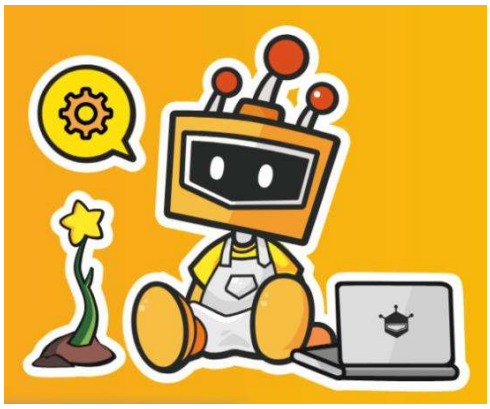
Micro:bit 板子和电脑间，我们可以用 USB 线这一硬件构建物理连接，但仅仅做到这一步就好比是买来了各式各样的硬件、组装好了电脑，但是没有软件无法使用这些硬件。那么要如何建立这两者之间信息的连接呢？

答案便是 Mind+！它为两者架起了虚拟的桥梁，从而实现代码的烧录、串口的连接，实时数据流的传输等功能。

Mind+是一款拥有自主知识产权的国产青少年编程软件，集成各种主流主控板及上百种开源硬件，支持人工智能（AI）与物联网（IoT）功能，既可以拖动图形化积木编程，也可以使用 Python/C/C++等高级编程语言，让大家轻松体验创造的乐趣。

注：本教程基于 Mind+V1.7.1 RC2.0 编写。

下载地址：http://Mindplus.cc/



三、什么是 pinpong 库

通过 Mind+软件我们可以使计算机与 micro:bit 连接起来，但如何才能控制 micro:bit 等硬件设备运行起来呢？

这里我们需要使用 pinpong 库。pinpong 库是一套控制开源硬件主控板的 Python 库，基于 Firmata 协议，5 分钟即可让你上手使用 Python 控制开源硬件。

借助于 pinpong 库，直接用 Python 代码就能给各种常见的开源硬件编程。其原理是给开源硬件烧录一个特定的固件，使开源硬件可以通过串口与电脑通讯，执行各种命令。

Pinpong 库的名称由“Pin”和“Pong”组成，“Pin”指引脚，“pinpong”为“乒乓球”的谐音，指信号的往复。

Pinpong 库的设计，是为了让开发者在开发过程中不用被繁杂的硬件型号束缚，而将重点转移到软件的实现。哪怕程序编写初期用其他板子开发，部署时改成了 micro:bit 板子，只要修改一下硬件的参数就能正常运行，实现“一次编写处处运行”。

注：关于 micro:bit 板子、Mind+软件及 pinpong 库的具体使用，我们将在后续的教程中再详细介绍。

第一课、趣味数字电导仪

一、实践情境

雷雨天气，我们常常被要求在路上行走时尽量远离电线杆和积水地段，这是因为水和电线能都够导电，而雷电和大风易使电线落入水中进而使积水带电引起触电危险。那么他们的导电能力如何呢？在这节课上，我们将设计一个数字导电仪，来对生活一些常见物质的导电能力探一探究竟！

二、实践目标

本实践项目运用 micro:bit 作为智能终端，通过土壤湿度传感器采集数据，来探究比较，水与铜片的导电性能强弱。

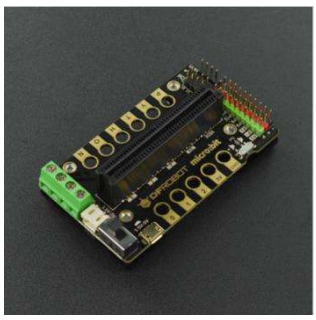
三、知识目标

1、掌握运用 micro:bit 作为智能终端，通过 Python 编程采集传感器数据的方法；
2、掌握通过 Python 编程对数据进行可视化处理的方法；
3、认识 micro:bit 开发板与 I/O 扩展板，掌握两者的连接方法；
4、认识土壤湿度传感器，了解土壤湿度传感器的工作原理和使用。

四、实践准备

硬件清单：

micro:bit 开发板 x1



I/O 扩展板 x1

Type-C&Micro 二合一USB 线x1

土壤湿度传感器及模拟连接线 x1

软件使用： Mind+编程软件 x1

其他:

1、一杯土壤（湿润）

2、一杯水（常温）

3、一块铜片（铜质材料即可）

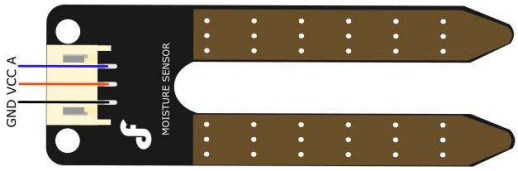
知识链接

土壤湿度传感器

简介： 这是一个简易的水分传感器，可用于检测土壤的水分。当土壤缺水时，传感器输出值将减小，反之将增大。传感器表面做了镀金处理，可以延长它的使用寿命。

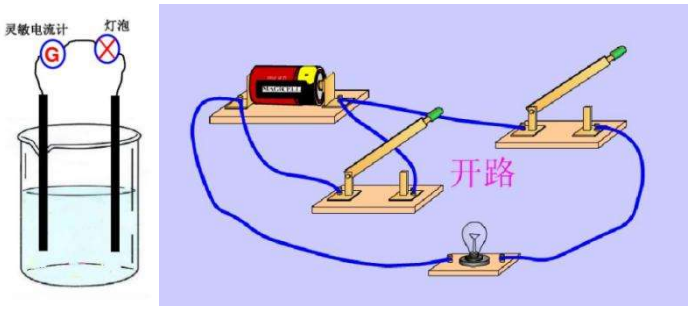
引脚说明：

标号	名称	功能描述
1	GND	电源负极
2	VCC	电源正极
3	A	信号线(模拟)



原理介绍：

土壤湿度传感器的两根金属条相当于电路中的开关，在提供电源的情况下，将金属条插入水中，金属条与水就能形成一条导通的电路，金属条之间亦会有电流流过，相当于开关被按下，电路导通，小灯亮起。



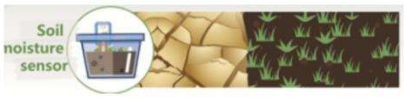
也正因此，我们选择土壤湿度传感器来作导电能力测试。

而所谓的检测土壤湿度，实际上就是利用水能够导电的特性来检测此时两块极板间的电流。水分越多，极板间的电流也就越大，导电能力也就越强，检测到的数值也就越大。

使用说明：

土壤湿度传感器在使用时需要将两金属条充分插入土壤中，并等待数秒直至数据稳定后，通过显示屏模块或软件可以读取土壤湿度对应的数值。

土壤湿度数值参考



0-300	300-700	700-900	900-1000
干燥	湿润	充足	过量

典型电压值（测试平台：10位AD，基准电压5V）：

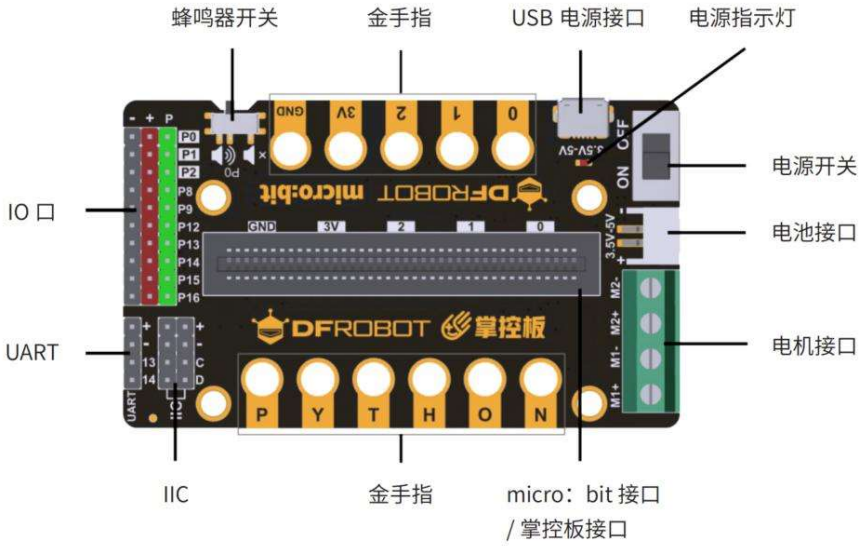
- 0 ~300 : 干燥土壤
- 300~700 : 湿润土壤
- 700~950 : 放到水中

Tips1：不同土壤介质的测量值会存在差异。土壤中的水分存在分布不均的情况，数据仅能代表局部湿度。

Tips2：传感器顶部的塑料部分不防水，请勿将传感器整个埋入土中。

I/O 扩展板

简介：这里的“I、O”分别是“input”输入和“output”输出的缩写，I/O 扩展板的作用是将 micro:bit 上的 IO 口以引脚口的形式引出，方便我们接入传感器、执行器等模块。IO 口与 micro:bit 上的接口一一对应。关于 I/O 扩展板的详细介绍可见附录 1。后续，我们可简称其为“扩展板”。

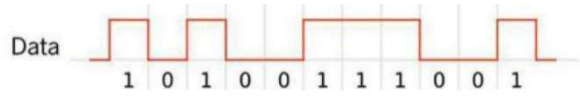


蜂鸣器

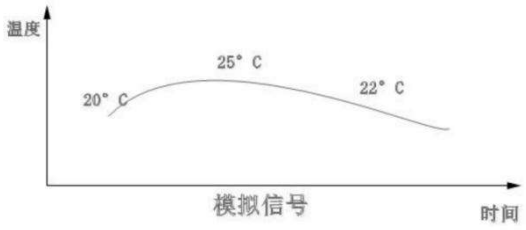
简介：蜂鸣器是一种能够发出类似蜂鸣声音的器件。这里的蜂鸣器已被内置在 I/O 扩展板上。

数字信号与模拟信号

数字信号：数字信号指自变量是离散的、因变量也是离散的信号，这种信号的自变量用整数表示，因变量用有限数字中的一个数字来表示。在计算机中，数字信号的大小常用有限位的二进制数表示。



模拟信号：模拟信号是指用连续变化的物理量表示的信息，其信号的幅度，或频率，或相位随时间作连续变化，或在一段连续的时间间隔内，其代表信息的特征量可以在任意瞬间呈现为任意数值的信号。



光看概念感觉特别抽象，那么我们用生活中的实例来理解，比如我们生活中常见的灯，只有开和关两个状态，非开即关。那么对灯来说开和关就是数字信号。再想想家中如果有一个温度计的话，温度变化是一个连续变化的的数值，并不能用某个特殊的状态来表示，温度的变化就是模拟信号。

五、实践过程

在本项目中，我们将利用土壤湿度传感器的工作原理，分两步设计一个趣味数字电导仪，来探究一下水与铜片的导电性能强弱。

1、实时检测土壤湿度进而检测水、铜导电性能。

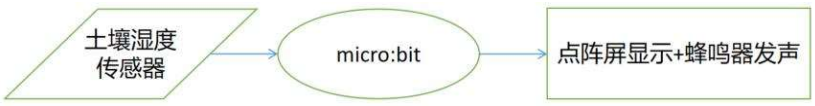
2、对数据进行可视化处理。

任务 1：土壤湿度实时检测

在这个任务中，我们将在 20℃的环境室温下，对杯子中的土壤进行湿度检测。为此，我们将分四步进行。首先是分析设计，接着进行硬件的搭建，之后通过软件编写程序，最后对实验进行调试修改。

1.分析设计

本实践任务主要是通过扩展板将土壤湿度传感器与 micro:bit 开发板相连接，实时获取土壤的湿度，并将检测到的数据信息显示在点阵屏上，同时，当数值大于一定值时，蜂鸣器发声以作提示。

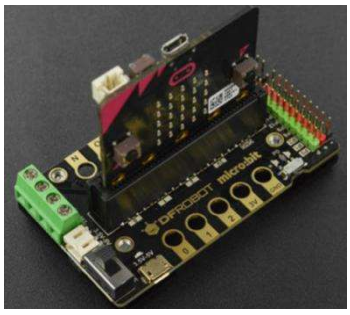


2.硬件搭建

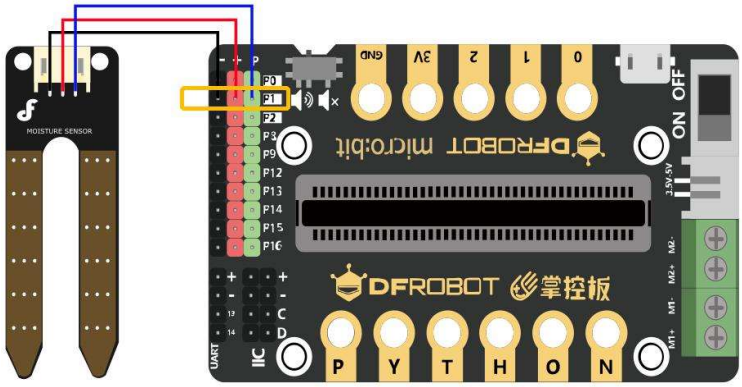
由于 micro:bit 上有很多无法直接使用的扩展引脚，因而我们需要借助 I/O 扩展板可以将这些划分的很细的引脚引出来使用，从而支持更多输入输出设备拓展。

这里，我们将分步骤使土壤湿度传感器先通过扩展板连接上 micro:bit，再将 micro:bit 接入计算机。

STEP1：将 micro:bit 开发板插入 I/O 扩展板，注意正反，LED 点阵屏朝扩展板的“micro:bit”图标方向



STEP2：通过传感器连接线将土壤湿度传感器连在扩展板的 P1 端口。其中，红线和黑线分别对应电源正极和负极，蓝线对应信号线，用来传输信号。



STEP3：通过 USB 连接线将 micro:bit 接到计算机。

3.软件编写

在连接好各个硬件之后，我们还无法使他们实现功能。这个时候我们需要给硬件编写程序来控制他们运行。而在首次编程时，我们将先对软件进行设置，之后再编写 Python 程序来检测土壤湿度。

软件设置：

在 Mind+软件中，我们需要手动创建项目文件、Python 文件并安装一些库文件以便编写程序。项目文件对应我们的项目，程序文件中存储的是编写的 Python 代码，在一个项目文件中，可以有一个或多个 Python 程序文件，而库文件的安装则有助于我们便捷地编写程序。具体操作流程如下，

STEP1：创建与保存项目文件

（1）启动 Mind+软件，点击右上角的“Python 模式”，

（2）点击左上角“代码”选择编程方式，

之后我们会看到下面这样的屏幕，右侧是文件目录区，左侧分别是代码编写区和终端区。

（3）创建完成后，点击左上方项目菜单中的“另存项目”，

（4）在弹出的界面中选择保存位置，输入文件名“m 趣味数字电导仪”，保存类型“Mind+”

Tips：文件名称可自取，上述文件名中的“m”指代“micro:bit”。

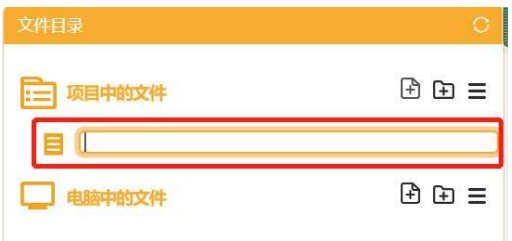
STEP2：创建与保存 Python 文件

（1）点击“文件系统”，

之后会显示出文件目录，也就是 Python 程序文件所保存的区域，

（2）点击“项目中的文件”栏中的加号，

之后，会弹出一个空白输入框，



（3）在弹出的输入框中写入“任务 1 土壤湿度检测.py”并回车确认，如下左图，

这样我们就成功创建一个名为“任务 1 土壤湿度检测.py”的 Python 程序文件了，

Tips：Python 程序文件一定要以“.py”格式结尾才能进行编程，否则无法打开。

（4）打开文件

创建完成后，我们双击“土壤湿度检测.py”文件即可对该文件在程序编写区内进行编程，图示如下。

STEP3：安装 pinpong 库

pinpong 库是一套用来控制开源硬件的 Python 库。

（1）点击库管理

（2）安装 pinpong 库

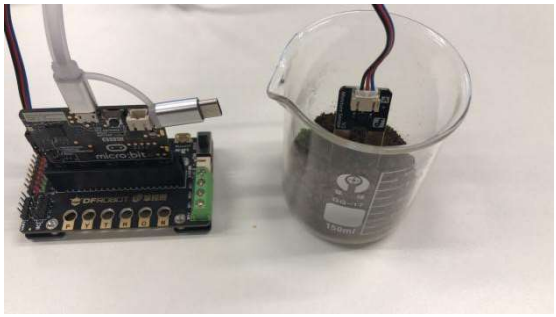
程序编写、运行及回顾：

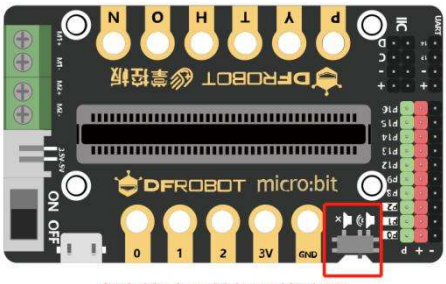
STEP1：编写 Python 程序

```
import time#导入时间库
from pinpong.board import Board,Pin,ADC#从 pinpong.board 包中导入 Board,Pin,ADC 模块
from pinpong.extension.microbit import *#导入 pinpong.extension.microbit 包中所有模块
Board('microbit').begin()#初始化，选择板型和端口号，不输入端口号则进行自动识别
adc0 = Pin(Pin.P1, Pin.ANALOG) #配置 P1 引脚为模拟输入模式
while True:#永久循环
    A1 = adc0.read_analog()  #读取模拟信号数值
    if A1 >5:
        music.play_buzzer(Pin.P0, "F/F3", 1)#P0 口，播放声音，音符"F/F3"，节拍 1
    display.scroll(A1)#点阵屏显示 A1 的值
    print(A1)#打印显示在终端
    time.sleep(5)#延时 5 秒,即每五秒检测一次
```

STEP2：运行程序并观察效果

（1）在运行程序之前，我们需要先手握土壤湿度传感器未镀金部分，将其插入土壤中，并打开 I/O 扩展板的蜂鸣器开关。





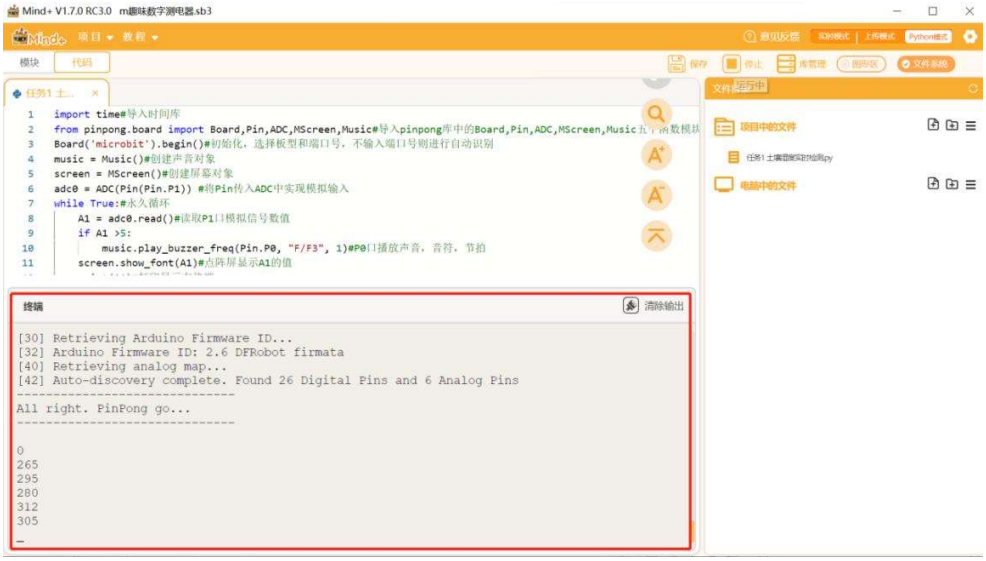
向右拨动开关打开蜂鸣器

（2）点击右上方的运行按钮

首次运行后，Mind+会自动给 micro:bit 板子烧录一段特定的程序，我们称之为固件。通过给 micro:bit 板子烧录固件能够将硬件设备与软件连接起来,继而依靠软件程序控制硬件运行。

（3）观察效果

我们可以看到数值 0 呈现在 micro:bit 点阵屏上，并且在软件终端界面中同样显示一个数值 0。之后，每隔 5 秒重复一次。这里的“0”，就是输出的电信号对应的模拟值。初始值为 0 是因为传感器执行检测的程序需要一定的时间，并不表示土壤中无水分。



STEP3：回顾程序

我们用 Python 代码实现了项目功能，现在让我们一起来回顾一下，看看程序是如何工作的吧。

1、导入库

```
from pinpong.board import Board,Pin,ADC#导入 pinpong.board 包中 Board,Pin,ADC 模块
from pinpong.extension.microbit import *#导入 pinpong.extension.microbit 包中所有模块
```

在进行土壤湿度实时检测时，我们用到了 micro:bit 主板上的模拟引脚——P1 引脚，而在 pinpong 库的“pinpong.board”包中已经提供了相应的 Board 类、Pin 类以及 ADC 类等来实现相关功能，因此在编程时，我们需要从中导入这些模块并在后续创建他们的实例化对象，以使用各自的相关功能。
同样的，为了能够实现 micro:bit 板子上各元器件的功能，我们也需要导入“pinpong.extension.microbit”这个包中的所以模块。

2、初始化主控板

```
Board('microbit').begin()#初始化，选择板型和端口号，不输入端口号则进行自动识别
```

“begin”的意思是开始，这是初始化主控板语句，通过它可以确定主控板类型以及端口号。在 Board(" ") 中输入主控板型号，在 begin() 中输入主控板端口号，不输入端口号则进行自动识别。这里我们一般只使用一块 micro:bit 主控板，所以不用识别端口号。

3、引脚初始化

```
adc0 = Pin(Pin.P1, Pin.ANALOG) #配置 P1 引脚为模拟输入模式
```

为了能够使连在 P1 引脚上的传感器将检测到的电信号转化为模拟量进行读取，我们需要配置 P1 引脚为模拟输入模式，定义其为变量“adc0”。

4、读取模拟量

```
A1 = adc0.read_analog()  #读取模拟信号数值
```

“read”的意思是读取，“adc0”引脚为 P1 引脚，接了土壤湿度传感器，所以变量“A1”为土壤湿度传感器的反馈值。
整体表示通过“read_analog()”方法来获得引脚“adc0”的模拟值，然后用变量“A1”来存储该值。

5、播放声音

```
music.play_buzzer(Pin.P0, "F/F3", 1)#P0 口，播放声音，音符"F/F3"，节拍 1
```

“play_buzzer()”是蜂鸣器发声的意思，“Pin.P0”指的是默认的 p0 口的蜂鸣器；“F/F3”指的是音符；“1”指的是节拍，可以改变数值进而变化声音的持续时间。
整体表示通过“play_buzzer()”方法使 P0 引脚上的蜂鸣器以 1 节拍来播放音符 F/F3。

6、屏幕显示

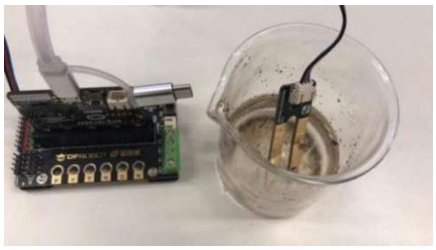
```
display.scroll(A1)#点阵屏显示 A1 的值
```

“scroll”是显示字符的意思，整体表示通过“scroll()”方法来将读取到的 A1 值显示在点阵屏上。

4.调试修改

通过上述实验，我们已经成功实现了每隔五秒对土壤的湿度进行一次检测。并且我们已经知道，传感器之所以能检测到电流，归根结底是因为土壤中的水分能够导电，使得土壤与传感器之间形成了电路。因此，倘若将土壤替换成同样能够导电的物质，例如铜片，我们依旧能够检测此时的电流强度，也就能以此来比较一下不同物质的导电能力到底孰强孰弱了。

STEP1：将土壤湿度传感器的镀金端插入水中，观察数据的变化。

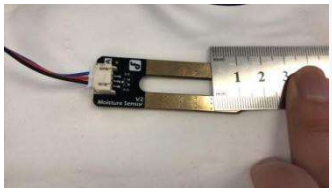


STEP2：自建一个 Excel 文件，将上述数据记录下来，并以 csv 格式保存。如下图，

	A	B
1	时间（秒	水
2		
3		
4		
5		
6		
7		

时间（秒）	水
0	0
5	755
10	746
15	741
20	739
25	736

STEP3：将土壤湿度传感器金属端压在铜片上，使其充分接触，观察数据的变化并记录在上述表格中。



	A	B	C
1	时间（秒	水	铜
2			
3			
4			
5			
6			
7			
8			
9			
10			

	A	B	C
1	时间（秒）	水	铜
2	0	0	0
3	5	755	811
4	10	746	808
5	15	741	813
6	20	739	810
7	25	736	810
8	30	733	812
9	35	731	809
10	40	729	810

任务 2：数据可视化

由于记录的表格数据并不能非常直观得呈现水和铜的导电能力强弱，因此，在这个任务中，我们将对上述检测到的数值可视化处理，进而分析两者的导电能力强弱。这里，我们将数据以折线图形式呈现即可。

1.数据可视化与分析

软件设置：

STEP1：创建与保存 Python 文件

创建一个 Python 程序文件“任务 2 数据可视化.py”，双击打开。

STEP2：安装库

（1）点击库管理

（2）安装 matplotlib 库

（3）安装 pandas 库

STEP3：导入数据文件（见附录 2）

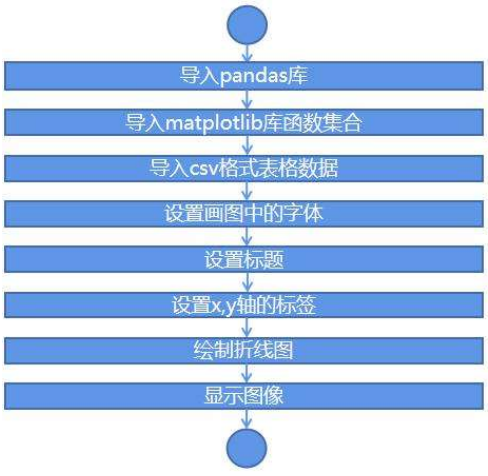
将自己记录的表格数据文件存放入 Python 程序文件所在文件夹，使其在同一目录下。

程序分析、编写及运行：

STEP1：分析程序

我们知道常见的数据采集方式有传感器采集和网络采集两种，这里我们已借助土壤湿度传感器，采集到了水与铜片的数值并记录在表格文件中，而为了便于对数据进行读取和分析，我们依旧通过编写程序使其以图表形式呈现出来。（程序中使用的参考数据表格见附录 2）

这里，我们将采用顺序结构的方式编写程序。具体流程如下，



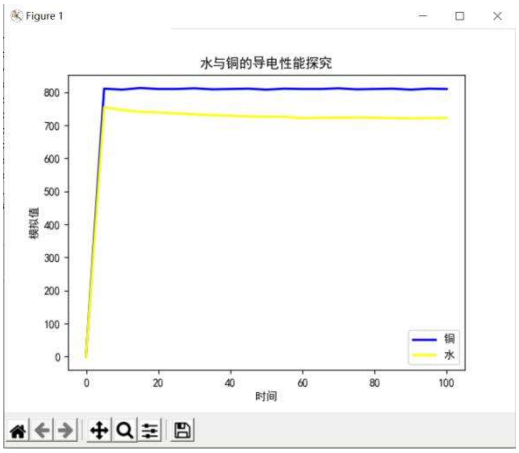
STEP2：编写 Python 程序

```
# -*- coding: utf-8 -*-
import pandas as pd#导入 pandas 库并用 pd 来表示
import matplotlib.pyplot as plt#导入 matplotlib 库中的 pyplot 绘图模块
data = pd.read_csv("数据表.csv",encoding='gb18030')#导入 csv 格式表格数据
```

```
plt.rcParams['font.sans-serif'] = ['SimHei']#设置中文字体
plt.title("水与铜的导电能力探究")#绘制标题
plt.ylabel("电压值")#绘制 x 轴标签
plt.xlabel("时间")#绘制 y 轴标签
x_data1=data.iloc[:,0]#选取表格第一列并以此作为折线 1 的 x 轴数据
y_data1=data.iloc[:,1]#选取表格第二列并以此作为折线 1 的 y 轴数据
x_data2=data.iloc[:,0]#选取表格第一列并以此作为折线 2 的 x 轴数据
y_data2=data.iloc[:,2]#选取表格第三列并以此作为折线 2 的 y 轴数据
plt.plot(x_data2,y_data2,color='blue',label="铜片",linewidth=2.0)#绘制折线，并设置折线的颜色、标签、线宽
plt.plot(x_data1,y_data1,color='yellow',label="水",linewidth=2.0)#绘制折线，并设置折线的颜色、标签、线宽
plt.legend()#显示图例（标签）
plt.show()#显示图像
```

STEP3：运行程序并分析数据

（1）运行程序，我们看到 csv 表格中的数据以图表的形式呈现了出来。



（2）观察上图，我们可以清楚地发现，在 20℃的温度下，随着检测时间的增加，水与铜片的模拟值逐渐趋于稳定，并且金属铜的模拟值大于水的模拟值。因此我们可以得出结论，在相同的条件下，铜的导电能力强于水。

六、巩固提高

1.项目小结

本节课我们以 micro:bit、扩展板及土壤湿度传感器为器件自制了一个电导仪，探究分析了水与铜的导电性能强弱。首先通过土壤湿度传感器检测土壤的湿度模拟值，之后以此方式采集水和铜片的数据，最后将测得的数据以折线图的形式可视化出来并加以分析。

2.项目拓展

经过上述的实验探究，我们已知在相同的条件下，水的导电能力是弱于金属铜的。这也变相解释了为何生活中常见的一些电缆线里面的材料往往是铜丝。不仅因为铜这种金属易得，更因为它的导电能力强。

那么对于水来说，是否可以通过变化外在的一些条件，使其导电能力增加呢？比如，改变水温。让我们按下列步骤完成第一个拓展探究吧!

STEP1：在烧杯中倒入 30ml 接近 100℃的开水，将传感器金属端轻轻插入其中，观察结果并记录在下表，

STEP2：将开水替换为 50℃左右的温水，重复上述操作，观察结果并作记录，

STEP3：将温水替换为接近 0℃的冰水，重复上述操作，观察结果并作记录。

	100℃开水	50℃温水	0℃冰水
均值			

想一想，上述结果形成的原因是怎样的呢？!而除了水和铜之外，宇宙中还有什么自然物质或人造物质能够导电呢？拿上我们的数字电导仪，对生活中常见的一些物质进行一次检测吧！看看他们的导电性能如何呢？！（如：木头、石头、硬币）

附录

附录 1

拓展阅读

I/O 扩展板

详细简介：I/O 扩展板能完全兼容 micro:bit 和掌控板两种主板。正面插入掌控，反面插入 micro:bit。引出 10 路数字/模拟 3Pin 口，两路 IIC 口以及一路 UART 口；板载两路电机驱动，且不占用额外引脚；板载 PH2.0 及 microUSB 两种供电口，既可以通过 usb 线也可以通过电池盒或者锂电池供电，供电电压 3.5-5V，板载开关，可以开关外接供电电源；板载一个高品质蜂鸣器，且带有开关控制，可以随时关闭蜂鸣器；引出了 9 个鳄鱼夹接口；分别兼容掌控的触控金手指和 micro:bit 金手指。扩展板兼容乐高尺寸孔位，可以与乐高进行拼插结合。

供电方式：

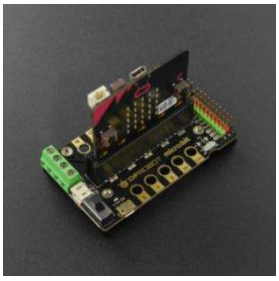
主板直接供电： 采用主板的 USB 或者电源口供电，此时，可以使用扩展板的各种扩展口及蜂鸣器。因主板驱动电流有限，此模式下无法使用电机驱动。

USB 供电口供电： 使用电脑 USB 口供电或者充电宝，或者手机充电头供电。接口为 microUSB。此模式下，扩展板的所有功能均可使用。

外接电池盒供电： 外接 PH2.0 接口输入电压为 3.5~5V，使用 PH2.0 接口的 3 节干电池盒或者 3.7V 锂电池，均可。此模式下，扩展板的所有功能均可使用。

Micro:bit 与扩展板连接方法

microbit 主板的 LED 点阵的那一面，对着 microbit 字样的那个方向插入，如图所示：



此时，扩展板支持的功能如下：

- 2 路电机驱动
- 0 1 2 3V GND 金手指
- USB 供电和 PH2.0 外接供电口供电及电源开关
- 蜂鸣器及蜂鸣器开关

Tips： microbit 下不支持 P Y T H O N 这 6 个触摸金手指。 蜂鸣器占用的 P0 口，如果要正常使用金手指 P0，请将蜂鸣器关闭。

附录 2

扫描首页项目包二维码可获取数据文件

第二课、简易气象站

一、实践情境

当我们去旅游景区的时候，常常在显示屏上看到或者在广播听到空气温度、空气湿度、负氧离子含量等气象信息数据。这些数据几乎都来自景区气象站的实时监测。然而，景区的气象信息都是针对室外的，那当我们待在室内时，如何才能实时获取室内的环境信息呢？如果你也有这些困惑，就请跟着我们来完成一个室内的简易气象站吧！

二、实践目标

本实践项目运用 micro:bit 作为智能终端，通过 DHT11 温湿度传感器采集数据，来制作一个简易的气象站。

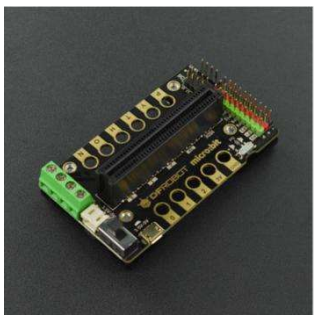
三、知识目标

1、 认识 DHT11 温湿度传感器，简单了解温湿度传感器的工作原理。
2、 认识 micro:bit 上的点阵屏，掌握其基本使用。
3、 使用 micro:bit 作为智能终端，掌握通过编写 Python 程序检测环境温湿度的方法。

四、实践准备

硬件清单：

micro:bit 开发板 x1



I/O 扩展板 x1

Type-C&Micro 二合一USB 线x1

DHT11 温湿度传感器及连接线 x1

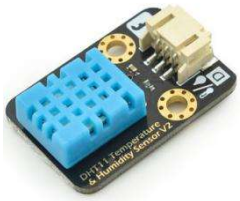
软件使用： Mind+编程软件 x1

知识链接

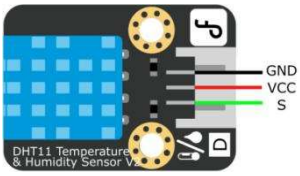
DHT11 数字温湿度传感器

简介：

DHT11 数字温湿度传感器包括一个电阻式感湿元件和一个 NTC 测温元件，可用来采集环境温度和湿度。



引脚说明：

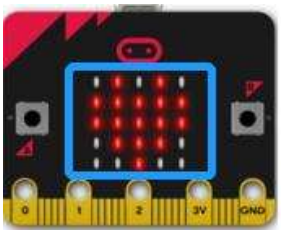


标号	名称	功能描述
1	GND	电源负极
2	VCC	电源正极
3	S	信号线（数字）

点阵屏

简介：

Micro:bit 板载的显示屏由 5 行 5 列总共 25 颗 LED 构成，因此被称为点阵屏，在上面我们可以实现一些简单的图像、字符等效果。



内置 LED 阵列说明		
功能举例说明		控制 LED 阵列显示内容
方法说明	.show("Image.HEART")	显示 HEART 图案
	.scroll("string")	显示字符 string
	.set_pixel(x,y)	点亮坐标 x,y 的灯
	.off_pixel(x,y)	熄灭坐标 x,y 的灯
	.set_brightness(value)	设置灯的亮度
	.clear()	关闭所有阵列

五、实践过程

在本项目中，我们将利用 DHT11 温湿度传感器，设计一个简易的气象站，来实时检测环境的温湿度。

1、实时监测环境温湿度

2、对温湿度数据进行可视化处理

Tips：下面课程中的数据是在夏季室温 30℃左右的环境下测量而得，不同季节测得的数据并不相同。

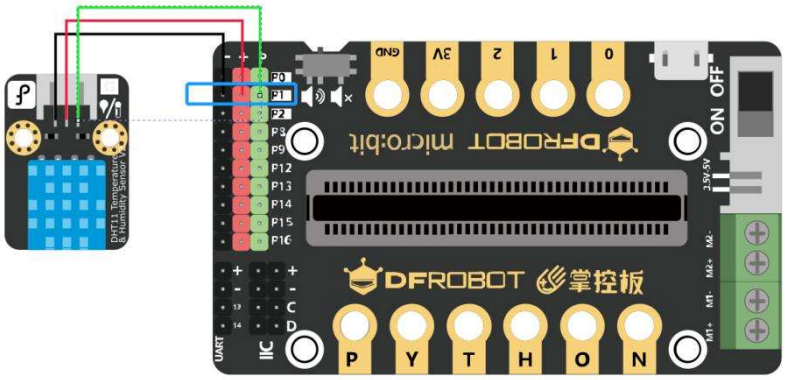
任务 1：环境温湿度实时检测

1.分析设计

在本任务中，我们将对室内的环境温湿度进行实时监测。这里，我们可通过扩展板将 DHT11 温湿度传感器与 micro:bit 开发板相连接，实时检测环境的数据并将其显示在点阵屏和软件终端上。

2.硬件搭建

STEP1：通过传感器连接线将 DHT11 温湿度传感器连在扩展板的 P1 端口。



STEP2：将 micro:bit 开发板插入 I/O 扩展板，并通过 USB 连接线将 micro:bit 接到计算机。

Tips：使用 I/O 扩展板时必然需要将开发板插入，而 micro:bit 与计算机的连接也始终需要 usb 线，因此，后续教程中我们将不再赘述该步骤。

3.软件编写

软件设置：

STEP1：创建与保存项目文件

启动 Mind+软件，选择“Python 模式”，另存项目并命名为“m 简易气象站”。

STEP2：创建与保存 Python 文件

创建一个 Python 程序文件“任务一.py”，双击打开。

Tips：由于之前已安装过 pinpong 库，这里我们无需再次安装。

程序编写、运行及回顾：

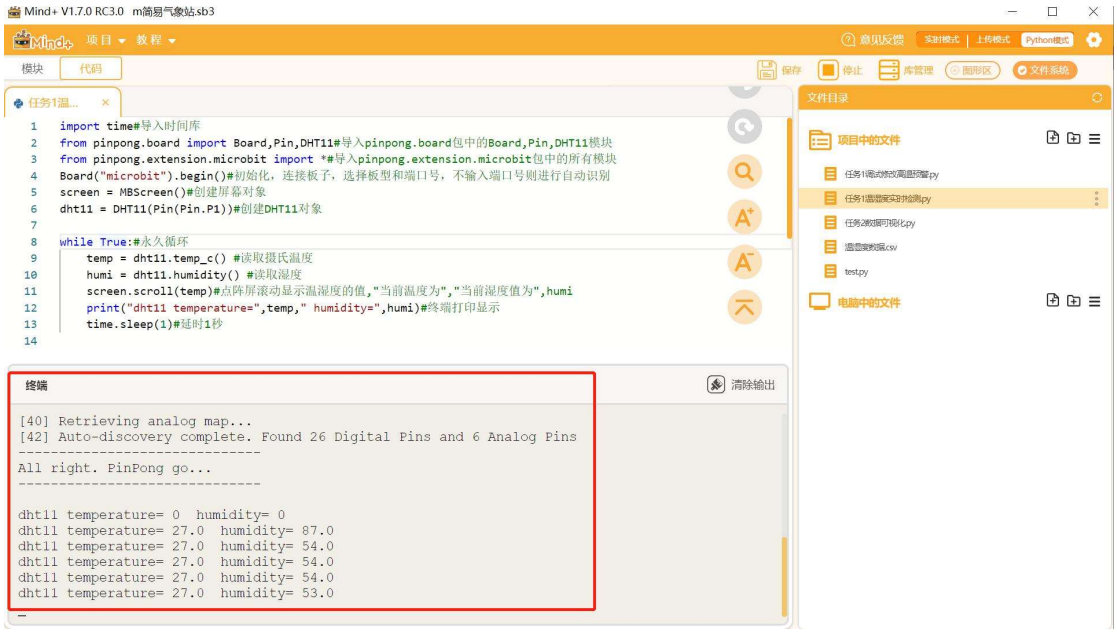
STEP1：编写 Python 程序

```
import time#导入时间库
from pinpong.board import Board,Pin,DHT11#导入 pinpong.board 包中的 Board,Pin,DHT11 模块
from pinpong.extension.microbit import *#导入 pinpong.extension.microbit 包中的所有模块
Board("microbit").begin()#初始化，连接板子，选择板型和端口号，不输入端口号则进行自动识别
dht11 = DHT11(Pin(Pin.P1))#创建 DHT11 对象

while True:#永久循环
    temp = dht11.temp_c() #读取摄氏温度
    humi = dht11.humidity() #读取湿度
    display.scroll(temp)#点阵屏滚动显示温度的值
    display.scroll(humi)#点阵屏滚动显示湿度的值
    print("dht11 temperature=",temp," humidity=",humi)#终端打印显示
    time.sleep(1)#延时 1 秒
```

STEP2：运行程序并观察效果

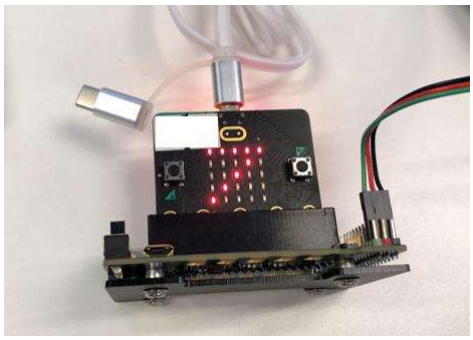
（1）运行程序，观察软件终端

点击 Mind+软件右上方的运行按钮，我们可以看到在终端中显示测得的温度和湿度数据。



（2）观察点阵屏

同时，观察点阵屏，我们可以发现测得的数据同样在屏幕上滚动显示。



STEP3：回顾程序

1、导入库

```
from pinpong.board import Board,Pin,DHT11#导入 pinpong.board 包中的 Board,Pin,DHT11 模块
```

在进行环境温度实时检测时，我们用到了 micro:bit 主板上的 P1 引脚和 DHT11 温湿度传感器，在 pinpong 库中已经提供了相应的 Board 类、Pin 类以及 DHT11 类来实现和他们相关的功能，因此在编程时，我们需要从“pinpong.board”**导入这些模块**并在后续创建他们的实例化对象，来使用各自的相关功能。

2、创建实例化对象

```
dht11 = DHT11(Pin(Pin.P1))#创建 DHT11 对象
```

为了能够使用 DHT11 类中的相应的函数功能，我们需要先创建一个实例化对象。

3、读取数值

```
temp = dht11.temp_c() #读取摄氏温度
humi = dht11.humidity() #读取湿度
```

使用 DHT11 需先导入 DHT11 模块，实例化对象后即可使用“temp_c()”和“humidity()”两个方法分别来读取温度和读取湿度。

4.调试修改

至此，我们成功制作了一个简易的气象站，实现了对室内环境温湿度的实时监测。并且我们发现，当气温达到 30℃，或相对湿度达到 50%时，我们就会感到闷热，这时候就可以打开空调解暑降温啦。而为了能及时打开空调，我们可进一步优化气象站，为其添加高温预警功能，实现当温度或湿度达到 30℃和 50%时，蜂鸣器发出警报声。

STEP1：编写程序

```
import time#导入时间库
from pinpong.board import Board,Pin,DHT11#导入 pinpong.board 包中的 Board,Pin,DHT11 模块
from pinpong.extension.microbit import *#导入 pinpong.extension.microbit 包中的所有模块
Board("microbit").begin()#初始化，连接板子，选择板型和端口号，不输入端口号则进行自动识别
dht11 = DHT11(Pin(Pin.P1))#创建 DHT11 对象
while True:#永久循环
    temp = dht11.temp_c() #读取摄氏温度
    humi = dht11.humidity() #读取湿度
    display.scroll(temp)#点阵屏滚动显示温度的值
    display.scroll(humi)#点阵屏滚动显示湿度的值
    time.sleep(5)#延时 5 秒
    print("dht11 temperature=",temp," humidity=",humi)#终端打印显示
    if temp >30 or humi >50:#条件
        music.play_buzzer(Pin.P0, "F/F3", 1)#P0 口，播放声音，音符"F/F3"，节拍 1
    time.sleep(1)#延时 1 秒
```

任务 2：数据可视化

至此，我们的气象站不仅能实时监测温湿度，亦能在高温时发出预警。但我们知道，真正的气象站不仅仅能实时检测，而且能对检测到的数据进行可视化处理，进而发现数据的特点以作深入研究，比如统计在不开空调或取暖器的情况下一天中室内温度低于 0℃的时长等等。因此，接下来，我们将加长实验时间，检测并记录夏季的某个上午的温湿度，并对数据进行可视化处理，进而探究一下当天上午，室内温湿度的变化情况。这里，我们将数据以折线图形式呈现即可。

1.数据可视化与分析

数据记录：

STEP1：自建一个 Excel 文件，将早上 6-11 点中，每隔十分钟的温度和湿度数据记录下来，并以 csv 格式

保存。

时间	温度	相对湿度
0	20	63
1	20	63
2	21	63
3	21	65
4	22	66
5	22	67
6	23	68
7	24	68
8	24	68
9	24	67
10	25	68

软件设置：

STEP1：创建与保存 Python 文件

创建一个 Python 程序文件“任务 2 数据可视化.py”，双击打开。

STEP2：导入数据文件

将自己测得并记录数据的表格文件存放入 Python 程序文件所在文件夹，使其在同一目录下。

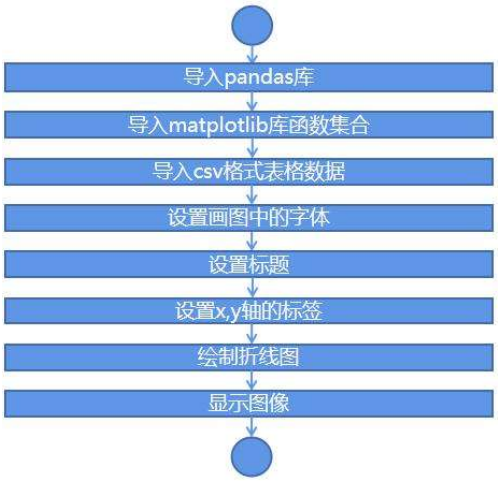
Tips：这里我们提供了一组数据（见附录 1）以作案例分析。

程序分析、编写并运行:

STEP1：分析程序

这里我们已借助 DHT11 温湿度传感器，采集到室内温度和湿度的数值并记录在了表格文件中，而为了便于对数据进行读取和分析，我们依旧通过编写程序使其以图表形式呈现出来。

整体上，我们将采用顺序结构的方式编写程序。具体流程如下，



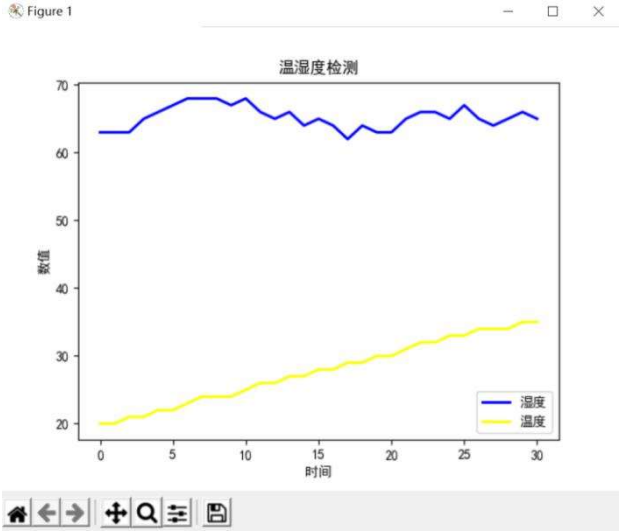
STEP2：编写 Python 程序

```
# -*- coding: utf-8 -*-
import pandas as pd#导入 pandas 库并用 pd 来表示
import matplotlib.pyplot as plt#导入 matplotlib 库中的 pyplot 绘图模块，并用 plt 表示
data = pd.read_csv("温湿度数据.csv",encoding='gb18030')#导入 csv 格式表格数据
```

```
plt.rcParams['font.sans-serif'] = ['SimHei']#设置中文字体
plt.title("温湿度检测")#绘制标题
plt.ylabel("数值")#绘制 x 轴标签
plt.xlabel("时间")#绘制 y 轴标签
x_data1=data.iloc[:,0]#选取表格第一列并以此作为折线 1 的 x 轴数据
y_data1=data.iloc[:,1]#选取表格第二列并以此作为折线 1 的 y 轴数据
x_data2=data.iloc[:,0]#选取表格第一列并以此作为折线 2 的 x 轴数据
y_data2=data.iloc[:,2]#选取表格第三列并以此作为折线 2 的 y 轴数据
plt.plot(x_data2,y_data2,color='blue',label="温度",linewidth=2.0)#绘制折线，并设置折线的颜色、标签、线宽
plt.plot(x_data1,y_data1,color='yellow',label="湿度",linewidth=2.0)#绘制折线，并设置折线的颜色、标签、线宽
plt.legend()#显示图例（标签）
plt.show()#显示图像
```

STEP3：运行程序并分析数据

（1）运行程序，我们看到 csv 表格中的数据以图表的形式呈现了出来。



（3）观察上图，我们可以清楚地发现，在上午时，随着时间的增加，室内温度值也在不断上升，而湿度值则相对趋于稳定。

2.调试修改

继续增加实验时间，看一看在一整天内环境温湿度是怎样变化的呢？

六、巩固提高

1.项目小结

本节课我们以 micro:bit、扩展板及 DHT11 温湿度传感器为器件自制了一个简易的气象站。首先通过

传感器实现了对室内环境温湿度的实时检测，之后添加了高温预警功能，最后将测得的数据以折线图的形式呈现后加以分析。

2.项目拓展

在闷热的夏季，我们可以借助气象站来高温预警，那如果是在寒冷的冬季，我们是否也能对低温进行警报呢？修改程序试一试吧！

附录

附录 1

扫首页项目包二维码可获取数据文件

第三课、自制通信设备（上）

一、实践情境

5G 元年的列车早已驶出，人工智能、大数据的浪潮还在涌动，云办公、云问诊成为防疫期间的热词。如今，物联网技术正处于时代发展的“风口”，其在安全防范、实时监测、追踪定位等领域都有广泛的应用。相较于传统的硬件设备，物联网技术使得各种硬件设备能够通过信息传输设备与互联网连接起来，继而进行信息的传递，以实现智能化识别与管理，为人们的日常生活带来便捷。本实践就以此为切入点，自制一个通信设备，实现智能终端借助物联网平台进行信息交换。

二、实践目标

本实践项目将以计算机为智能终端，自制一个基于 WiFi 的通信设备，实现与物联网平台的连接并向平台发送信息。

三、知识目标

1、了解物联网、MQTT 以及 SIoT
2、掌握搭建 SIoT 物联网平台的方法
3、掌握通过 Python 编程实现向 SIoT 物联网平台发送信息的方法
4、掌握从 SIoT 物联网平台网页端查看消息记录的方法

四、实践准备

软件使用： Mind+编程软件 x1、SIoT 系统软件 x1

Tips: 根据自己电脑的系统，对应下载 SIoT 软件压缩包。下载链接：http://Mindplus.dfrobot.com.cn/siot

知识链接

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。

物联网是互联网的一个延伸，互联网的终端是计算机（PC、服务器），而物联网的终端是硬件设备（手机、电脑、传感器等等）。

在物联网上，通过 MQTT 协议可以传输消息，每个人也可以应用电子标签将真实的物体上网联结，并以此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、设备等进行集中管理、控制，实现物和物相联。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。

MQTT

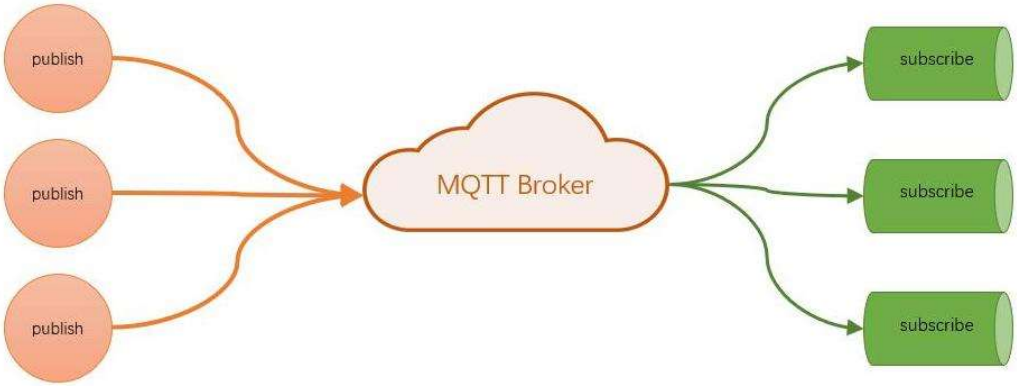
简介：

MQTT 是物联网中用于传输消息的一种协议（Message Queuing Telemetry Transport，消息队列遥测传输协议），它是基于客户端-服务器的消息发布/订阅传输协议。

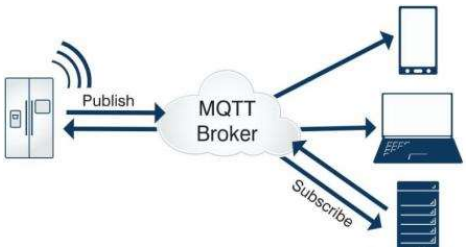
MQTT 协议是轻量、简单、开放和易于实现的，这些特点使它适用范围非常广泛。

MQTT 协议实现方式

MQTT 系统由与服务器通信的客户端组成，服务器就是我们要搭建的物联网平台的载体。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服务器。举个例子，邮局大家都不陌生，我们可以到邮局寄信和收信。当我们寄信时，对于邮局而言我们就是信息发布者 Publish，当收信时我们就成了订阅者 Subscribe。同时，无论是寄信者还是收信者，我们都是客户。而邮局这个收寄信的平台就是所谓的服务器，为我们提供收信和寄信的服务。



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服务器。然后，服务器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，是指订阅者具体要接收的内容。

SIoT

简介：

SIoT 是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 也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。

五、实践过程

在本项目中，我们将分两步，自制一个基于 WiFi 的通信设备，实现一台计算机向物联网平台发送信息。

1、搭建 SIoT 物联网平台

2、向 SIoT 物联网平台发送消息

任务 1：平台搭建

1. 搭建步骤

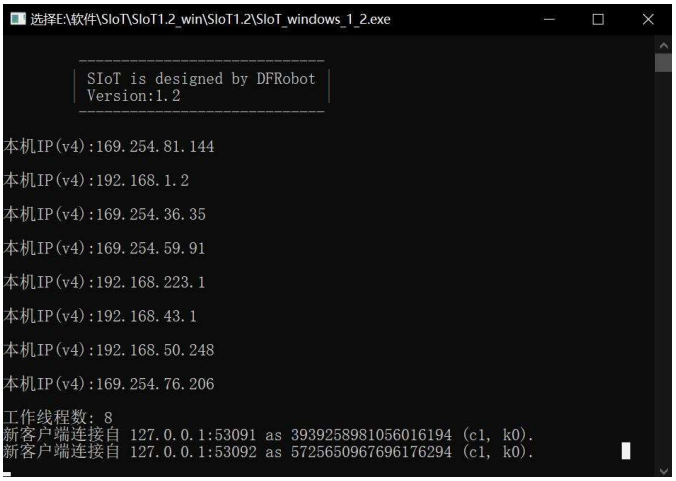
STEP1：电脑连接 WiFi

将电脑连接到 WiFi。

Tips：提供 WIFI 的路由器或手机热点可以不连接互联网，因为使用 SIoT 实现物联网应用时，只需要使用路由器或手机热点建立一个局域网即可。

STEP2：运行 SIoT 系统

双击运行 SIoT 应用程序，可以看到一个黑色的窗口。本课程以 win10 系统为例。



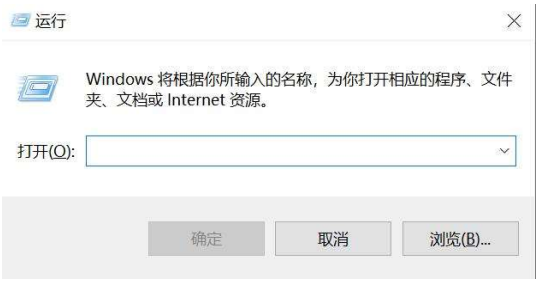
Tips：使用 SIoT 过程中一定不要关闭该窗口。

STEP3：电脑获取 IP 地址

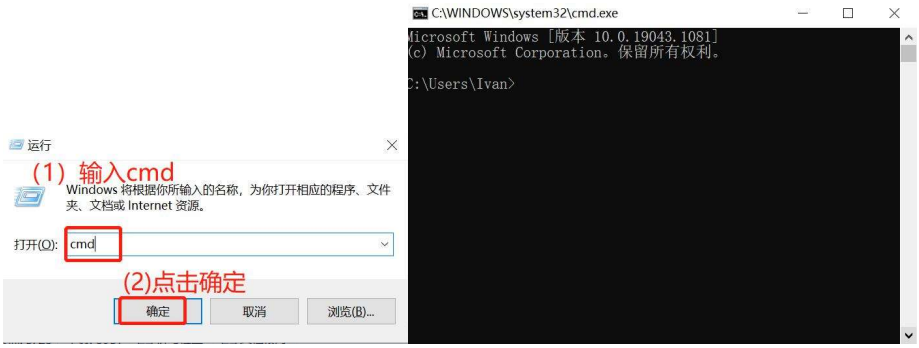
电脑每次连接 WIFI，都会生成一个 IP 地址，每个 IP 地址对应的电脑都是唯一的。运行 SIOT 程序后会在该电脑上建立一个服务器，其他设备要访问它，就需要知道该电脑的 IP 地址。

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。

（1）同时按下键盘上“WIN”+“R”，弹出如下运行窗口。

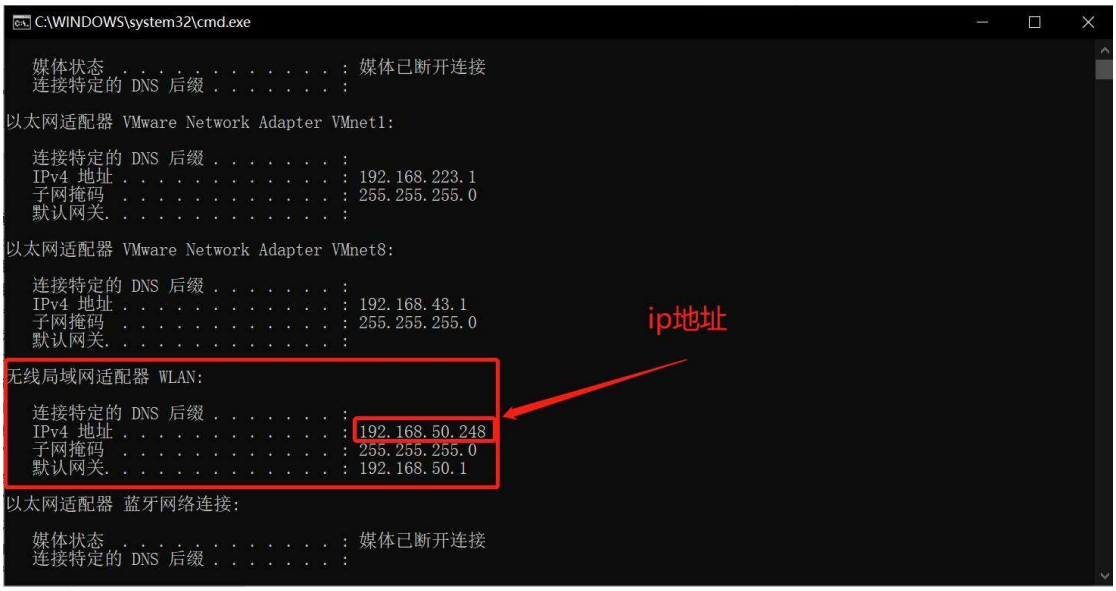


（2）输入“cmd”，点击确定，弹出小黑框。



（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 192.168.50.248。

Tips：连上不同的 WiFi，IP 地址是不一样的。



STEP4：打开服务器网页管理界面

（1）打开电脑浏览器，在网址栏输入 STEP3 中获得的 IP 地址，加上“:8080”，如，192.168.50.248:8080

Tips：“:”须在英文输入法下。

（2）点击键盘 enter 键，打开即为网页管理界面，如下图：

（3）打不开怎么办?

- 检查 SIoT 的小黑窗是否打开
- 检查 IP 地址是否错误，如果有多个 IP 地址就一个一个尝试
- 关闭网络防火墙

STEP5：登录账号

账号：siot，密码：dfrobot

输入账号、密码并点击登录，如下图，

Tips：网页端账号密码都是统一的

任务 2：向平台发送信息

1. 分析设计

在本任务中，我们可使计算机向搭建好的 SIoT 物联网平台上发送信息，并通过平台网页端查看相应接收到的信息。

2. 软件编写

在编写代码之前，我们依旧需要先对软件进行一些设置。

软件设置：

STEP1：创建与保存项目文件

启动 Mind+软件，选择“Python 模式”，另存项目并命名为“m 自制通信设备（上）”。

STEP2：创建与保存 Python 文件

创建一个 Python 程序文件“任务二.py”，双击打开。

STEP3：安装 SIoT 库

点击“库管理”，手动输入“pip install siot”进行安装。

Tips：Mind+软件版本在 1.7.1 以上无须手动安装 SIoT 库。

程序编写、运行及回顾：

STEP1：编写 Python 程序

```
import siot#导入 siot 库
import time#导入时间库

SERVER = "192.168.50.248"#MQTT 服务器 IP 地址，输入个人实际 Ip
CLIENT_ID = ""#在 SIoT 上，CLIENT_ID 可以留空
IOT_UserName ='siot'#用户名
IOT_PassWord ='dfrobot'#密码
IOT_pubTopic  = 'microbit/001'#“topic”为“项目名称/设备名称”

siot.init(CLIENT_ID, SERVER, user=IOT_UserName, password=IOT_PassWord)#初始化，确定输入的用户名和密码正确
siot.connect()#连接
siot.loop()#循环

while True:#永久循环
```

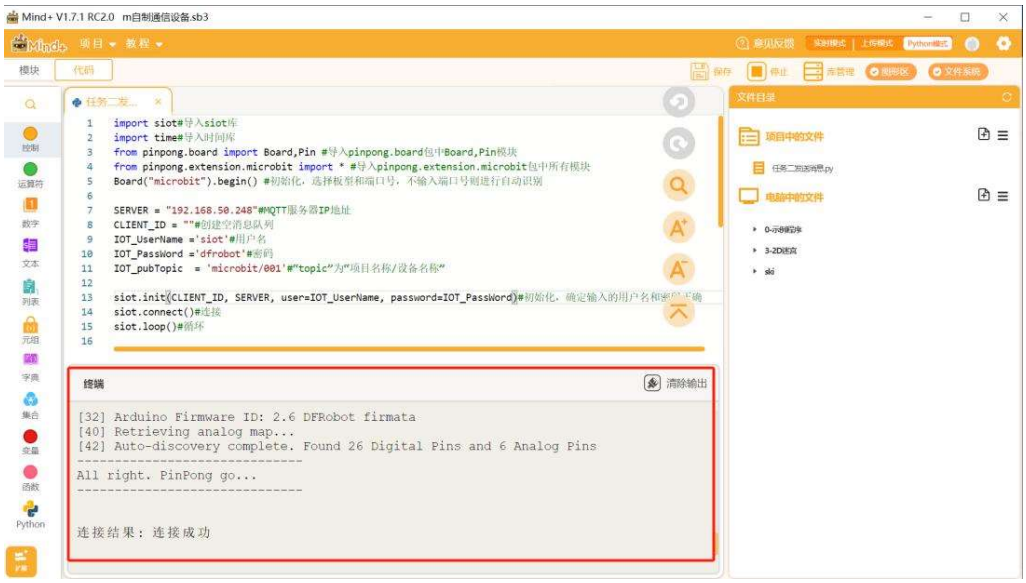
```
siot.publish(IOT_pubTopic, "Hello World") #发送消息
time.sleep(1)#延时 1 秒
```

Tips：上述“SERVER”中输入的是自己电脑的实际 IP 地址。

STEP2：运行程序并观察效果

（1）运行程序，观察软件终端

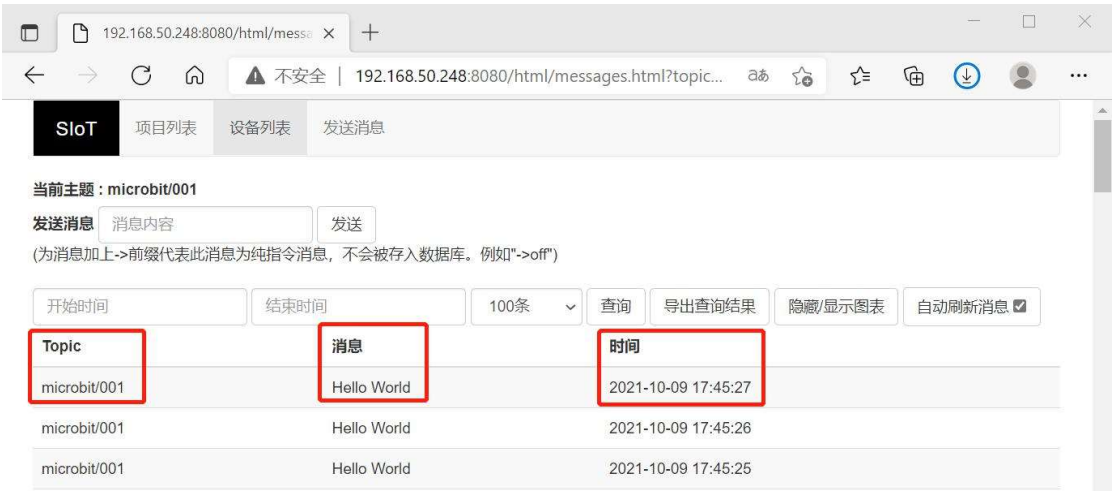
点击 Mind+软件右上方的运行按钮，我们可以看到在终端中显示“连接成功”的提示字样。



（2）平台查看消息

打开网页界面，先点击“设备列表”，可以看到“项目 ID”、“名称”分别对应为程序中“IOT_pubTopic”的信息；再点击“查看消息”，在弹出的页面中可以看到设备的 MQTT 消息记录；最后单击“自动刷新消息记录”，勾选后可以使消息实时自动刷新显示。

可看到程序中对 Topic:“microbit/001”的消息记录，伴有具体的消息内容及发送时间，如下图，



STEP3：回顾程序

1、导入库

```
import siot#导入 siot 库
```

为了能实现物联网相关的功能，我们需要先导入 siot 库，以便后续使用其中的函数功能。

2、设置 MQTT 相关参数

```
SERVER = "192.168.50.248"#MQTT 服务器 IP 地址
CLIENT_ID = ""#创建空消息队列
IOT_UserName ='siot'#用户名
IOT_PassWord ='dfrobot'#密码
IOT_pubTopic = 'microbit/001'#“topic”为“项目名称/设备名称”
```

我们需要在程序中设置好 MQTT 相关的参数以便连接上物联网平台。其中，“SERVER”指 IP 地址；“CLIENT_ID”指客户端标识符，它是客户端到服务器的唯一标识，这里我们可以不输入内容；“IOT_UserName”指物联网平台账户的用户名；“IOT_PassWord”指账号对应的密码；“IOT_pubTopic”指主题，包括项目 ID 和设备名称。这里的项目 ID 为“microbit”，设备名称为“001”，可自行更改。

Tips：IOT_pubTopic 的命名方式为“项目 ID/设备名”，且/必须在英文输入法下。

3、物联网平台相关操作

在连接物联网平台时，我们需要先初始化验证 IP 账号等信息是否正确，之后才能连接上 SIoT 平台并保持登录状态，接着可向 SIoT 平台发送消息进行通信。

（1）初始化，确定上述输入的 IP 地址、客户端标识符、账号用户名密码等信息是否正确：
siot.init(CLIENT_ID, SERVER, user=IOT_UserName, password=IOT_PassWord)

（2）连接 SIoT 物联网平台：siot.connect()

（3）保持物联网平台登录状态：siot.loop()

（4）向物联网平台的 Topic 发送消息
siot.publish(IOT_pubTopic, "Hello World")
这里的“IOT_pubTopic”指的是消息接收方，“Hello World”指的是具体要发送的内容，可以使用任何要发送的内容来作替代，如果内容存储在一个变量中则无需添加引号，这里我们可以发送 255 字符内的字符串，包括中文、英文等。

3. 调试修改

我们已经实现了每隔一秒自动发送消息至我们的 SIoT 物联网平台。但是，我们发现，每次发送的消息都是“Hello World”这一句固定不变，那如何才能随心所欲地自己决定要发送的消息内容呢?

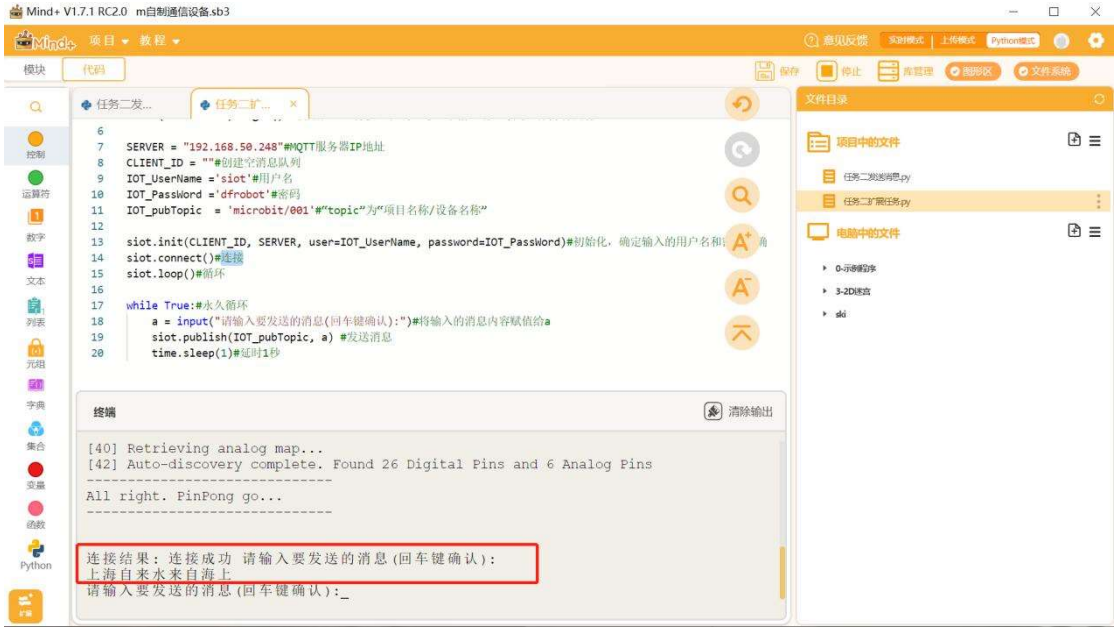
相信你一定早就想到了，没错，我们可以修改一下程序，借助“input()”输入函数来实现。

STEP1：修改 Python 程序

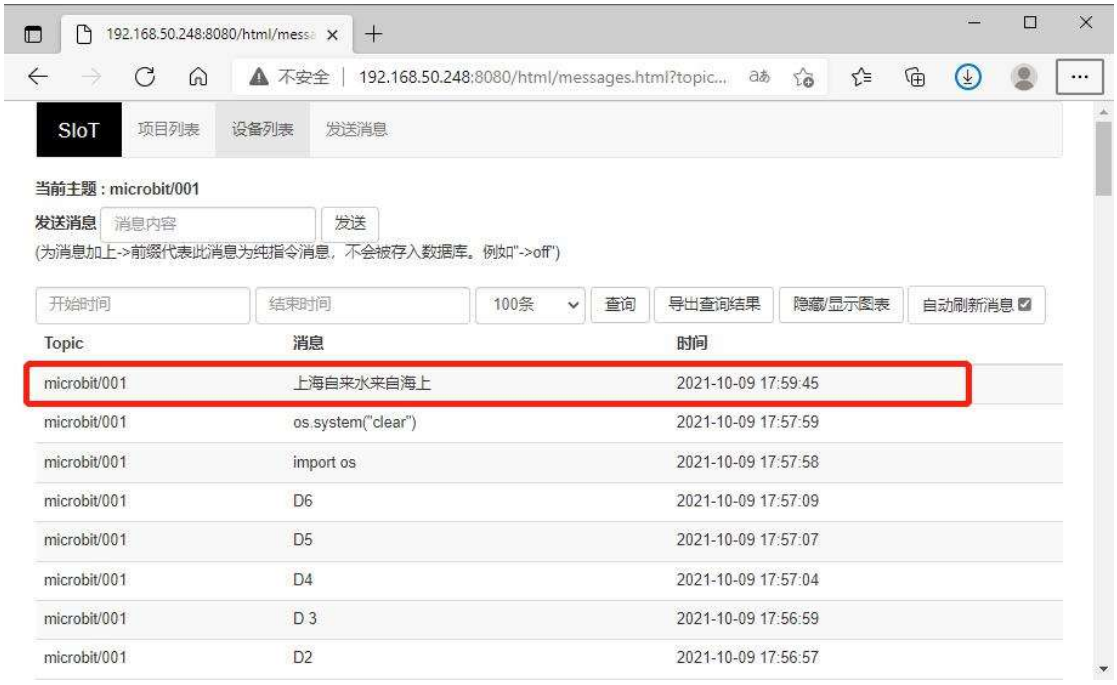
```
import siot#导入 siot 库
import time#导入时间库

SERVER = "192.168.50.248"#MQTT 服务器 IP 地址，输入个人实际 Ip
CLIENT_ID = ""#在 SIoT 上，CLIENT_ID 可以留空
IOT_UserName ='siot'#用户名
IOT_PassWord ='dfrobot'#密码
IOT_pubTopic  = 'microbit/001'#“topic”为“项目名称/设备名称”

siot.init(CLIENT_ID, SERVER, user=IOT_UserName, password=IOT_PassWord)#初始化，确定输入的用户名和密码正确
siot.connect()#连接
siot.loop()#循环

while True:#循环
    a = input("请输入要发送的消息(回车键确认):")#将输入的消息内容赋值给 a
    siot.publish(IOT_pubTopic, a) #发送消息
    time.sleep(1)#延时 1 秒
```

STEP2：运行程序

STEP3：在终端输入要发送的消息内容，并按下回车键确认



STEP4：查看消息记录



六、巩固提高

1.项目小结

通过实践及对内容的理解，我们知道了物联网是可实现万物互联的一种网络，MQTT 是物联网中一种用来传输消息的协议，而 SIoT 是以软件形式呈现的 MQTT 服务器程序。启动 SIoT 软件后，这台计算机也就成了一个标准的 MQTT 服务器，我们可以通过服务器的网页管理界面登录物联网平台的账号，在这里查看物联网平台上设备的所有消息记录。同样的，要想访问服务器，向物联网平台发送信息，使用任何一款 MQTT 客户端程序就可以了。这里，我们使用的是在 Mind+软件内编写的 Python 程序。

事实上，由于我们是在同一台计算机上一边运行 SIoT 系统，一边又编写 Python 程序访问服务器，所以这台计算机既是服务器，又是客户端。另外，物联网平台有很多种，这里我们利用 SIoT 系统生成的就称之为 SIoT 物联网平台。它是一种在局域网环境下即可建成的平台，而无须连入互联网。

因此，在实际生活中，只要有一台计算机开启 SIoT 服务，局域网内的所有计算机皆可编写程序与平台进行通信。

2.项目拓展

思考：

在生活中，无论是打电话还是发短信，通信往往都是双向的。在这节课上，我们实现了计算机向物联网平台发送消息，那我们是否也可以从物联网平台发送消息至计算机呢？