# 物联网毕业设计

## 绪论

### 选题背景

随着科技的发展，计算机电子技术迅猛发展，已经成为生活中不可缺少的部分。 目前人们绝大多数都是采用PC进行网络数据传送，但由于成本高，限制了应用的 范围。而嵌入式系统却越来越受到人们的青睐。它采用嵌入式的微处理器，支持 TCP/IP协议，它已成为网络发展新阶段的标志。

物联网是新一代信息技术的重要组成部分。其英文名称是The Internet of things。顾名思义，物联网的意思就是物物相连的互联网。这有两层意思：第 一，物联网是建立在互联网之上的，是互联网的拓展和延伸；第二，其用户端扩 展和延伸到了物品与物品之间，进行信息通信和交换。物联网有如下特征：

首先，广泛应用了各种感知技术。在物联网中部署了大量的多种传感器，每个传 感器都能从外界采集信息，不同类的传感器捕获的信息不同。而且获得的数据具 有实时性，按照一定的规律来采集数据，不断更新数据。

其次，它是建立在互联网上的网络。物联网技术的核心和基础仍是互联网，通过 各种无线和有线网络与互联网结合起来，将物体的信息准确实时地传递出去，数 据传输过程中必须适应各种网络协议。

还有，物联网本身也具有一种智能处理的能力，能够智能控制物体。物联网从传 感器中获得数据，然后进行分析，处理处有意义的数据，来适应不同用户的需求。

### 设计内容

设计主要是关于基于RESTful服务的网络服务构建，可采用有线网络、无线网络、 手机GSM网络等与Internet相关，通过手机、电脑 、移动设备等登录到网页可实 现控制家电的上的，并可实时查看诸如温度等一些信息的基本内容。

硬件设计时，采用Arduino单片机系统，作为一个基于Atmega328芯片的最小系统， Arduino可以系统代码。Arduino主要用于展示LED灯的控制，通过与Raspberry PI开发板相连来获取实时状态。Raspberry PI作为一个ARM开发板，由于其运行 的是Linux系统，在软件方面有着相对于其他开发板较好的支持，在这里是作为 数据传输设备以用来进行模块分离。

软件设计时，由于一个物联网系统其核心是以网络为基础的，需要优先考虑网络 方面的优化，学需要考虑数据库等的问题。

用户界面设计时，随着近来来平板、手机等移动设备的流行，在设计时不能再以 桌面程序为核心，需要考虑不用设备之间的兼容性等问题，这里便以网页为核心 作为显示。而，随着云计算技术的流行，未来的物联网系统必然也会基于云计算 技术构建。作为一个可视化的网页来说，实时的状态显示等是较为重要，同时我 们需要考虑的是用户体验。

### 设计的目的及其意义

设计以简化物联网系统为主，简化一个可扩展的最小的物联网系统，以简化系统 的逻辑为起点，为广大的用户提供一个良好的了解物联网系统方面知识的框架。

### 国内外发展现状和趋势

物联网是建立在互联网技术之上的。目前，我国物联网发展与全球同处于起步阶 段，初步具备了一定的技术、产业和应用基础，呈现出良好的发展态势。把单片 机应用系统和Internet连接也已经是一种趋势。

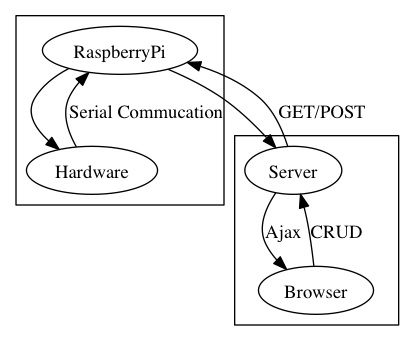
目前无线通信网络已经覆盖各地，是实现“物联网”必不可少的设施，可以将安置 在每个物品上的电子介质产生的数字信号通过无线网络传输出去。“云计算”技术 的运用，使数以亿计的各类物品的实时动态管理变得可能。

物联网技术的推广已经取得一定的成效。在多方面已经开始应用，如远程抄表， 电力行业，视频监控等等。以及在物流领域和医疗领域也都日趋成熟，如物品存 储及运输监测，远程医疗，个人的健康监护等。除此之外在环境监控，楼宇节能， 食品等方面也开展了广泛应用。

尽管在这些领域已经取得一些进展，但应认识到，物联网发展技术还存在一系列 制约和瓶颈。有几个方面可以表现出来：核心技术与国外差距较大，集成服务能 力不够，缺乏骨干企业，应用水平不高，信息安全存在隐患。我们国家在PC架构 领域还没有主动权，软件产品很少。目前，计算环境正在向以网络为中心发展， 有很多产品不必也windows兼容，因此，研究单片机系统接入网络，前途宽广。

# 系统总体设计方案

Arduino+Raspberry Pi+Laravel+JSON+RESTful+Ajax+Python+HighCharts



系统框架图

Arduino与Raspberry Pi通过串口通信的方式实现通信，相互传输所需要的数据， Raspberry Pi将资源传于互联网上对应的接口，接口可以在互联网上被访问。 Laravel框架构架于服务器之上，将Raspbery Pi获取过来的数据存储于MySQL数 据，再以REST服务的方式共享数据，互联网上的其他设备便可以通过网络来访问 这些设备。Ajax用于将后台的数据以不需要刷新的方式传递到网站前台，通过 HighCharts框架显示给终端用户。

## 网络服务设计

物联网的核心也就是网络服务，而网络服务在某种意义上来说，就是需要打造一 个多平台的通信协议，在使机器、家电、设备等连上计算机网络。基本的物联 网系统，不仅能控制设备，还可以在远程查看状态。而复杂的物联网系统可以让 互联网上的设备之间实现互联与通信，也就是物联网的最终目标所在——使物体与 物体之间的交互成为可能，不需要人为去干预。

设备在现实世界就是一种资源，在互联网上也应该是一种资源，互联网上的网页 就相当于是一种资源。

## 硬件方案选择

### 单片机选择

#### Arduino

Arduino，是一个开放源代码的单芯片微电脑，它使用了Atmel AVR单片机，采用 了基于开放源代码的软硬件平台，构建于开放源代码 simple I/O 接口板，并且 具有使用类似Java，C 语言的Processing/Wiring开发环境。

Arduino开发板封装了常用的库到开发环境中，可以让用户在开发产品时，将主 要注意力放置于所需要实现的功能上，而不是开发的过程中。在为Arduino写串 口程序时，我们只需要用Serial.begin(9600)以9600的速率初始化串口，而在往 串口发送数据时，可以用Serial.write(‘1’)的方式向串口发送字串’1’。

#### 51

单片机[[1]](#footnote-34)，又称微控制器，是把中央处理器、存储器、定时/计数器 （Timer/Counter）、各种输入输出接口等都集成在一块集成电路芯片上的微型 计算机。与应用在个人计算机中的通用型微处理器相比，它更强调自供应（不用 外接硬件）和节约成本。它的最大优点是体积小，可放在仪表内部，但存储量小， 输入输出接口简单，功能较低。

51单片机相较于Arduino开发板，不仅代码复杂，由于系统比较古老而不方便于 快速开发。

## 软件方案选择

### 数据通讯方式选择

#### REST

REST[[2]](#footnote-38) 从资源的角度来观察整个网络，分布在各处的资源由URI确定，而客户端的 应用通过URI来获取资源的表征。获得这些表征致使这些应用程序转变了其状态。 随着不断获取资源的表征，客户端应用不断地在转变着其状态，所谓表征状态转 移。

#### SOAP

简单对象访问协议是交换数据的一种协议规范，使用在计算机网络Web服务中， 交换带结构信息。SOAP为了简化网页服务器从XML数据库中提取数据时，节省去 格式化页面时间，以及不同应用程序之间按照HTTP通信协议，遵从XML格式执行 资料互换，使其抽象于语言实现、平台和硬件。

### 数据通信格式选择

#### JSON

JSON[[3]](#footnote-42)是一种轻量级的数据交换格式。 易于人阅读和编写。同时也易于机 器解析和生成。 它基于JavaScript Programming Language, Standard ECMA-262 3rd Edition - December 1999的一个子集。 JSON采用完全独立于语 言的文本格式，但是也使用了类似于C语言家族的习惯[[4]](#footnote-43)。 这些特性使 JSON成为理想的数据交换语言。

JSON相对于XML来说可以减少文件的大小，同时我们可以用于网站前端的数据通讯。

对于基于浏览器的客户端使用的web服务更倾向于使用JSON作为表述格式。

#### XML

可扩展标记语言[[5]](#footnote-45)，是一种标记语言。标记指计算机所能理解的信息符号， 通过此种标记，计算机之间可以处理包含各种信息的文章等。如何定义这些标记， 既可以选择国际通用的标记语言，比如HTML，也可以使用像XML这样由相关人士 自由决定的标记语言，这就是语言的可扩展性。XML是从标准通用标记语言 （SGML）中简化修改出来的。它主要用到的有可扩展标记语言、可扩展样式语言 （XSL）、XBRL和XPath等。

XML具有良好的可读性，有着较好的库支持，从Java语言到其他语言，如Linux系 统上libxml等对XML的支持比较好。

## 网络服务方案选择

### 语言选择

**PHP Laravel**

PHP [[6]](#footnote-48) 是一种开源的通用计算机脚本语言，尤其适用于网络开发并可嵌入 HTML中使用。PHP的语法借鉴吸收了C语言、Java和Perl等流行计算机语言的特点， 易于一般程序员学习。PHP的主要目标是允许网络开发人员快速编写动态页面， 但PHP也被用于其他很多领域。

**Laravel**

Laravel是一套简洁、优雅的PHP Web开发框架。它可以让你从面条一样杂乱的代 码中解脱出来；它可以帮你构建一个完美的网络APP，而且每行代码都可以简洁、 富于表达力。

**Java Spring**

**Java**

Java是一种可以撰写跨平台应用软件的面向对象的程序设计语言，是由Sun Microsystems公司于1995年5月推出的Java程序设计语言。Java 技术具有卓越的 通用性、高效性、平台移植性和安全性，广泛应用于个人PC、数据中心、游戏控 制台、科学超级计算机、移动电话和互联网，同时拥有全球最大的开发者专业社 群。在全球云计算和移动互联网的产业环境下，Java更具备了显著优势和广阔前 景。

**Spring**

Spring是一个开源框架，是为了解决企业应用程序开发复杂性。Spring框架的主 要优势之一就是其分层架构，分层架构允许使用者选择使用哪一个组件，同时为 J2EE 应用程序开发提供集成的框架。Spring使用基本的JavaBean来完成以前只 可能由EJB完成的事情。然而，Spring的用途不仅限于服务器端的开发。从简单 性、可测试性和松耦合的角度而言，任何Java应用都可以从Spring中受益。

## 其它

### 数据通讯设备

**Raspeberry PI**

Raspberry Pi是一款针对电脑业余爱好者、教师、小学生以及小型企业等用户的 迷你电脑，预装Linux系统，体积仅信用卡大小，搭载ARM架构处理器，运算性能 和智能手机相仿。

在接口方面，Raspberry Pi提供了可供键鼠使用的USB接口，此外还有千兆以太 网接口、SD卡扩展接口以及1个HDMI高清视频输出接口，可与显示器或者TV相连。

**Debian**

广义的Debian是指一个致力于创建自由操作系统的合作组织及其作品，由于 Debian项目众多内核分支中以Linux宏内核为主，而且 Debian开发者 所创建的 操作系统中绝大部分基础工具来自于GNU工程 ，因此 “Debian” 常指Debian GNU/Linux。

**Linux**

Linux是一套免费使用和自由传播的类Unix操作系统，是一个基于POSIX和UNIX的 多用户、多任务、支持多线程和多CPU的操作系统。它能运行主要的UNIX工具软 件、应用程序和网络协议。它支持32位和64位硬件。Linux继承了Unix以网络为 核心的设计思想，是一个性能稳定的多用户网络操作系统。

## 辅助语言选择

**Python** Python, 是一种面向对象、直译式计算机程序设计语言，由Guido van Rossum于 1989年底发明，第一个公开发行版发行于1991年。Python语法简洁而清晰，具有 丰富和强大的类库。它常被昵称为胶水语言，它能够很轻松的把用其他语言制作 的各种模块（尤其是C/C++）轻松地联结在一起。常见的一种应用情形是，使用 python快速生成程序的原型（有时甚至是程序的最终界面），然后对其中有特别 要求的部分，用更合适的语言改写，比如3D游戏中的图形渲染模块，速度要求非 常高，就可以用C++重写。

**Ruby**

Ruby，一种为简单快捷的面向对象编程（面向对象程序设计）而创的脚本语言， 在20世纪90年代由日本人松本行弘开发，遵守GPL协议和Ruby License。

Python相对于Ruby有着更好的跨平台能力，同时有理好的可读性，加之Ruby语言 没有对串口通讯及Windows系统更好的支持。

## 串口通信模块

**Pyserial**

封装了串口通讯模块，支持Linux、Windows、BSD(可能支持所有支持POSIX的操 作系统)，支持Jython(Java)和IconPython(.NET and Mono).

在使用PySerial之后，我们只需要

ser=serial.Serial("/dev/ttyACM0",9600)  
 ser.write("1")

就可以向串口发送一个字符1。

## 网页通信

**Ajax** AJAX [[7]](#footnote-54) 是由Jesse James Gaiiett创造的名词，是指一种创建交互式网页 应用的网页开发技术。

系统主要用Ajax来实现实时温度显示，通过直接访问JSON数据的情况下，可以在 不需要刷新页面的情况下直接读取数据。

## 数据可视化框架选择

**HighCharts**

Highcharts 是一个用纯JavaScript编写的一个图表库， 能够很简单便捷的在 web网站或是web应用程序添加有交互性的图表，并且免费提供给个人学习、个人 网站和非商业用途使用。HighCharts支持的图表类型有曲线图、区域图、柱状图、 饼状图、散状点图和综合图表。

**D3.js**

# 本地系统设计

## 硬件设计

### Raspberry Pi

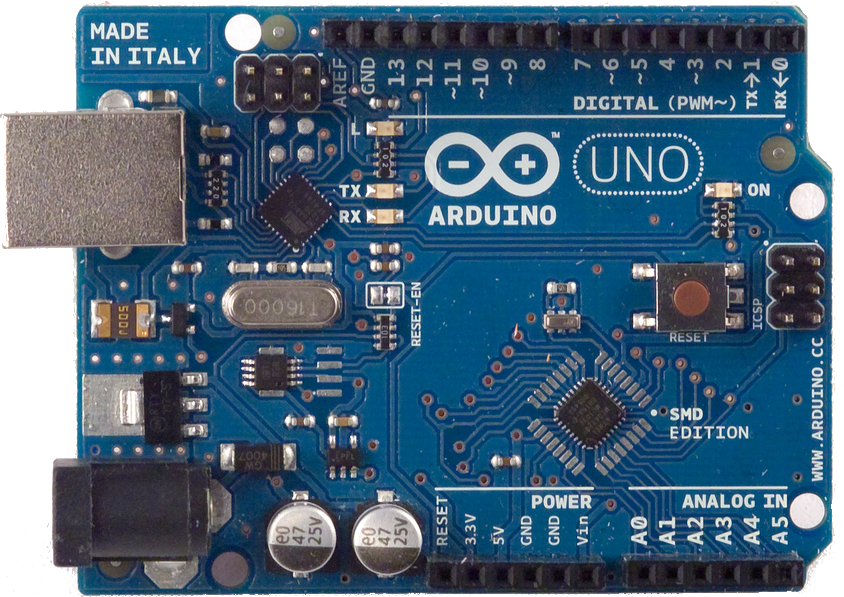
Raspberry Pi开发板与Arduino开发板，通过USB方口线连接。 Raspberry Pi可以直接运行Debian GNU/Linux系统，通过网线上网，并从服务器 中读取数据，同时借由Python语言收发串口数据。

## 软件设计

在本地我们需要解决的问题可以如下描述，Arduino开发板从串口一直读取数据， Raspberry Pi从URL中验证数据、解析数据，再将数据发送到串口,我们可以用下面的伪代码来描述:

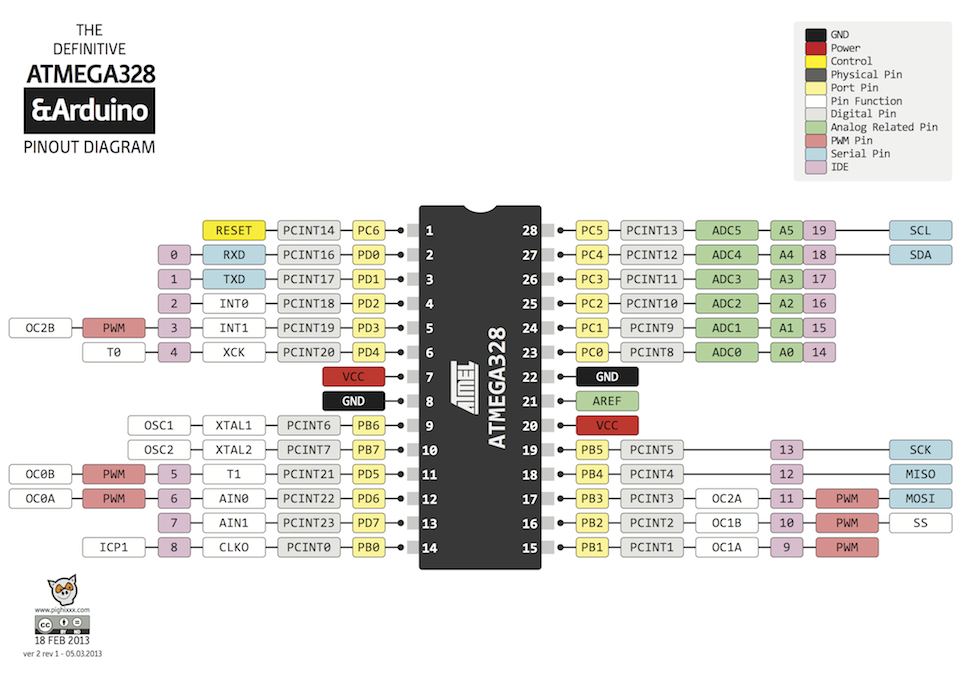
arduino:  
 begin  
 repeat  
 wait(serial.open)  
 data:=receive\_data()  
 led\_status:=parse(data)  
 if led\_status  
 oped(led.id)  
 util false  
 end  
raspberrypi:  
 begin  
 repeat  
 json:=get\_data(url)  
 if validate(json).success()  
 data:=parse（json)  
 serial.write(data)  
 util false  
 end

## Arduino



Arduino开发板

Arduino UNO用的处微控制器是Atmega328，它与Arduino芯片的对应关系如下所示



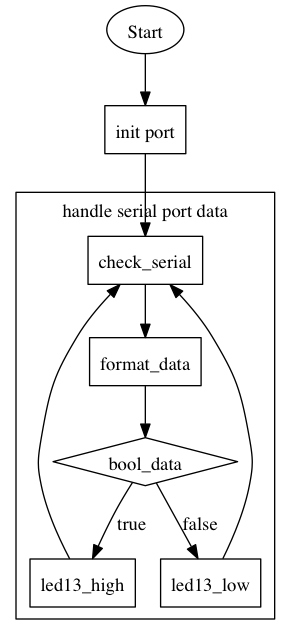
Arduino管脚Atmega328对应图

其主要参数如下所示:

高性能,低功耗的AVR®8位微控制器•先进的RISC结构

* 131条指令
* 绝大多数为单时钟周期执行
* 32个通用工作寄存器
* 全静态工作
* 高达20 MIPS的吞吐量,在20 MHz
* 片上2周期乘法器高耐用性非易失性内存段
* 8K字节的系统内可编程Flash存储器（ATMEGA88PA中）
* 512字节的EEPROM（ATMEGA88PA）
* 1K字节的片内SRAM（ATMEGA88PA）
* 写/擦除次数：10,000次,000 EEPROM
* 数据保存：20年在85°C/100年在25°C(1)
* 可选的引导具有独立锁定位代码段在系统编程的片上引导程序真正的同时读
* 写操作
* 编程软件安全锁外设特点
* 两个8位定时器/计数器具有独立预分频器和比较模式
* 1个16位定时器/计数器具有独立预分频器,比较模式,并捕获模式
* 具有独立振荡器的实时计数器
* 6个PWM通道
* 8通道10位ADC在TQFP和QFN / MLF封装温度测量
* 6通道10位ADC引脚PDIP封装温度测量
* 可编程的串行USART
* 主/从机模式的SPI串行接口
* 面向字节的两线串行接口（飞利浦公司的I2C兼容）
* 独立的片内振荡器的可编程看门狗定时器
* 片上模拟比较器
* 中断和唤醒引脚电平变化单片机的特殊功能
* 上电复位以及可编程的掉电检测
* 内部校准振荡器
* 外部和内部中断源
* 6种睡眠模式：空闲模式,ADC噪声抑制,省电,掉电,待机,扩展Standby

Arduino部分硬件程序如下所示，主要负责从串口中读入数据，并用led灯显示。 程序流程图如下所示



Arduino程序流程图

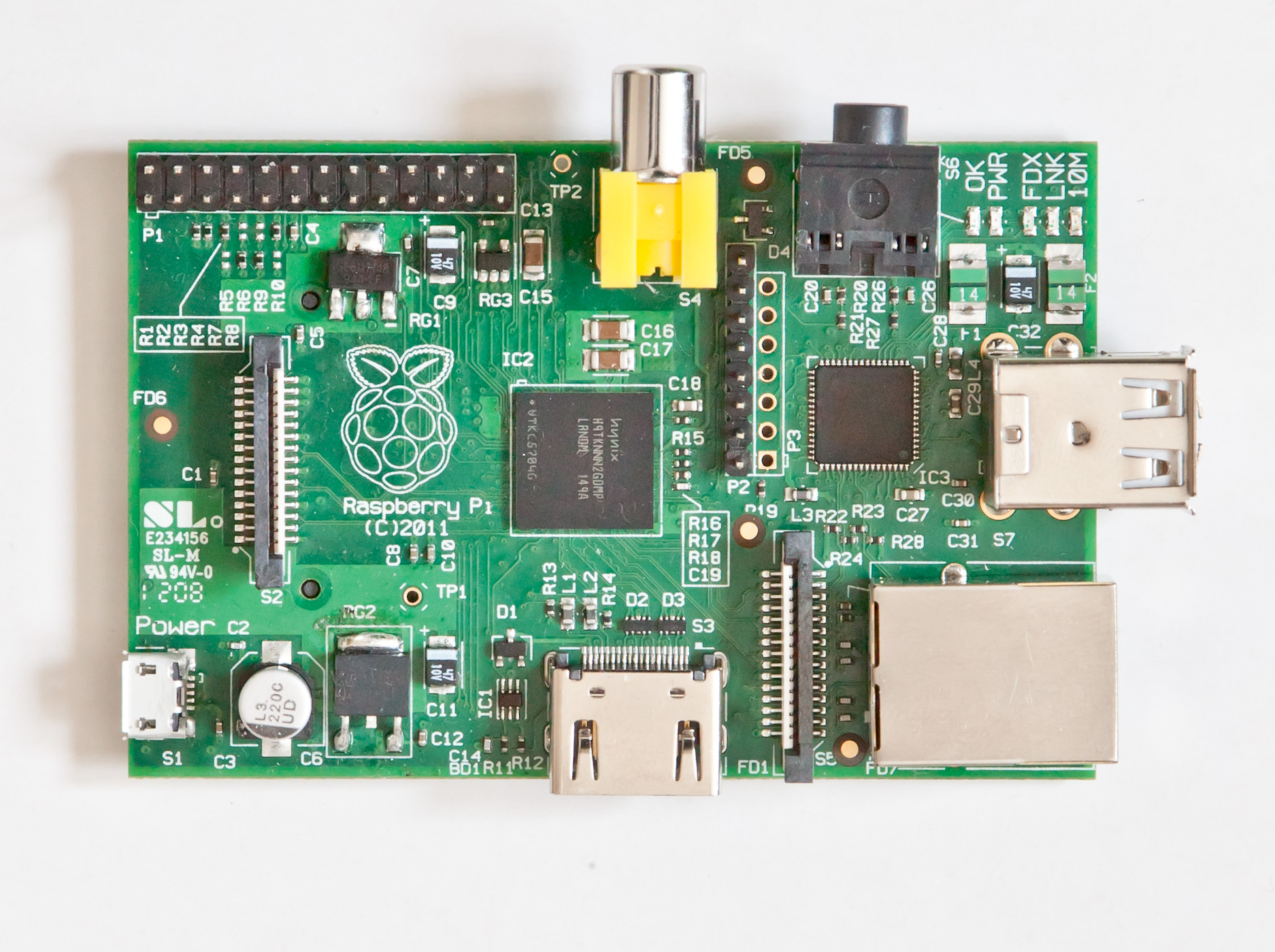
系统主要的功能在于接收和传递数据。

代码如下所示

void setup() {  
 Serial.begin(9600);  
 pinMode(13,OUTPUT);  
 pinMode(12,OUTPUT);  
 }  
  
 int serialData;  
 void loop() {  
 String inString = "";  
 while (Serial.available()> 0)  
 {  
 int inChar = Serial.read();  
 if (isDigit(inChar)) {  
 inString += (char)inChar;  
 }  
 serialData=inString.toInt();  
 Serial.print(serialData);  
 }  
 if(serialData==1){  
 digitalWrite(12,LOW);  
 digitalWrite(13,HIGH);  
 }else{  
 digitalWrite(13,LOW);  
 digitalWrite(12,HIGH);  
 }

## Raspberry Pi

Raspberrypi如下所示的开发板



Raspberry Pi开发板

Raspberry Pi程序

Python 程序流程图

Python程序流程图

### 获取数据

Raspberry Pi端的主要功能便是将数据从 <http://www.xianuniversity.com/athome/1> [^domain] 下 载下来并解析数据，再将数据用串口通讯的方式传递给Arduino。

在Debian系统中，自带了python语言，python有良好的动态特性，同时有强大的自建库功能。 在python语言中可以用自带的urllib2库打开并下载网页的内容，将上述网址中 的JSON数据下载到本地。

数据采用的是JSON格式，具有良好的可读性，同时方便于解析，相比于XML格式 又可以减少文件大小，

[{  
 "id": 1,  
 "temperature": 10,  
 "sensors1": 22,  
 "sensors2": 11,  
 "led1": 0  
 }]

JSON的 将上述中的数据取出来后，通过python中的json库，将json数据转换为数组，将 取出数据中的第一个结果中的id的值。

### 串口通讯

由于python中没有用于串口通讯的库，需要寻找并安装这样一个库，这里就用到 了pip这样的包管理工具——用于管理python的库。

#### 安装pyserial

pip常用命令有install、uninstall以及search，install顾名思义就是安装，安 装pip库如下所示[[8]](#footnote-71)，如后代码如下所示，$[[9]](#footnote-72)开头:

$pip install pyserial

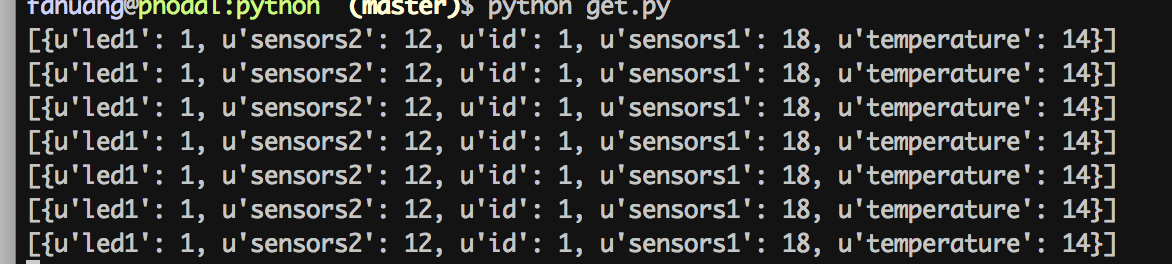
### python串口通讯

在Linux内核的系统[[10]](#footnote-74)中虚拟串口用的节点是ttyACM，位于/dev目录下。

serial.Serial("/dev/ttyACM0",9600)

串行接口是一种可以将接受来自CPU的并行数据字符转换为连续的串行数据流发送出去， 同时可将接受的串行数据流转换为并行的数据字符供给CPU的器件。一般完成这种功能 的电路，我们称为串行接口电路。

便是打开这个设备，以9600的速率传输数据。



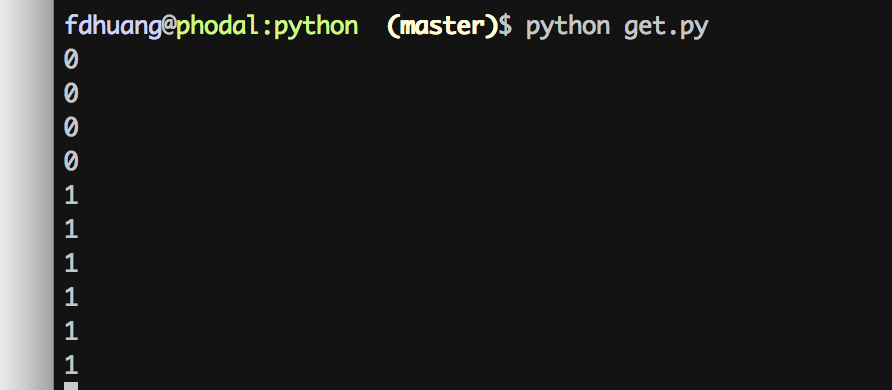
python返回json数据

系统还需要对上面的数据进行处理，只拿其中的结果



python处理完后的结果

当改变led的状态后，便可以得到下面的结果



改变状态后的结果

# 网络系统设计

## 网络服务程序设计

对于物联网系统网络的核心是构建一个RESTful服务，而这构建RESTful的核心便 是基础的HTPP协议。基础的HTTP协议便是:GET、POST、PUT、DELETE。它们分别 对应四种基本操作：GET用来获取资源，POST用来新建资源（也可以用于更新资 源），PUT用来更新资源，DELETE用来删除资源。

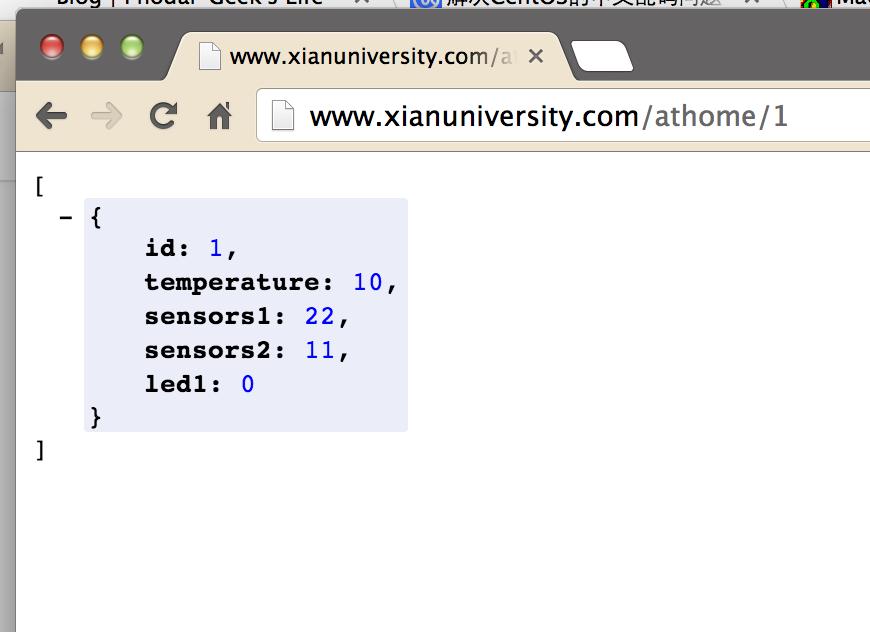
简要的来说，一个GET动作便是在打开一个网页的时候，看到的内容，便是GET到 的资源。而在获取取到网页的内容之前，会有一个POST动作到所要打开的网站的 服务器。

### 基本的REST服务

REST服务实际上是充当着网络与设备的传输介质，构建一个REST服务也就相当于 获取一个URL下的某个数据

$curl http://www.xianuniversity.com/athome/1

返回结果如下所示



JSON结果返回图

假设有这样一个资源用于呈现led的状态，即 http://localhost/status/1 [[11]](#footnote-82)，获取这个LED的状态便发出了类似下面这样的请求:

GET /status/1 HTTP/1.1  
 Host:localhost  
 Cotent-Type:application/json;charset=UTF-8

在客户端发出上述的请求的时候，服务端需要对其做出响应，构造出一个下面的结果

[{  
 "status":1  
 }]

1代表给予灯的状态应该是亮的,在那之后需要做的便是将其通过串口发送给单片 机进行处理，对应于一个关机的结果便是

[{  
 "status":0  
 }]

这样就完成了基本的状态设计。而对于系统最后需要解析的数据的结果来说，还需要加入其他元素，

[{  
 "id": 1,  
 "temperature": 10,  
 "sensors1": 22,  
 "sensors2": 11,  
 "led1": 0  
 }]

这里也涉及到了json数据结构的设计，可以将上面的结果设计为

[{  
 "id": 1,  
 "temperature": 10,  
 "sensors":[{  
 "sensor":22,  
 "sensor":11,  
 }],  
 "led1": 0  
 }]

这种具有更好的可读性，然而在对于网速速度要求高的情况下，会表现得不好， 同时会造成额外的系统开销。对于这样一个需要不断读取数据的系统来说，采用 单层结构的json数据会更具有优势。

在设计这样一个接口的时候，需要考虑客户端可能需要获取全部的数据

GET /status HTTP/1.1  
 Host:localhost  
 Cotent-Type:application/json;charset=UTF-8

设计好这样的接口有助于显示在系统的前台，而这也是无法在物联网系统中产生 统一协议的原因之一，复杂的接口无法用于简单功能的场景。

POST / HTTP/1.1  
 Host: example.com  
 User-Agent: Go 1.1 package http  
 Content-Length: 45  
 Authorization: 123456  
 Accept-Encoding: gzip

一个PUT动作但是我们更新资源，就好比是我们创建一个日志或者一个说说一样。 DELETE动作，便是删除动作了，而这也是一个物联网系统服务所需要的。

## 系统前台设计

在对系统前台设计的时候，在考虑不同移动设备的兼容的同时，也需要保持一个 良好可用的结构。而系统在前台的主要功能是在于控制物体的状态、显示一些数 值的变化，控制物体状态的关键在于如何将数据由前台POST到后台，在网页端可 以用POST，而在移动端则可以用JSON API。

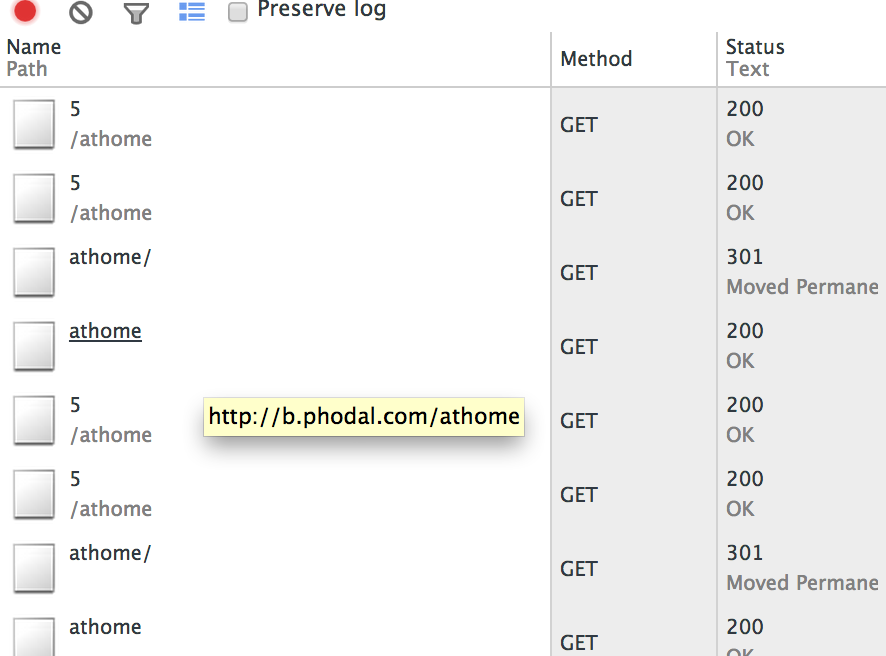
### Ajax

* AJAX : Asynchronous JavaScript and XML（异步的 JavaScript 和 XML）。
* AJAX 不是新的编程语言，而是一种使用现有标准的新方法。
* AJAX 是与服务器交换数据并更新部分网页的艺术，在不重新加载整个页面的情况下。

剥离后的Ajax部分代码如下所示，主要用的是 jQuery 框架的 getJSON 来实现的

begin  
 data:=get\_data(url)  
 if data.get\_success  
 temperature:=data.push(temperature)

系统会不断从后台获取数据结果，如下所示



后台获取数据

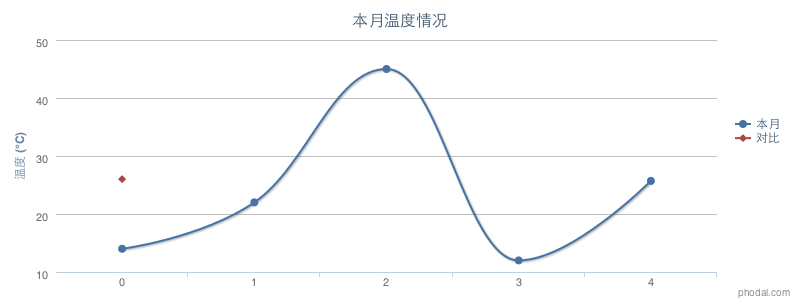
在Javascript语言中有函数库可以直接用于获取后台数据——getJSON，可以从指定的URL中获取结果。

* url 用于提供 json 数据的地址页
* data(Optional) 用于传送到服务器的键值对
* callback(Optional) 回调函数，json 数据请求成功后的处理函数

var dataLength = [];  
 function drawTemp() {  
 var zero = [];  
 $.getJSON('/athome/', function(json) {  
 var items = [];  
 dataLength.push(json.length);  
 $.each(json, function(key, val) {  
 zero.push(val.temperature);  
 });  
 };

实际上，我们做的只是从 /athome/ 下面获取数据，再将数据堆到数组里面，再 把这部分放到图形中。

* 兼容性：兼容当今所有的浏览器，包括 iPhone、IE 和火狐等等；
* 对个人用户完全免费；
* 纯JS，无BS；
* 支持大部分的图表类型：直线图，曲线图、区域图、区域曲线图、柱状图、饼装图、散布图；
* 跨语言：不管是 PHP、Asp.net 还是 Java 都可以使用，它只需要三个文件：一个是Highcharts 的核心文件 highcharts.js，还有 a canvas emulator for IE 和 Jquery类库或者 MooTools 类库；
* 提示功能：鼠标移动到图表的某一点上有提示信息；
* 放大功能：选中图表部分放大，近距离观察图表；
* 易用性：无需要特殊的开发技能，只需要设置一下选项就可以制作适合自己的图表；
* 时间轴：可以精确到毫秒;



chart

## 系统后台设计

系统使用Laravel框架作为系统底层，需要配置其运行环境[[12]](#footnote-88)，创建数 据库[[13]](#footnote-89)，对应于上面生成的最后的JSON数据，创建对应的数据库:

|  |
| --- |
| |表名 |数据类型| |
| |id |整数 | |
| |temperature |浮点数 | |
| |sensors1 |浮点数 | |
| |sensors2 |浮点数 | |
| |led1 |布尔型 | |

public function up()  
 {  
 Schema::create('athomes', function(Blueprint $table)  
 {  
 $table--->increments('id');  
 $table->float('temperature');  
 $table->float('sensors1');  
 $table->float('sensors2');  
 $table->boolean('led1');  
 $table->timestamps();  
 });  
 }

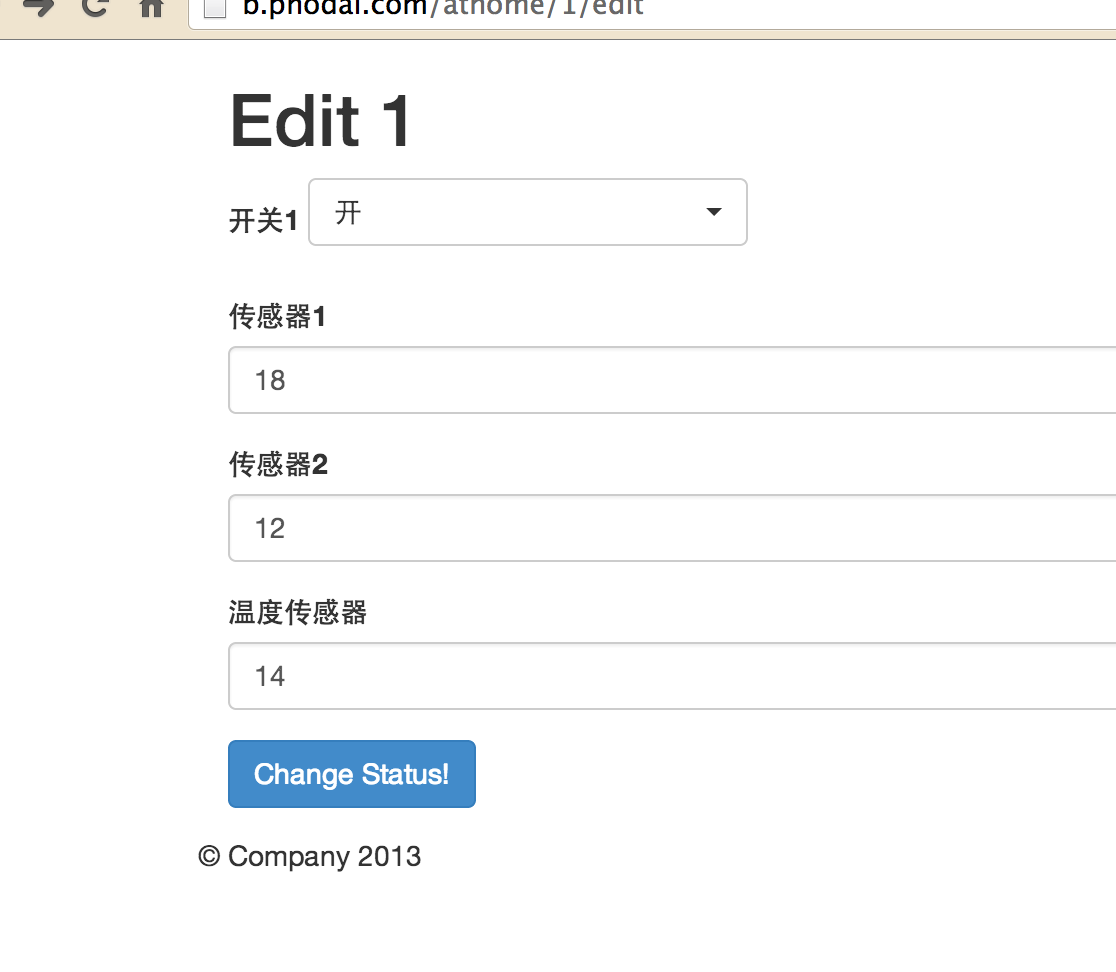
当POST数据的时候，便是将数据存往数据库，而GET的时候则是从数据库中拿出 数据再渲染给浏览器，GET、PUT、DELETE、POST便是对就于数据库的Create、Refresh、Update、Delete

系统使用MySQL作为数据库，开始的时候需要创建数据库，在数据库中执行

CREATE DATABASE IF NOT EXISTS iot default charset utf8 COLLATE utf8\_general\_ci;

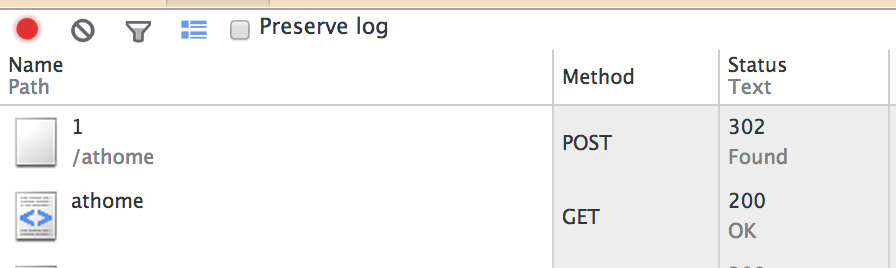
以创建athomes这样一个数据库，同时将PHP与数据库配置好

'mysql' => array(  
 'driver' => 'mysql',  
 'host' => 'localhost',  
 'database' => 'iot',  
 'username' => 'root',  
 'password' => ' ',  
 'charset' => 'utf8',  
 'collation' => 'utf8\_unicode\_ci',  
 'prefix' => '',  
)



控制界面

当按下Change Status按钮时，系统发生了如下变化



GET POST数据

系统会先向服务器发送数据，也就是POST请求，在请求结束后，系统将会刷新页面，也就是GET请求。

1. 全称单片微型计算机（英语：Single-Chip Microcomputer） [↑](#footnote-ref-34)
2. Representational State Transfer [↑](#footnote-ref-38)
3. JavaScript Object Notation [↑](#footnote-ref-42)
4. 包括C, C++, C#, Java, JavaScript, Perl, Python等 [↑](#footnote-ref-43)
5. eXtensible Markup Language，简称: XML [↑](#footnote-ref-45)
6. PHP：Hypertext Preprocessor，即超文本预处理器 [↑](#footnote-ref-48)
7. Asynchronous JavaScript and XML（异步JavaScript和XML) [↑](#footnote-ref-54)
8. 在Windows系统中需要先安装pip，再安装pyserial。 [↑](#footnote-ref-71)
9. 指在\*nix系统的终端中执行的命令。 [↑](#footnote-ref-72)
10. 在Windows系统上，只需要将/dev/ttyACM0改为对应的com口。

    import json import urllib2 import serial import time

    url=“http://www.xianuniversity.com/athome/1”

    while 1: try: date=urllib2.urlopen(url) result=json.load(date) status=result[0][“led1”] ser=serial.Serial(“/dev/ttyACM0”,9600) if status==1 : ser.write(“1”) elif status==0: ser.write(“0”) time.sleep(1) except urllib2.URLError: print “Bad URL or timeout” [↑](#footnote-ref-74)
11. 在本地进行web开发时，浏览器可以识别localhost，配置好Hosts时相当于127.0.0.1。 [↑](#footnote-ref-82)
12. 这里用的是Linux+Nginx+MySQL+PHP。 [↑](#footnote-ref-88)
13. 创建数据的代码:CREATE DATABASE IF NOT EXISTS bbs default charset utf8 COLLATE utf8\_general\_ci; [↑](#footnote-ref-89)