# 

基于物联网的项目

——花盆伴侣



# 目录

[认识物联网 2](#_Toc26776)

[物联网的由来 2](#_Toc12802)

[物联网的定义 2](#_Toc28628)

[物联网架构 3](#_Toc27035)

[基于物联网项目的特点 5](#_Toc7748)

[常见应用场景 6](#_Toc10045)

[1.3.2 物联网项目开发实例 9](#_Toc12453)

[【思考与练习】 9](#_Toc7142)

[1.功能特点 10](#_Toc9831)

[原理解释： 11](#_Toc14853)

[3.硬件组成 问题分析 11](#_Toc24059)

[4.软件设计 11](#_Toc11548)

\*点击页面左上角DF创客社区，有意外惊喜哦！

# 认识物联网

## 物联网的由来

“物联网”概念产生于20世纪90年代。1995年比尔·盖茨在出版的《未来之路》中提出“物—物互联”的设想，设想中提出“电子别针”的概念，将“电子别针”与家庭电子服务设施接通，并通过“电子别针”去感知来访者的位置，控制室内的照明和温度，控制电话和音响、电视等家电设备等。1998年，美国麻省理工学院Auto-ID 实验室项目提出了利用RFID、无线网络与互联网，构建物—物互联的物联网的概念与解决方案。而真正引起各国政府与产业界的重视是在2005年国际电信联盟（ITU）发布的互联网研究报告《物联网（Internet of Things，IOT）》之后。报告指出，无所不在的“物联网”通信时代即将来临，世界上所有的物体从轮胎到牙刷、从房屋到纸巾都可以通过因特网主动进行交换。射频识别技术（RFID）、传感器技术、纳米技术、智能嵌入技术将到更加广泛的应用。

2009年1月，IBM首席执行官彭明盛提出“智慧地球”构想，其中物联网为“智慧地球”不可或缺的一部分，而奥巴马在就职演讲后已对“智慧地球”构想提出积极回应，并提升到国家级发展战略。智慧地球将传感器嵌入和装备到电网、铁路、桥梁、隧道、公路、建筑、供水系统、大坝、油气管道等各种物体中，并通过超级计算机和云计算组成物联网，实现人与物的融合。通过在基础设施和制造业中大量嵌入传感器，捕捉运行过程中的各种信息，然后通过无线网络接入到互联网，通过计算机分析、处理和发出指令，反馈给控制器，远程执行指令。通过智慧地球技术的实施，人类可以以更加精细和动态的方式管理生产与生活，提高资源利用率和生产能力，改善人与自然的关系。

## 物联网的定义

维基百科描述：“英文Internet of Things（缩写IoT），是互联网、传统电信网等资讯承载体，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，以及搜寻位置、防止物品被盗等，类似自动化操控系统，同时透过收集这些小事的数据，最后可以聚集成大数据，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。”

物联网是在互联网、移动通信网等通信网络的基础上，针对不同应用领域的需求，利用具有感知、通信与计算能力的智能物体自动获取物理世界的各种信息，将所有能够独立寻址的物理对象互联起来，实现全面感知、可靠传输、智能处理，构建人与物、物与物互联的智能信息服务系统。

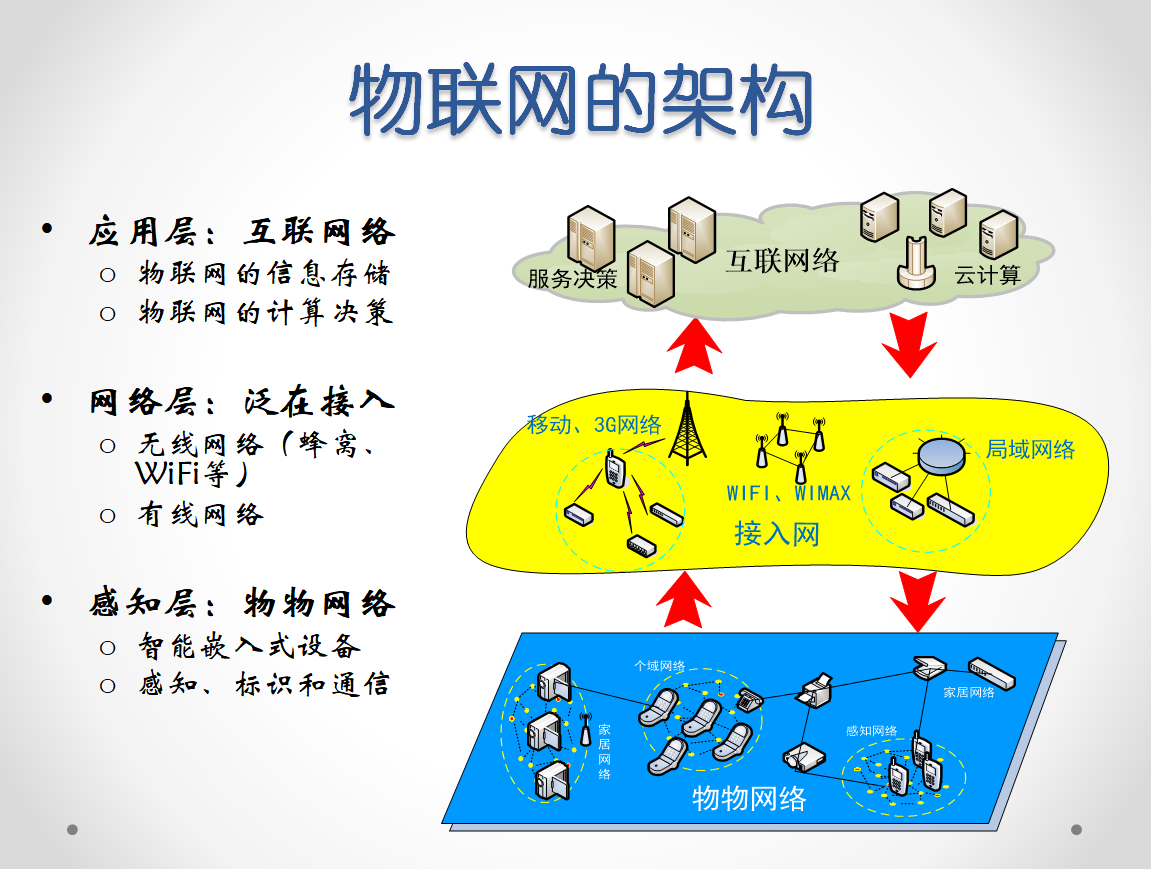


简单的来说，物联网就是一套物联物、人联物的智能系统，是将信号分散采集、集中处理、信息共享的一种理念装置。比如说，一个简单的IoT项目是物联网节点实时的采集CO2浓度，并定时的上传到云端，云端经过判断后，当浓度大于10%就发送邮件给你的邮箱，或者记录空气质量（作为环境监测的重要数据依据）。在这个例子中，CO2传感器需要和IoT主板连接，IoT主板需要通过WiFi连接到公网，实时的检测的CO2浓度数据并上传到云端服务器。服务器通过你早已设置好的逻辑进行计算，判断CO2浓度，将其结果通知到你的邮件，从而达到远程检测的目的。

## 物联网架构

物联网架构由三层组成：应用层、网络层、感知层

各层功能如图：





## 基于物联网项目的特点

物联网是在互联网基础上的延伸和扩展的网络，是物理世界与信息网络的无缝连接。在很多物联网项目中，除了包含人机互动、多机通信外，还有一个很重要的因素是互联网。正是因为有互联网的支持，才可以跨越空间的局限，实现任意距离的人与物、物与物的连接。在有些项目里，大多数物联网设备本身不具有直接访问Internet的能力，它们通过多机通信方式相互通信，并通过一个类似于网关设备访问Internet。而用户对设备发送的信息也是通过Internet传递给网关，并由网关转发给各个设备。这种方式比较适用于智能家居等场景中，系统拓扑结构如图1.3.2所示。

洗衣机

Internet

网关

电视机

空调

冰箱

用户

有些物联网应用因为要控制的设备较多，设备之间的相距较远，或者本身需要在移动中使用，所以设备本身需要具备访问Internet的能力，如共享单车上安装了一个智能锁，里面内置了移动网络通信模块，每把智能锁都可以通过GPRS、4G或NB-IOT等移动网络连接到Internet。当用户需要扫码骑车时，通过移动网络向共享单车运营商发起开锁请求，而共享单车运营商也通过移动网络对特定的智能锁设备发送开锁命令，系统拓扑结构如图1.3.3所示。

用户

Internet

智能锁

智能锁

智能锁

…………

物联网三大特点：

全面感知

“感知”是物联网的核心。物联网是具有全面感知能力的物品和人所组成的，为了使物品具有感知能力，需要在物品上安装不同类型的识别装置，例如：电子标签（Tag）、条形码与二维码等，或者通过传感器、红外感应器等感知其物理属性和个性化特征。利用这些装置或设备，可随时随地获取物品信息，实现全面感知。

可靠传递

数据传递的稳定性和可靠性是保证物-物相连的关键。为了实现物与物之间信息交互，就必须约定统一的通信协议。由于物联网是一个异构网络，不同的实体间协议规范可能存在差异，需要通过相应的软、硬件进行转换，保证物品之间信息的实时、准确传递。

智能处理

物联网的目的是实现对各种物品（包括人）进行智能化识别、定位、跟踪、监控和管理等功能。这就需要智能信息处理平台的支撑，通过云计算、人工智能等智能计算技术，对海量数据进行存储、分析和处理，针对不同的应用需求，对物品实施智能化的控制。

## 常见应用场景

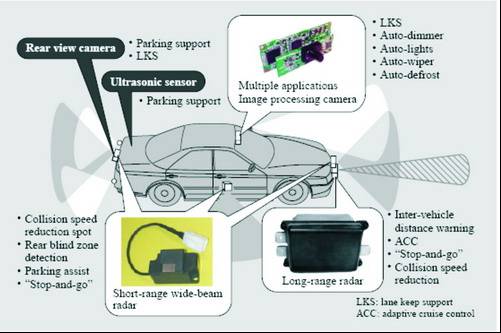
物联网生产线



物流自动化分拣系统



智能家居



智能交通“车联网”



路灯控制系统



自动售卖机

**1.3.2 物联网项目开发实例**

在市场上能够买到一些智能花盆或者智能盘栽方面的产品。这些产品大都可以通过网络获取花盆的环境信息，并对植物进行远程监控。

|  |  |
| --- | --- |
| IMG_256 | IMG_256 |

图4.3.4智能花盆

【思考与练习】

请找三款热销的智能花盆产品，了解具体参数并填写下面表格。

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 产品 | 智能浇灌 | 状态监测 | 通知提醒 | 支持手机管理 | 其他优势 |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

智能花盆产品自带的花盆一般都很小，不能种体积较大的植物。利用开源硬件，结合土壤湿度传感器和光照度传感器等传感器，对家庭常见盆栽进行改造，使之变得“智能”是挺有意思的一项工作。

**1.功能特点**

“盆栽伴侣”是一款能实时监测盆栽植物生长状态，并实现智能浇灌的系统。这个项目可以和任意一个花盆结合，让普通的花盆变身为智能花盆，用户还可以通过互联网实时了解花盆的状态并进行远程浇水控制。这一项目的功能需求分析如表4.3.1所示。

表4.3.1 “盆栽伴侣”的项目分析

|  |  |  |
| --- | --- | --- |
| 需求 | 描述 | 分析 |
| 实时监控 | 温度、光线等环境的监测，能够实时上传到Web服务器，通过Web平台能够实时查看植物生长状态。 | 采集这些数据，可以帮助用户科学地养护盆栽。 |
| 智能浇灌 | 能够自动根据土壤湿度进行智能浇灌，也通过Web平台可以远程给植物手动浇灌。 | 自动浇灌能够避免用户因为疏忽而“伤害”了盆栽。手动浇灌则促进用户和盆栽之间的互动。 |
| 感情互动 | 通过LED，以表情的形式显示植物的当前状态，当生长环境状态出现异常，能够实时报警。 | 养花种草本来就为了陶冶情操，和植物的互动不仅有趣，也有意义。 |
| 通用兼容 | 能够安装在常见的花盆中，操作方便。 | 通用性强是这一项目最重要的优势。 |

**2.项目规划**

“盆栽伴侣”主要由智能终端、Web服务器两大模块组成。智能终端实时采集环境光线、环境温度和土壤湿度等信息，通过网络上传到Web服务器。Web服务器将数据存入数据库，并通过网页实时显示出来，系统框架如图4.3.6所示。

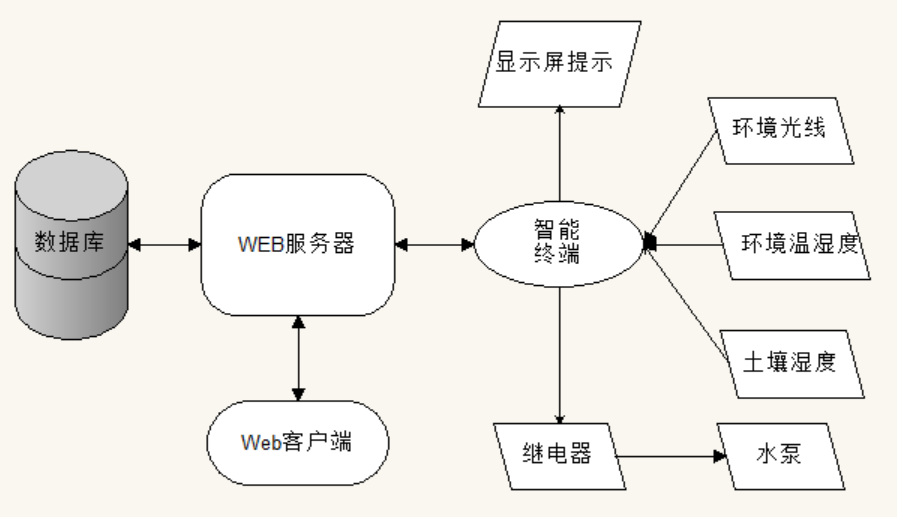


图4.3.6 盆栽伴侣整体框架图

**原理解释：**

**3.硬件组成 问题分析**

micro:bit自带了LED阵列，可以用来做花盆的信息提示器。为了获取土壤湿度，需要外接一个土壤湿度传感器。micro:bit虽然自带了温度传感器，但精度不高，所以需要使用第三方的温度传感器。为了实现访问互联网，还需要wifi模块OBLOQ。为了实现浇水，直流电水泵是必须的，但水泵需要较大的电压和电流驱动，因此还需要配备一个继电器模块，以实现用小电流来控制大电流等水泵。

表4.3.2中列出了该项目所需的器材。

表4.3.2 器材清单

|  |  |  |
| --- | --- | --- |
| **序号** | **器材名称** | **数量** |
| 1 | micro:bit | 1 |
| 2 | micro USB线 | 1 |
| 3 | 温度传感器 | 1 |
| 4 | 土壤湿度传感器 | 1 |
| 5 | 光线传感器 | 1 |
| 6 | micro:bit扩展板 | 1 |
| 7 | Wifi模块OBLOQ | 1 |
| 8 | 潜水泵 | 1 |
| 9 | 潜水泵电源连接线 | 1 |
| 10 | 继电器模块 | 1 |
| 11 | 橡胶水管 | 1 |

其中带WiFi模块的Microbit主控板是智能终端，各种传感器、继电器等设备连接到主控， 收集各自的数据信息。实物图如图4.3.7所示:

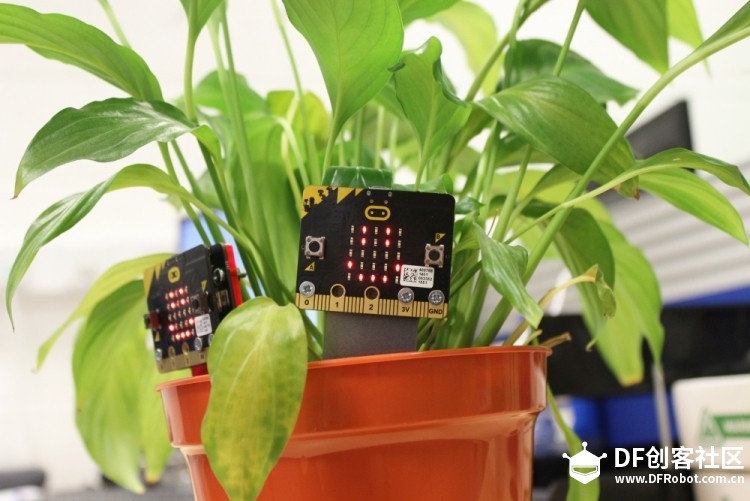


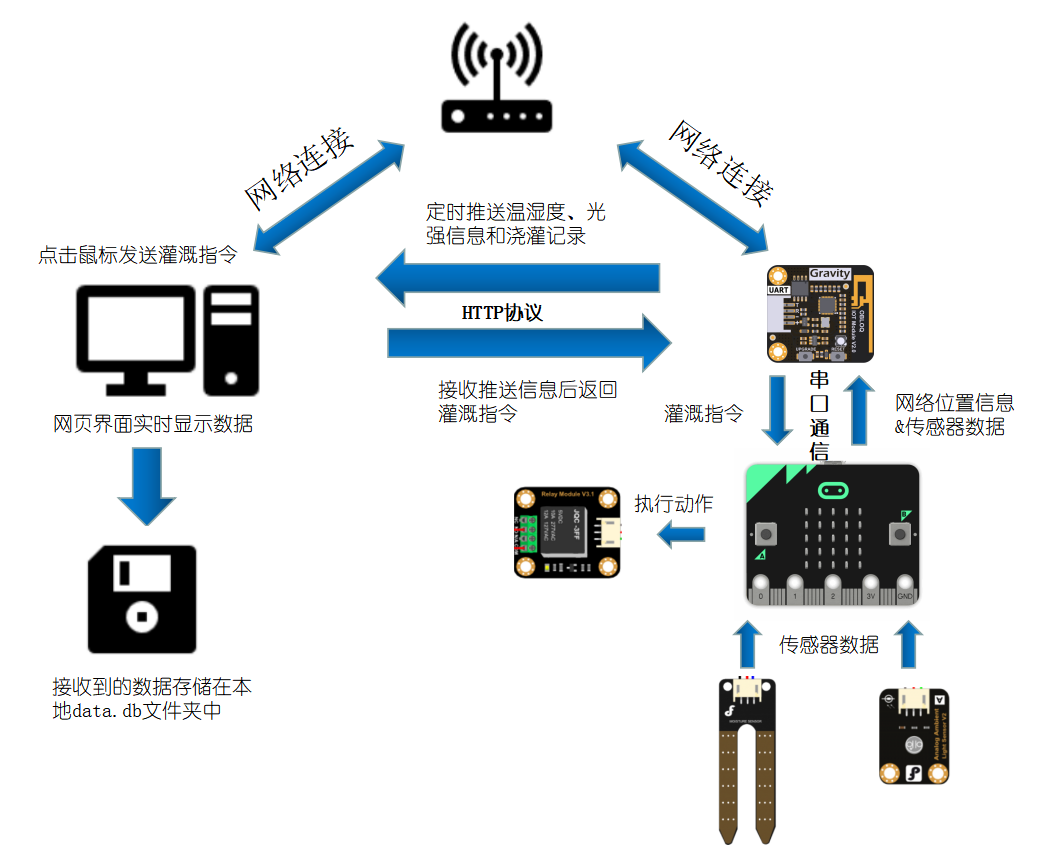
图4.3.7 硬件实物

**4.软件设计**

“盆栽伴侣”涉及到智能终端和Web服务器两大模块。二者之间的数据传输以HTTP协议通过GET或者POST方法传送数据给Web服务器。HTTP协议（HyperText Transfer Protocol，超文本传输协议）是用于从WWW服务器传输超文本到本地浏览器的传输协议，它的工作方式是客户机与服务器之间的请求-应答协议。在本例中就是通过Web服务器从网页接收来自智能终端的数据，也可以从网页发送数据至智能终端，实现两者信息交换。在客户机和服务器之间进行请求-响应时，两种最常被用到的方法是：GET 和 POST。

* GET - 从指定的资源请求数据。
* POST - 向指定的资源提交要被处理的数据

可以从以下的网络拓扑图看到



智能终端连接至互联网后传感器的数据能定时传送至Web服务器，Web服务器能把接收到的数据存储至数据库文件（如data.db文件），也可以通过Web客户端（如网页）显示出来。同样的，Web客户端可以发送指令至Web服务器，智能终端每一次发送数据后，Web服务器返回的信息中包含了是否浇灌的指令，即：返回2则浇灌，默认情况下是1。当智能终端获取了Web发送的浇灌指令后，启动水泵进行浇灌，持续1-2秒后停止。

数据交换遵循如下表所示的格式：

表4.3.3 数据交换描述

|  |  |  |
| --- | --- | --- |
| **数据交互动作** | **格式** | **返回值** |
| 智能终端发送数据及获取指令 | /input?val1=[环境温度]& val2=[环境光线]& val3=[花盆湿度] | 2，浇灌指令（服务器返回）  1，默认值  0，系统异常 |
| Web客户端发送浇灌指令 | /set?open=1 | 1，表示指令发送成功，其他信息表示系统异常（智能终端返回） |

因为数据量不大，可以采用Python自带的数据库Sqlite3。数据库中有一张sensorlog表，用于存储各种传感器数据。表中各字段设置如表4.3.4所示。

表4.3.4 数据库设置

|  |  |  |
| --- | --- | --- |
| **字段** | **数据类型** | **作用** |
| logid | INTEGER，自动编号 | 记录编号，主键 |
| sensor1 | float | 环境温度 |
| sensor2 | float | 环境光线 |
| sensor3 | float | 花盆湿度 |
| updatetime | time | 上传时间 |

Web服务器则采用Python的Flask（如何加载库）框架编写。首先为Web服务器编写一个主页模版index.html，并在模版中插入一个table标签。

在Flask框架中，通过render\_template绑定该模版,并把从数据库中查询到的环境数据显示到主页中。参考代码如下：

|  |
| --- |
| #在浏览器窗口中显示数据  @app.route("/")  def hello():  db = sqlite3.connect(DATABASE)  cur = db.cursor()  cur.execute("SELECT \* FROM sensorlog")  data = cur.fetchall()  cur.close()  db.close()  temp1 = data[len(data) - 1]  return render\_template('index.html',sensor1=temp1[1], sensor2=temp1[2], sensor3=temp1[3], sensor4=temp1[4]) |

Web服务器获取智能终端上传的环境数据，并且保存到数据库中。参考代码如下：

|  |
| --- |
| #用于接收Obloq发出的http数据，并储存至文件data.db中。同时根据浇水变量的值  #返回相应的浇灌指令。  @app.route("/input",methods=['POST','GET'])  def add\_data():  try:  global watering  if request.method == 'POST':  sensorvalue = request.form.get('val')  else:  sensorvalue = request.args.get('val')  ss=sensorvalue.split(',') #  if len(ss)==4: #  nowtime = datetime.datetime.now()  nowtime = nowtime.strftime('%Y-%m-%d %H:%M:%S')  db = sqlite3.connect(DATABASE)  cur = db.cursor()  cur.execute("INSERT INTO sensorlog(sensor1,sensor2,sensor3,sensor4,updatetime) VALUES(%f,%f,%f,%f,'%s')" %(float(ss[0]),float(ss[1]),float(ss[2]),float(ss[3]),nowtime))  db.commit()  db.close()  if watering == 1:  watering = 0  return '2' #当浇灌变量为1时，返回给Obloq字符串"2"，使其启动浇灌2秒钟，同时将浇灌变量重置为0。  else:  return '1' #当浇灌变量不为1时，返回给Obloq字符串"1"，使其保持停止浇灌状态。    else:  return '0' #指令无法被识别或通信出错时，返回给Obloq字符串"0"  except:  return '0' #指令无法被识别或通信出错时，返回给Obloq字符串"0" |

作为智能终端的micro:bit通过串口与Wifi模块通信，定时向Web端传递温度、湿度等环境数据，同时根据Web端的返回值来确定是否浇水。当Wifi模块配置好网络参数后，以下代码就可以通过GET方式向Web端上传一组环境数据。

|  |
| --- |
| errno,resp = Obloq.get("input?id=1&val="+str(temp)+','+str(light)+','+str(moisture)+','+str(watering),500) |

**操作过程:**

1. 对于web服务器的编写，我们需要用到Python，对于Windows系统需先安装最新Python，并搭建Python的开发环境

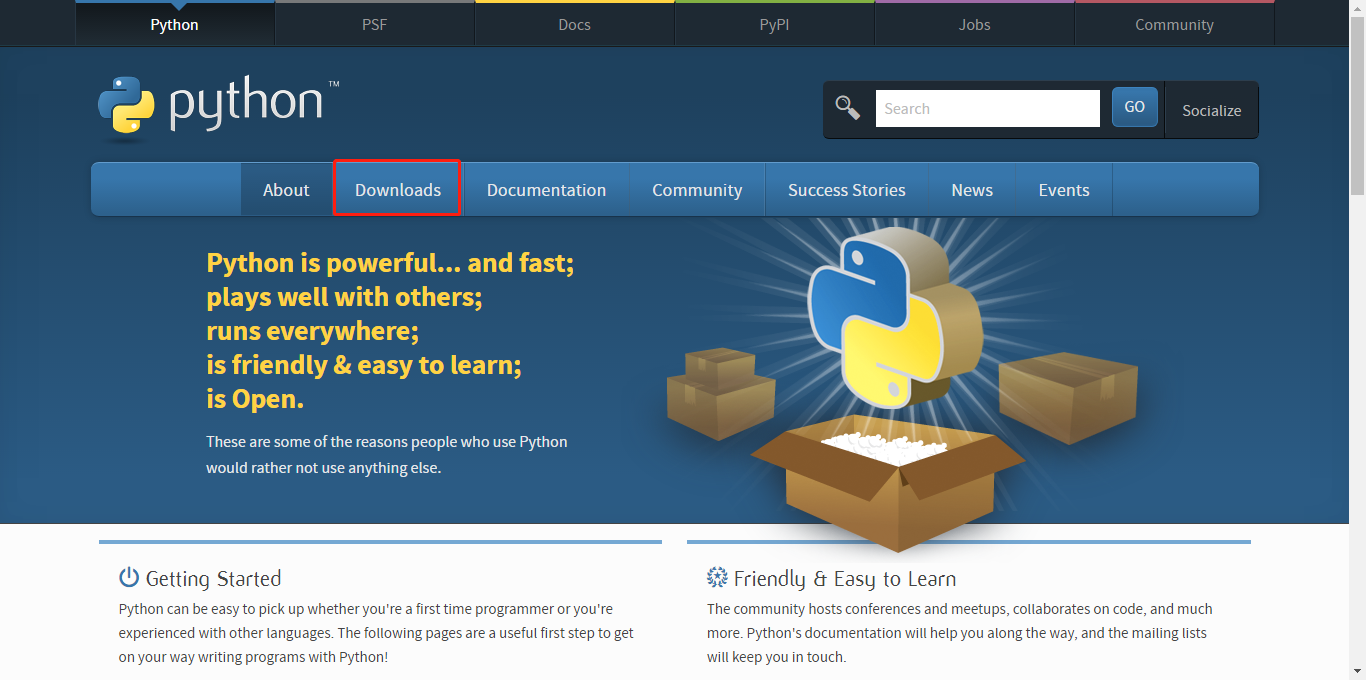
**本地搭建Python开发环境**

Python可应用于多平台包括windows、 Linux 和 Mac OS 等，以下介绍在windows和 Mac OS系统搭建Python开发环境。

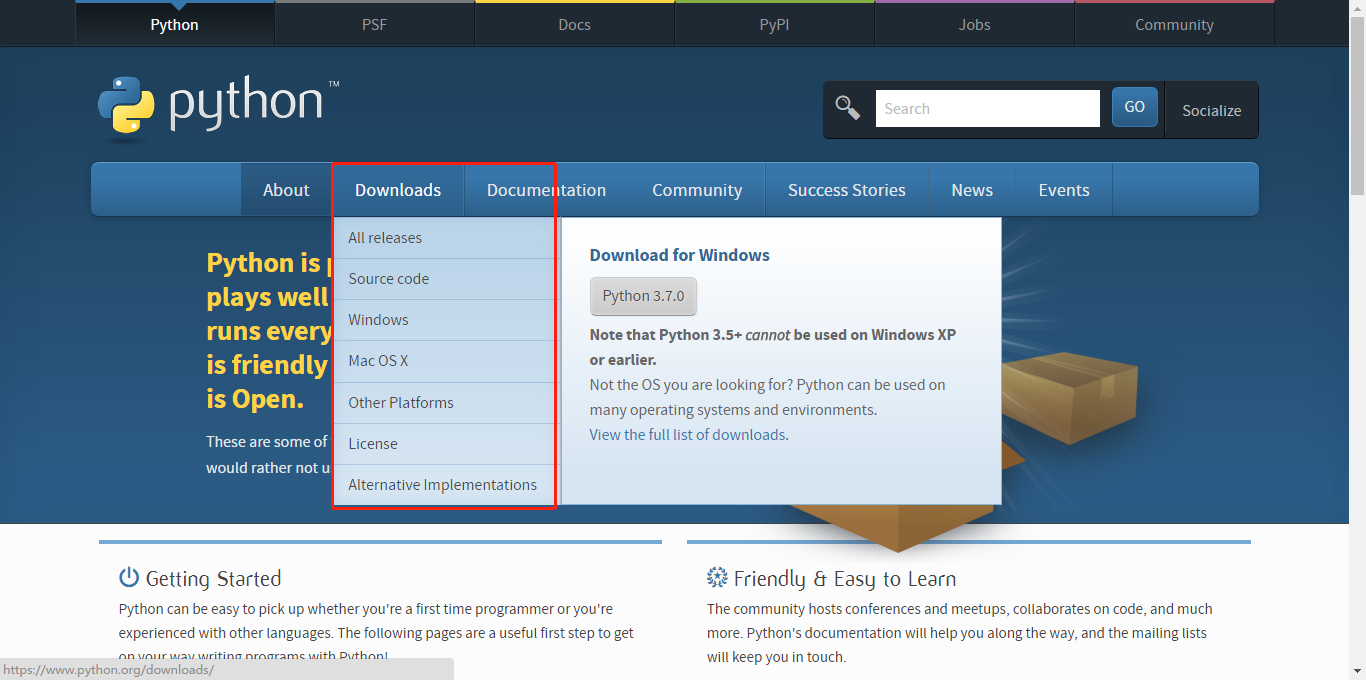
**Python下载**

关于Python最新源码、二进制文档、社区、新闻资讯等可以在Python的官网查看到：

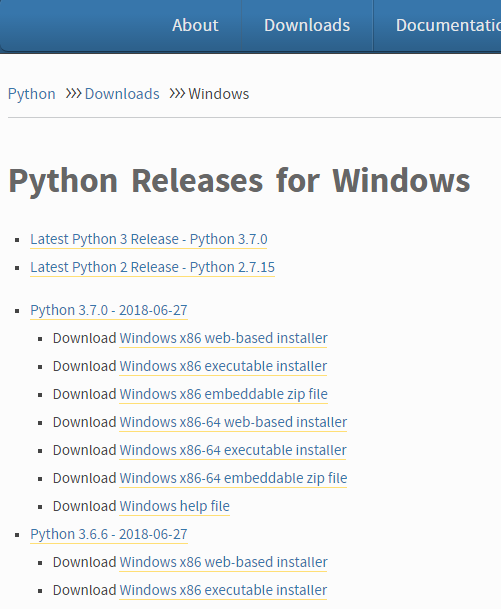
Python官网：[https://www.python.org/](https://www.python.org/" \t "https://www.runoob.com/python/_blank)



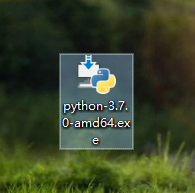
光标置于上图红色区域Download可以选择下载 Python 不同平台的安装包：



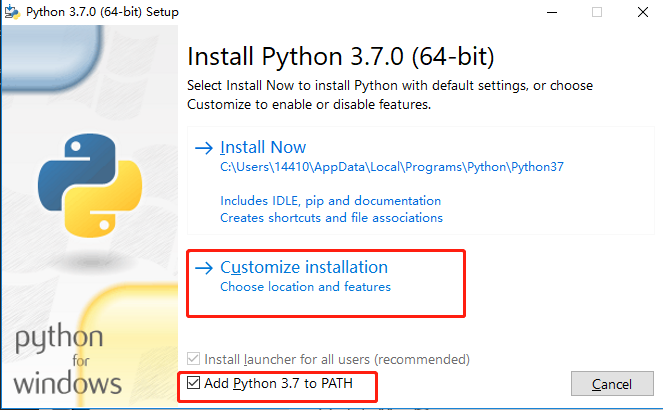
点击选择Windows,在弹出的界面，建议选择下图红色的稳定版安装包，进行下载。



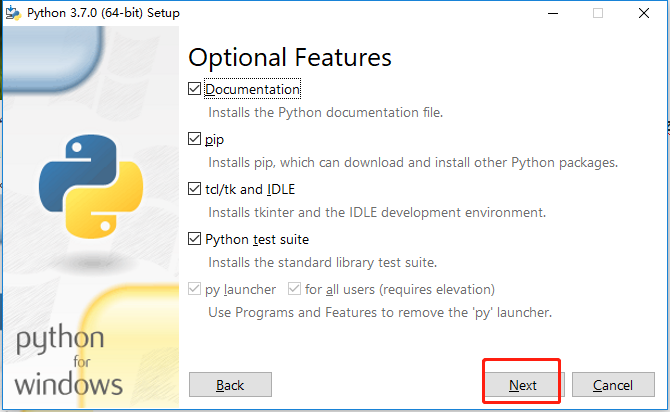
下载完成后安装包如图：



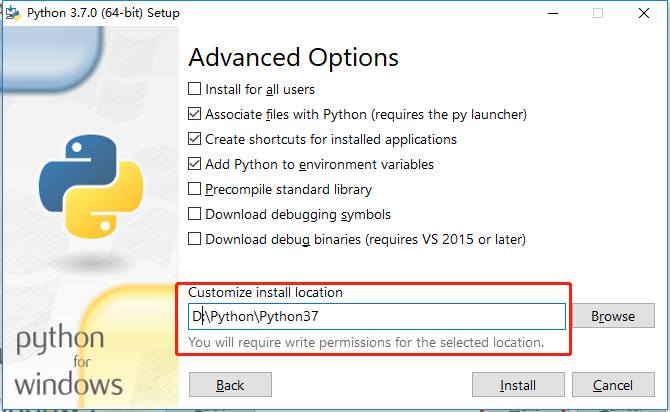
右键“以管理员身份运行”进行安装，勾选“Add Python 3.7 to PATH”,选择Custoomize installation,



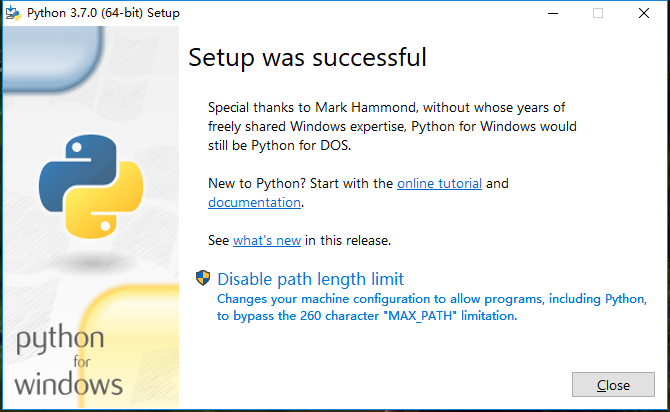
点击Next，



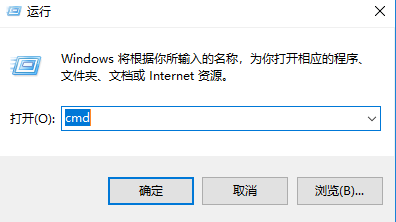
更改本地的安装路径，直接点击Install,进行安装。

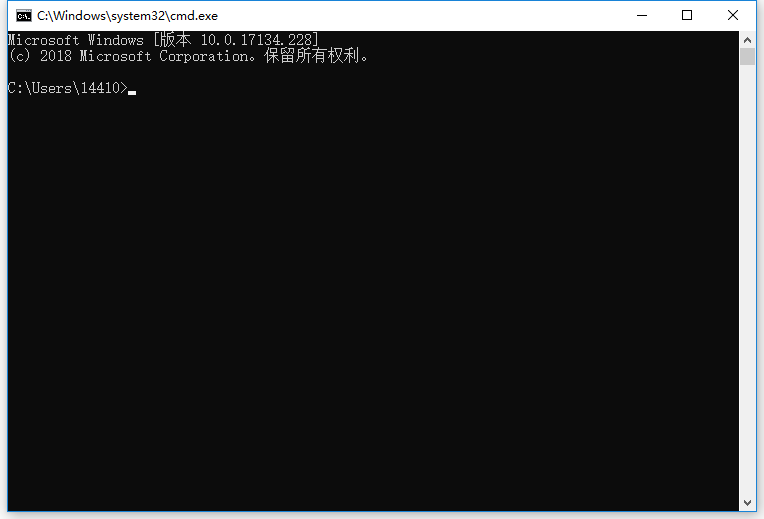


安装过程持续约1分钟，安装时间和电脑配置相关，安装完成后的界面如下图。

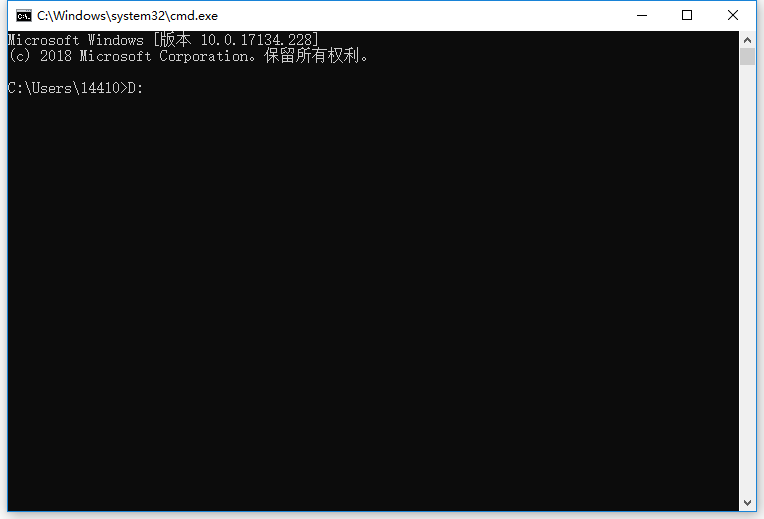


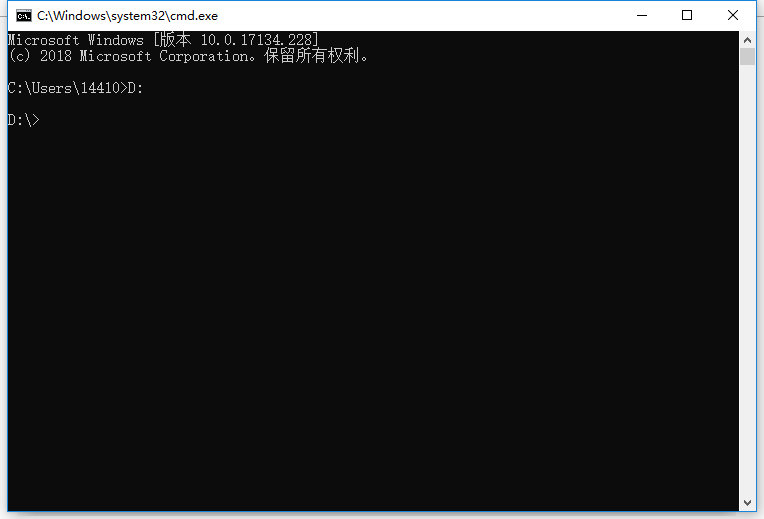
由于代码中用到Flask框架，在安装好Python后安装Flask框架，安装Flask前先安装虚拟环境，win10环境下，运行win+R快捷键打开运，输入cmd回车，

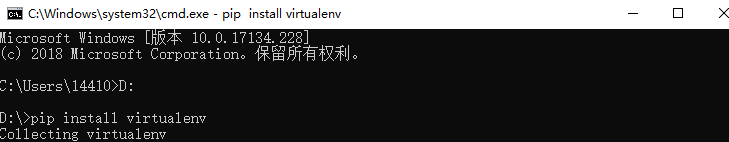


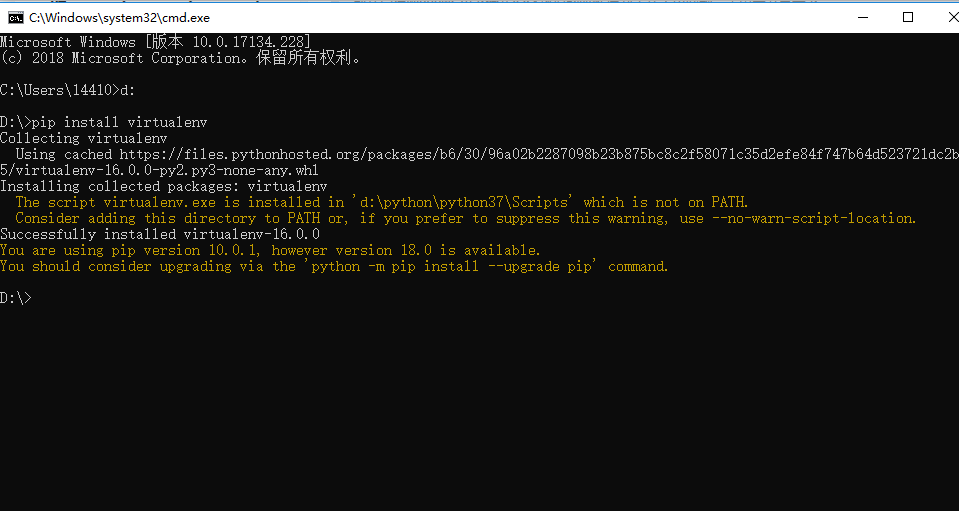


前面python软件安装在D盘，切换到D盘下安装虚拟环境，直接输入”D:”回车后进入D盘，

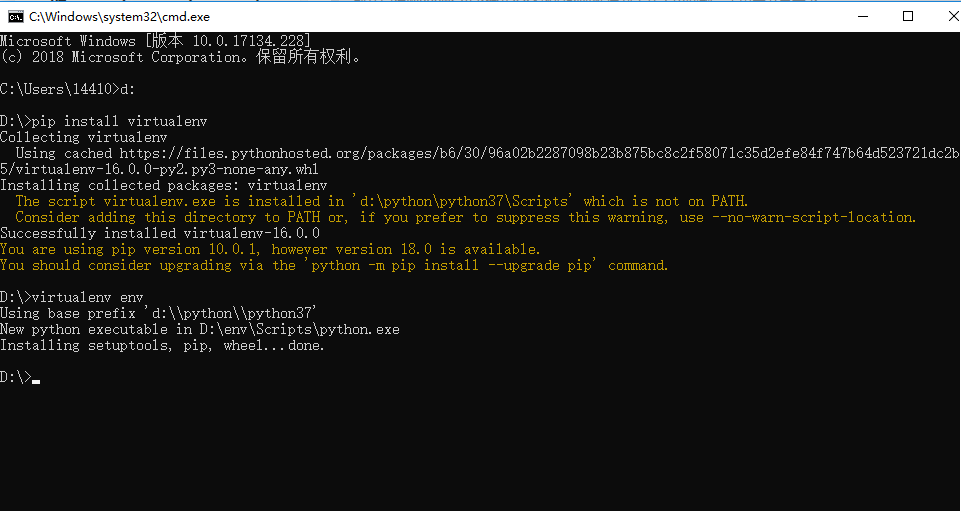


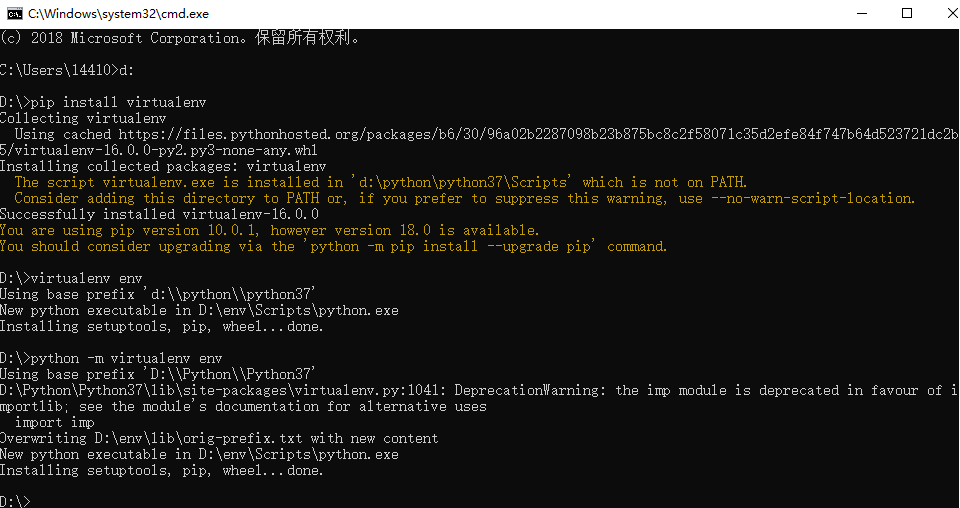
输入 pip install virtualenv回车，安装虚拟环境，





安装完成后，输入 virtualenv env 开启虚拟环境





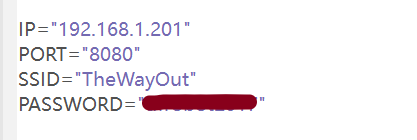


至此，整个pyrhon开发环境配置完毕。

Microbit支持Micropython编写代码，可以使用Microbit的编辑器BXY Python Editor（编辑器下载地址： <http://docs.dfrobot.com.cn/bxy/>）进行代码的编辑，



终端Microbit主控板的参考代码在文件中，打开BXY软件后打开microbit.py文件，这个代码是配套不同网络的代码，如果更换了网络环境，需要对代码左简单修改。主要是对IP和Wifi的配置：

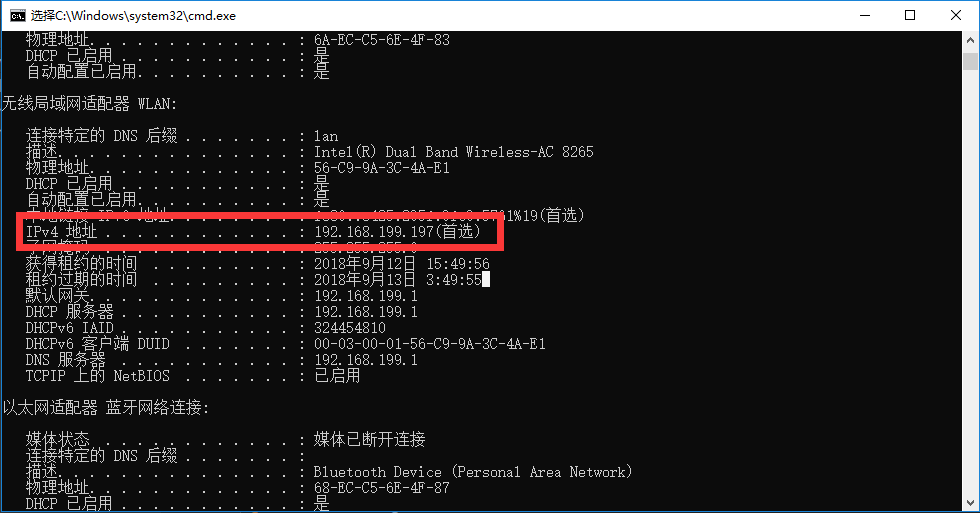


上图代码IP是个人电脑的IP，在程序下载前，需要查看本地IP并修改。修改方法如下：

Win+R,输入cmd回车，



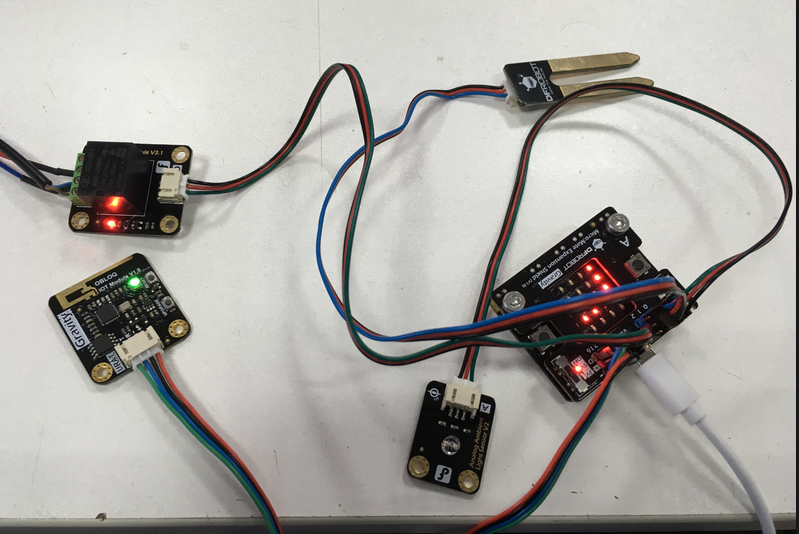
接着输入 ipconfig/all回车，



选取IPv4地址，填入Microbit代码的IP并进行文件保存。

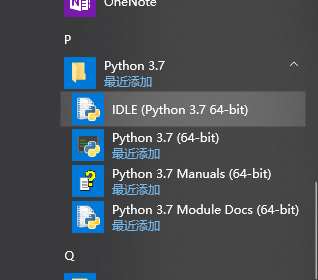
SSID 和PASSWORD是WIFI的账号和密码，选取我们使用的WIFI直接填入，保存文件。至此，终端的程序已经完成。USB线连接Microbit主板直接点击烧录进行程序的下载，下载完毕后，弹出烧录完毕的提示窗，





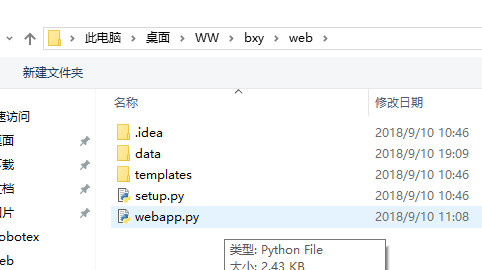
硬件连接好以后，可以看到WIFI模块的指示灯由红色——蓝色——绿色渐变。绿色代表已经连上WIFI。

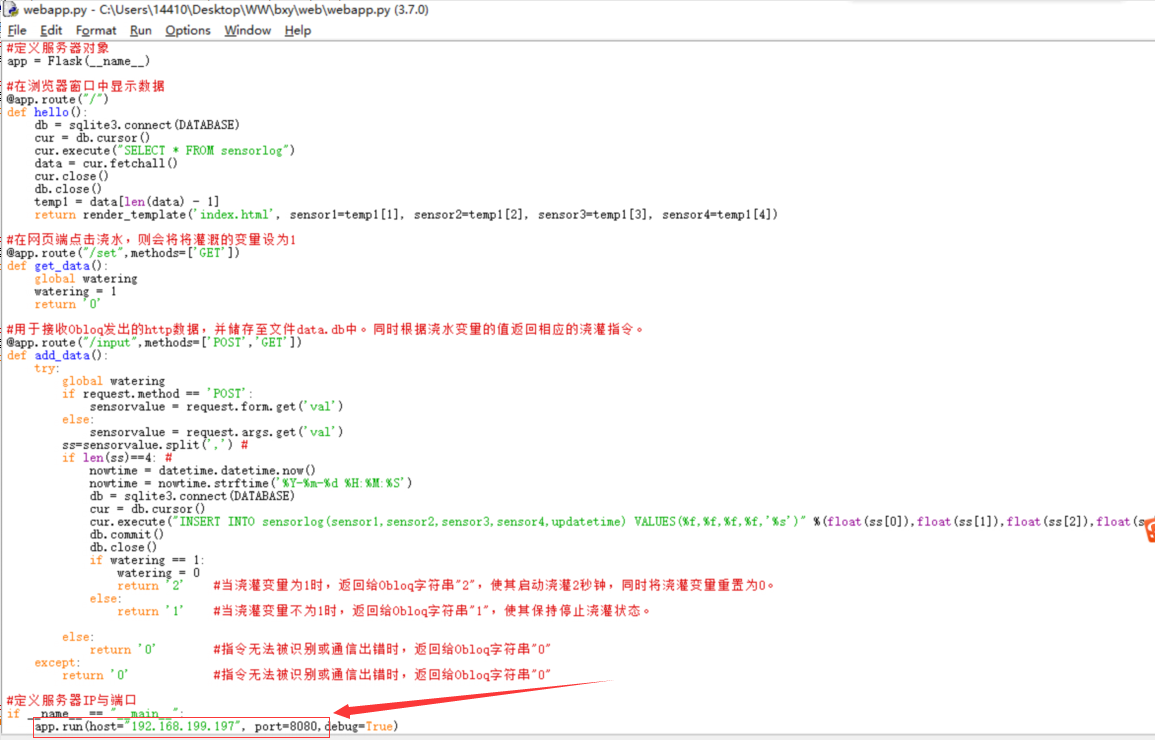
配置好终端后，需要对服务器端代码进行修改。在“开始”中打开python的代码编辑器IDLE，





打开服务器端代码：file->open ,选择webapp.py文件

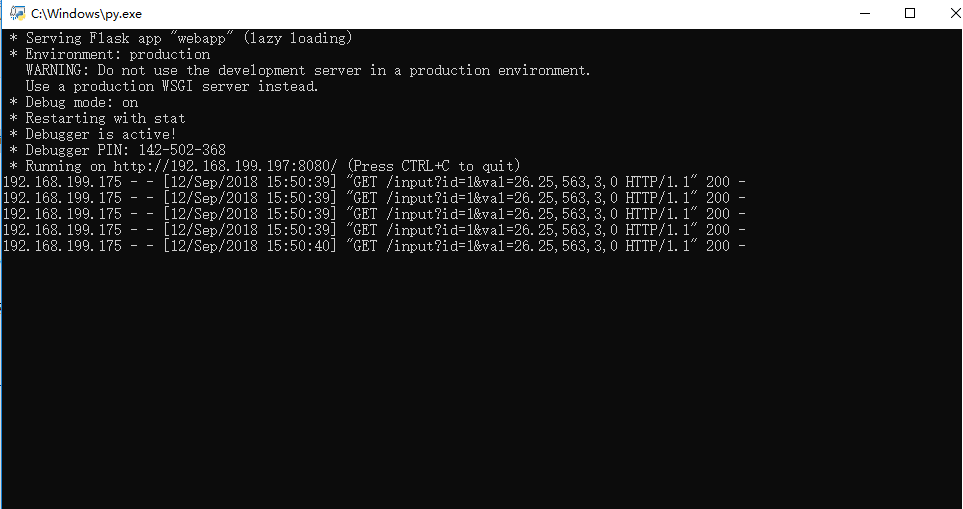


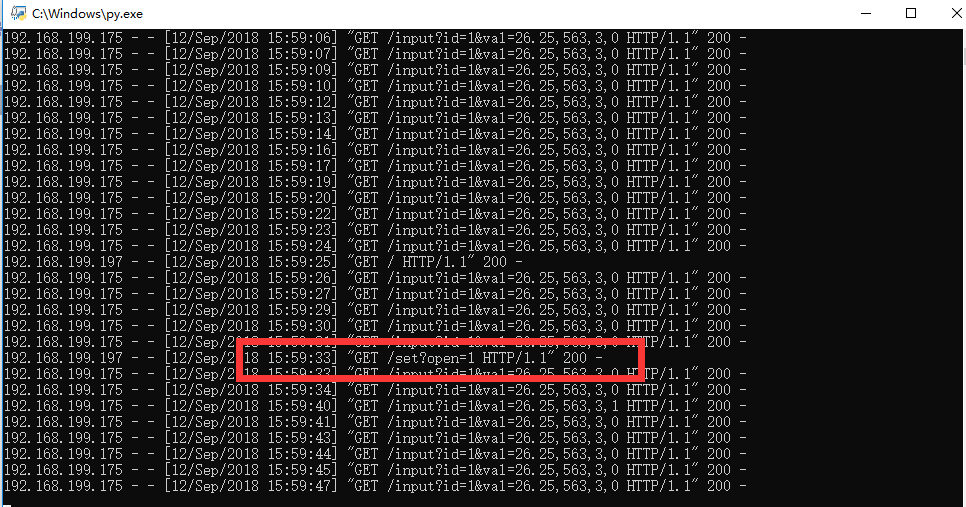


和修改终端IP的方法一样，修改代码最下面的IP地址并保存。至此所有代码全部修改完毕。

直接双击运行setup.py（操作步骤），用于首次使用本系统，运行一次就行。

双击运行webapp.py，及web客户端（浏览器），可以在对应IP网址中查看数据，结合例程的IP为http://192.168.199.197:8080/ ，可以在cmd中看到客户端通过GET协议获取id=1的各个数据，Micro:bit每隔500毫秒会向服务器发送数据（即图中来自192.168.199.197的信息）。





红圈1：当点击网页中的“浇水”按钮时，服务器会接收到“/set?open=1”指令（即图中来自192.168.199.197的信息），会返回给Obloq浇灌指令。

红圈2：Obloq完成浇灌后，发送的表示浇灌历史的变量由0变为1

1. 打开http://192.168.199.197:8080/ （IP地址同上一次的） ，每1秒更新一次数值，（为什么数据没有实时更新）点击“浇水”，水泵开启浇水持续2秒
2. 点击“浇水”即可打开继电器开启浇灌。

**【实践与体验】**

**盆栽伴侣的编程实现**

“盆栽伴侣”是一个比较综合的项目，不仅能够实时获取花盆的环境信息，还能够远程给植物浇水。该项目最大的难度在于对各个模块的编程实现。通过编写代码，可以更好的理解这一项目的各个模块是如何协同工作的。

实践内容：

1.编写Web服务器代码，实现上传数据的存储与显示。

2.实现智能终端通过Wifi模块上传环境数据并获取Web端的返回值。

3.实现读取温湿度传感器、光线传感器、土壤湿度传感器的值，并实现对继电器模块的控制。

4.调试项目。

实践步骤：

1.在Python环境下，利用Flask框架编写Web服务器代码，使得服务器端可以正确接收GET或POST方式的传值。在编写代码过程中，可以先通过Chrome浏览器的Postman插件模拟给Web服务器发起GET请求。



图4.3.8 Web页面效果

2.连接micro:bit和Wifi模块，编写Web服务器通信代码，实现将测试数据发送给Web服务器，并从Web服务器中获取返回值。

3.将温度传感器、光线传感器、土壤湿度传感器与micro:bit连接，编写代码获取这些传感器的值；通过micro:bit的数字口控制继电器模块。

4.项目整体调试。

这个项目的调试难点在于micro:bit通过Wifi模块和Web服务器的连接。因为micro:bit和Wifi模块的通信是使用UART功能，如果发生错误，难以判断问题在Web服务器还是micro:bit端，建议参考下面两种方法来调试。

方法1：利用面包板，将连接Wifi的引脚并联出来，连接到一个串口模块上，同步将信息发送到电脑串口。

方法2：利用无线功能，将uart的所有信息同步发送到其他的micro:bit板子上。而作为接收信息的micro:bit板子再用print的方式发送到电脑来查看。当然，在教室里多人同时使用会造成信息干扰，要加上地址码进行区分。

欢迎登陆DFRobot创客社区!

DFRobot 创客社区: www.dfrobot.com.cn