



太原理工大学  
TAIYUAN UNIVERSITY OF TECHNOLOGY

## 课程设计

课程名称： 物联网感知课程设计

设计名称： 基于物联网的智能家居系统

专业班级： 物联网 1502 学号： 2015003588

学生姓名： 刘港

指导教师： 李爱萍

2018 年      1 月      14 日

## 太原理工大学课程设计任务书

学生姓名	刘港	专业班级	物联网 1502
课程名称	物联网感知课程设计		
设计名称	基于物联网的智能家居系统	设计周数	2
设计任务主要设计参数	<p>运用已学的编程工具及物联网实验箱设计一个《基于物联网的智能家居》系统，该系统可实现如下功能：</p> <ol style="list-style-type: none"> <li>1. 能够完成温度、湿度、光照强度、触摸器数据的采集</li> <li>2. 能够将采集的数据保存至本地和云端数据库</li> <li>3. 能够多元化地制作相应的显示界面，将采集的变量值实时显示并绘制相应的图形，显示界面包括本地显示和远程端 web 显示</li> <li>4. 能够根据传感器的采集的数据进行智能化控制，包括空调，加湿器的启动和关闭，窗帘和灯光的智能控制</li> </ol> <p>设计的基本要求如下：</p> <ol style="list-style-type: none"> <li>1. 能够正确地读取相应传感器的数据</li> <li>2. 能将采集的数据做正确处理和显示</li> </ol> <p>设计的扩展要求如下：</p> <ol style="list-style-type: none"> <li>1. 将采集的数据保存至本地和云端数据库</li> <li>2. 多元化地制作相应的显示界面，将采集的变量值实时显示并绘制相应的图形，显示界面包括本地显示和远程端 web 显示</li> <li>3. 根据传感器的采集的数据进行智能化控制，包括空调，加湿器的启动和关闭，窗帘和灯光的智能控制</li> </ol>		
设计内容设计要求	<ol style="list-style-type: none"> <li>1、收集技术资料：理解设计任务、查阅相关资料、搭建开发平台。</li> <li>2、确定总体设计思想：方案论证比较、确定总体设计方案。</li> <li>3、感知层和传输层的设计：感知模块的选择、传输模式与接口程序的设计。</li> <li>4、应用层的设计：各模块的功能说明、程序流程、代码编写。</li> <li>5、系统调试与运行：验证各项功能的实现。</li> <li>6、书写设计报告：按照课程设计报告的要求，编写设计报告。</li> </ol>		
主要参考资料	<ol style="list-style-type: none"> <li>1、徐勇军. 物联网实验教程. 北京：机械工业出版社. 2011</li> </ol>		

指导教师签名：

日期：

目录

功能分析.....4

概要设计.....4

详细设计.....6

    硬件连接 .....6

    传感器程序设计 .....6

    端口数据读取设计 .....7

    端口数据数据库存储设计 .....8

    数据智能处理设计 .....9

    界面设计 ..... 17

系统说明..... 23

设计总结..... 23

附录 ..... 23

# 课程设计报告：基于物联网的智能家居系统

## 程序设计课程设计报告

专业：物联网工程 班级：1502 姓名：刘港 学号：2015003588 完成日期：2018.1.14

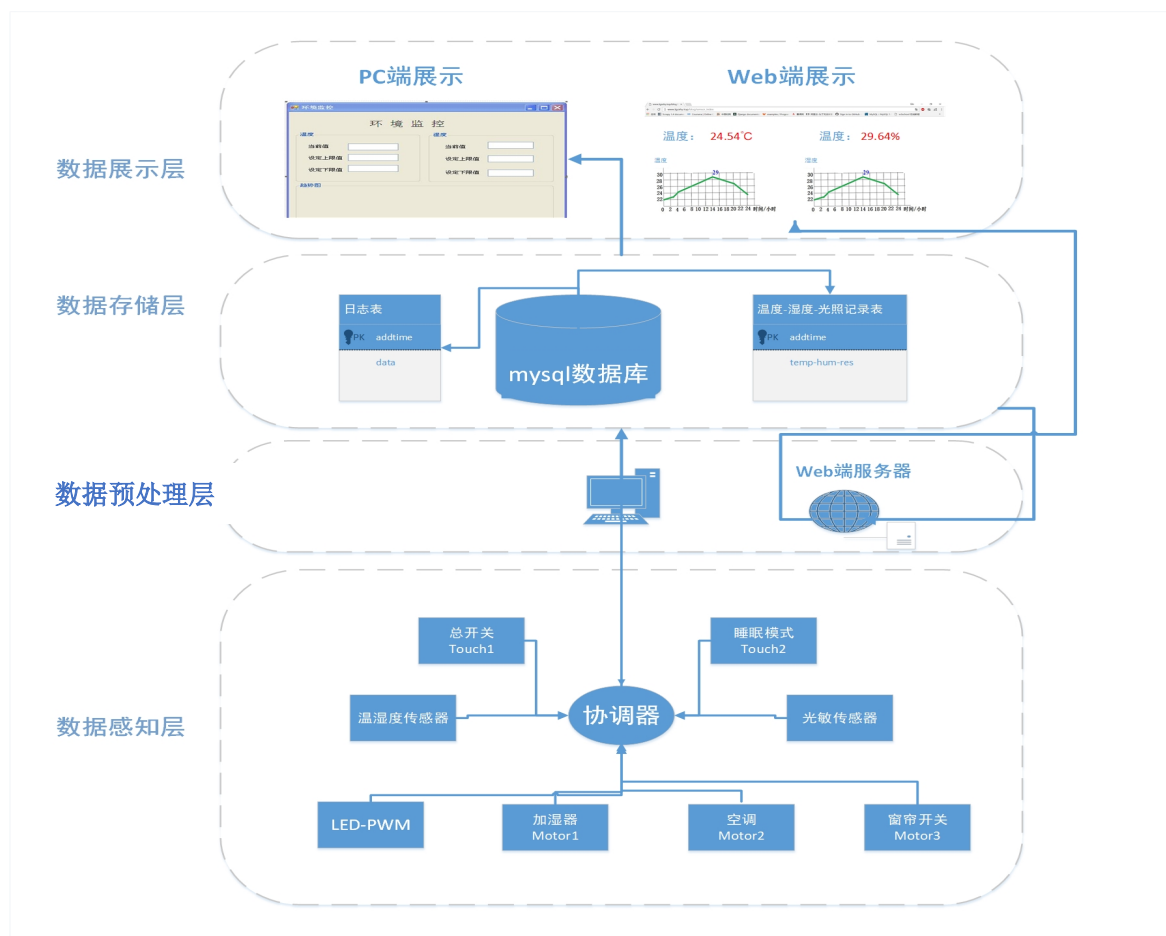
### 一. 功能分析

智能家居系统主要包括对环境变量的采集、显示和控制。

1. 能够完成温度、湿度、光照强度、触摸器数据的采集
2. 能够将采集的数据保存至本地和云端数据库
3. 能够多元化地制作相应的显示界面，将采集的变量值实时显示并绘制相应的图形，显示界面包括本地显示和远程端 web 显示
- 4 能够根据传感器的采集的数据进行智能化控制，包括空调，加湿器的启动和关闭，窗帘和灯光的智能控制

### 二. 概要设计

#### 1. 系统组成



## 2. 数据库设计

Sensorinfo 当前各传感器状态表

Temp\_hum 温湿度传感器数据表

Photores 光敏传感器数据表

Log 端口数据日志表

```
mysql> desc sensorinfo;
```

Field	Type	Null	Key	Default	Extra
id	int(11)	YES		0	
flag	varchar(10)	YES		OPEN	
sleep_flag	varchar(10)	YES		CLOSE	
set_temp	float	YES		30	
set_hum	float	YES		30	
temp	float	YES		0	
hum	float	YES		0	
ill	float	YES		0	
motor1_status	varchar(10)	YES		CLOSE	
motor2_status	varchar(10)	YES		CLOSE	

10 rows in set (0.06 sec)

```
mysql> desc temp_hum;
```

Field	Type	Null	Key	Default	Extra
id	int(10) unsigned	NO	PRI	NULL	auto_increment
temp	float	YES		NULL	
hum	float	YES		NULL	
addtime	datetime	YES		NULL	

4 rows in set (0.00 sec)

```
mysql> desc photores;
```

Field	Type	Null	Key	Default	Extra
id	int(10) unsigned	NO	PRI	NULL	auto_increment
illumination	float	YES		NULL	
addtime	datetime	YES		NULL	

3 rows in set (0.09 sec)

```
mysql> desc log;
```

Field	Type	Null	Key	Default	Extra
addtime	datetime	YES		NULL	
data	varchar(40)	YES		NULL	

2 rows in set (0.00 sec)

### 3. 传感器选型

选择 SHT10 单片数字温湿度、光敏、触摸传感器来监测环境。SHT10 采用 CMOSens 专利技术将温度湿度传感器等、A/D 转换器及数字接口无缝结合，使传感器具有体积小、响应速度快、接口简单、性价比高等特点。

电路连接时，接口简单，两线连接，SHT10 的两线串行接口（bidirectional 2-wire）在传感器信号读取和电源功耗方面都做了优化处理，其总线类似 I2C 总线但并不兼容 I2C 总线。

### 4. 界面设计

Web 端采用 python 开源框架 django

Pc 端采用 python Tkinter 和 matplotlib

## 三. 详细设计

### 1. 硬件连接

SHT10 共有 4 个引脚，除了电源和地，数据和时钟引脚分别连接单片机 P06、P07。

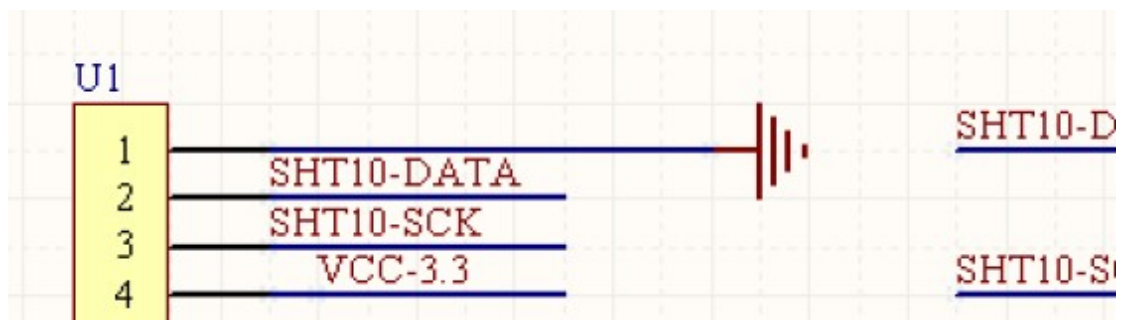


图 2 系统硬件框图

### 2. 相关传感器程序设计

(1) 端口初始化程序

```
static void GPIOInit(void)
{
    POSEL &= ~(1<<6); //时钟接口初始化，IO 口功能选择
    PODIR |= (1<<6); //IO 口方向选择
    POSEL &= ~(1<<7); //数据接口初始化
```

```
PODIR |= (1<<7); //IO 口方向选择
}
```

功能：初始化 SHT10 接口； 入口参数：无； 返回值：无。

## (2) 传感器采集程序

```
static uint16 ReadSHT10(uint8 param)
{
double temp;
uint8 i;
uint16 result;
uint16 SORH = 0;
DATA_OUTPUT;
DATA_HIGH;
SCK_OUTPUT;
SCK_LOW;
//通讯复位
for( i=0; i<10; i++ )
{
SCK_HIGH;
MicroWait(30);
SCK_LOW;
MicroWait(30);
}
.....
```

功能：读取 SHT10 的温湿度值； 入口参数：0x00，温度； 0x01，湿度； 返回值：温湿度值。

## 3. 端口数据读取程序设计：

通过 USB 串口连接协调器和 PC 机

```
#coding=utf8
'''*****
*模块： 协调器
*主要功能： 1.作为其他传感器的网关，与 pc 机进行连接交互
            2.采集管理各传感器信息
*****'''
import serial
import binascii
import mysqlcon as mc
import time
'''*****
*函数： CoorConnectPc
*主要功能： 与 PC 机进行端口连接
*相关参数： 端口： COM3
            波特率： 115200
```

```

        数据位: 8
        停止位: 1
        校验位: None
*****'''
def CoorConnectPc():
    ser = serial.Serial(
        port='COM3',
        baudrate=115200,
        bytesize = 8,
        stopbits = 1,
        parity = 'N',
    )
    return ser

```

从端口读取数据

```

'''*****
*函数:  ReadPort
*主要功能:  端口数据读取
*****'''
def ReadPort(ser):
    data = ''
    data=str(binascii.b2a_hex(ser.read(16)))
    while 1 :
        if (data)!='':
            print data
            insertlog(data)
            break
    return data

```

## 4. 端口数据存储数据库设计

数据库的连接

```

#coding=utf8
'''*****
*模块:  mysqlcon
*主要功能:  本地 mysql 数据库连接
*****'''
import pymysql
def mysqlconnect():
    con=pymysql.connect(
        host='localhost',
        port=3306,
        user='root',

```



```

        passwd='liugang666',
        database='sensor',
        charset='utf8',
    )
    return con
'''*****'''
*模块:  linux_mysqlcon
*主要功能:  远程 mysql 数据库连接
*****'''
def linux_mysqlconnect():
    con=pymysql.connect(
        host='59.110.159.69',
        port=3306,
        user='root',
        passwd='liugang666',
        database='sensor',
        charset='utf8',
    )
    return con

```

## 5. 数据库数据智能处理设计

### 温湿度模块

```

#coding=utf8
'''*****'''
*模块:  温湿度模块
*主要功能:  1.采集温度和湿度
            2.将采集到的温度湿度保存到数据库中
*****'''
import mysqlcon as mc
import time
import Motor as mt
'''*****'''
*函数:  temp_hum_store
*主要功能:  将温湿度数据进行转换, 并存入数据库中
*****'''
def temp_hum_store(data):
    xh=float(int(data[10:12],16))
    xl=float(int(data[12:14],16))
    temp=(xh*256+xl)/100
    ph=float(int(data[14:16],16))
    pl=float(int(data[16:18],16))
    hum=(ph*256+pl)/100

```

```

        con=mc.mysqlconnect()
        cursor=con.cursor()
        cursor.execute('insert into temp_hum(temp,hum,addtime)
values(%s,%s,%s)',(str(temp),str(hum),time.strftime("%Y-%m-%d %H:%M:%S",
time.localtime()))))
        con.commit()
        #更新状态

        cursor.execute('update sensorinfo set temp=%s,hum=%s where
id=0',(str(temp),str(hum)))
        con.commit()
        cursor.close()
        con.close()
        #linux mysql
        linux_con=mc.linux_mysqlconnect()
        linux_cursor=linux_con.cursor()
        linux_cursor.execute('insert into temp_hum(temp,hum,addtime)
values(%s,%s,%s)',(str(temp),str(hum),time.strftime("%Y-%m-%d %H:%M:%S",
time.localtime()))))
        linux_con.commit()
        linux_cursor.close()
        linux_con.close()
        temp_hum={'temp':temp,'hum':hum}
        print 'temp: ',temp,' hum: ',hum
        return temp_hum

'''*****
*函数:   judge_temp_hum
*相关参数:   ser  端口句柄
              now_temp_hum  现在的温度和湿度
              set_temp_hum  设定的触发温湿度
*主要功能:   当温度大于设定值打开电机 2 空调
              小于设定温度时关闭
              当湿度低于设定值时打开电机 1 加湿器
              大于设定值时关闭
*****'''
def judge_temp_hum(ser,now_temp_hum,set_temp_hum):
    #开空调
    if now_temp_hum['temp'] > set_temp_hum['min_temp']:
        mt.opp_motor(ser,'open','02')
    else:
        mt.opp_motor(ser,'close','02')
    #开加湿器
    if now_temp_hum['hum'] < set_temp_hum['min_hum']:

```

```

        mt.opp_motor(ser,'open','01')
    else:
        mt.opp_motor(ser,'close','01')

```

## 光敏模块

```

#coding=utf8
'''*****'''
*模块: 光敏模块
*主要功能: 1.采集光照强度
           2.将采集到的光照强度保存到数据库中
*****'''
import mysqlcon as mc
import time
import PWM as pwm
'''*****'''
*函数: temp_hum_store
*主要功能: 将光敏数据进行转换, 并存入数据库中
*返回值: 光照强度
*****'''
def photores_store(data):
    ph=float(int(data[10:12],16))
    pl=float(int(data[12:14],16))
    p=ph*256+pl
    con=mc.mysqlconnect()
    cursor=con.cursor()
    cursor.execute('insert into photores(illumination,addtime)
values(%s,%s)',(str(p),time.strftime("%Y-%m-%d %H:%M:%S",
time.localtime()))))
    con.commit()
    #更新状态
    cursor.execute('update sensorinfo set ill=%s where id=0',str(p))
    con.commit()

    cursor.close()
    con.close()
    #linux mysql
    linux_con=mc.linux_mysqlconnect()
    linux_cursor=linux_con.cursor()
    linux_cursor.execute('insert into photores(illumination,addtime)
values(%s,%s)',(str(p),time.strftime("%Y-%m-%d %H:%M:%S",
time.localtime()))))
    linux_con.commit()
    linux_cursor.close()

```

```

linux_con.close()
print 'light: ',p,'xl'
return p

'''*****
*函数:  dimming
*相关参数:  light 光照强度
*主要功能:  根据光照强度进行 LED 调光
*****'''
def dimming(ser,light):
    if light<50 :
        pwm.led_light(ser,'09')
    elif light<100:
        pwm.led_light(ser,'07')
    elif light<200:
        pwm.led_light(ser,'05')
    elif light<300:
        pwm.led_light(ser,'03')
    else:
        pwm.led_light(ser,'00')

```

## 触摸模块

```

#coding=utf8
'''*****
*模块:  触摸模块
*主要功能:  1.智能家居总开关 touch1
            2.睡眠模式开关 touch2
*****'''
import mysqlcon as mc
'''*****
*函数:  touch_check
*相关参数:  num 选择总开关或者睡眠开关
*主要功能:  根据 num 来选择开关的开和关
*****'''
def touch_check(ser,num,flag,sleep_flag):
    #总开关触摸传感器
    if num=='01':
        flag=not flag
        if flag==True:
            f1='OPEN'
        else:
            f1='CLOSE'
    #更新状态

```

```

con=mc.mysqlconnect()
cursor=con.cursor()
cursor.execute('update sensorinfo set flag=%s where id=0',f1)
con.commit()
#睡眠开关传感器
if num=='02':
    sleep_flag=not sleep_flag
    if sleep_flag==True:
        f2='OPEN'
    else:
        f2='CLOSE'
con=mc.mysqlconnect()
cursor=con.cursor()
cursor.execute('update sensorinfo set sleep_flag=%s where id=0',f2)
con.commit()
cursor.close()
con.close()

```

## 数字调光模块

```

#coding=utf8
'''*****
*模块： 数字调光模块
*主要功能： 根据光照强度进行自动 LED 调光
*****'''
import binascii

'''*****
*函数： led_light
*相关参数： num 亮度调节值
*主要功能： 根据 num 亮度值进行 LED 调光
*****'''
def led_light(ser,num):
    senddata='ccee010a01'+num+'0000000000000000ff'
    senddata=binascii.b2a_hex(senddata)
    ser.write(senddata)

```

## 电机和灯光控制模块

```

#coding=utf8
'''*****
*模块： 电机及灯光控制模块
*主要功能： 1.控制电机的转动
              2.控制电机的状态灯光

```

```

*****'''
import binascii
import time
import mysqlcon as mc
'''*****

*函数:  op_motor
*主要功能:  电机启动和停止
*相关参数:  ser 发送端口连接句柄
            op  选择启动或停止操作
            num 选择电机号
*****'''

def opp_motor(ser,op,num):
    con=mc.mysqlconnect()
    cursor=con.cursor()

    if op=='open':
        senddata='ccee'+num+'090b000000000000000000ff'
        senddata=binascii.b2a_hex(senddata)
        try:
            ser.write(senddata)
            cursor.execute('update sensorinfo set motor%s_status=%s where
id=0',(num[: -1], 'OPEN'))
            con.commit()
            print 'motor '+num+' start success!'
        except:
            print 'motor '+num+' start failed!'

    if op=='close':
        senddata='ccee'+num+'090b000000000000000000ff'
        senddata=binascii.b2a_hex(senddata)
        try:
            ser.write(senddata)
            cursor.execute('update sensorinfo set motor%s_status=%s where
id=0',(num[: -1], 'CLOSE'))
            con.commit()
            print 'motor '+num+' close success!'
        except:
            print 'motor '+num+' close failed!'

    cursor.close()
    con.close()
'''*****

*函数:  status_motor
*主要功能:  电机状态灯光控制
*相关参数:  ser 发送端口连接句柄

```

```

        data 电机回复数据
*功能： 1.当电机回复数据为 EE CC NO 09 DD 09 时
        开启 LED1 灯
        2.当电机回复数据为 EE CC NO 09 DD 0b 时
        关闭 LED1 灯
        LED1 灯为电机状态灯，电机开启时为亮状态
*****'''
def status_motor(ser,data):
    num=data[4:6]
    if data[10:12]=='09':
        senddata='ccee'+num+'0901000000000000000000ff'
        senddata=binascii.b2a_hex(senddata)
        ser.write(senddata)
        print 'motor ',num,' light open!'
    if data[10:12]=='0b':
        senddata='ccee'+num+'0902000000000000000000ff'
        senddata=binascii.b2a_hex(senddata)
        ser.write(senddata)
        print 'motor ',num,' light close!'
'''*****
*函数： time_motor
*主要功能： 电机在开启设定的时间后自动关闭
*相关参数： ser 发送端口连接句柄
            num 选择电机
            set_time 自动关闭延迟时间
*****'''
def time_motor(ser,num,set_time):
    opp_motor(ser,'open',num)
    time.sleep(set_time)
    opp_motor(ser,'close',num)

```

## 程序主入口模块

```

#coding=utf8
'''*****
*模块： 程序主入口
*****'''
import time
import string
import mysqlcon as mc
import Coordinator as ci
import Motor as mt
import TempAndHum as th
import PhotoResistor as pr

```

```

import Touch as tc
import thread
from Display import *

'''*****
*函数: dealdata
*主要功能: 1.实时接收数据并存入数据库
           2.接收到有效数据后进行状态更新
*****'''
def dealdata():
    set_temp_hum={}
    set_temp_hum['min_temp']=input("please input min start temp:\n")
    set_temp_hum['min_hum']=input("please input min start hum:\n")

    #设置温度值更新
    con=mc.mysqlconnect()
    cursor=con.cursor()
    cursor.execute('update sensorinfo set set_temp=%s,set_hum=%s where
id=0',(set_temp_hum['min_temp'],set_temp_hum['min_hum']))
    con.commit()
    cursor.close()
    con.close()

    flag=True
    sleep_flag=False
    ser=ci.CoorConnectPc()
    while 1 :
        data=ci.ReadPort(ser)

        if data[8:10]!='aa':
            #温湿度模块
            if data[6:8]=='03':
                now_temp_hum=th.temp_hum_store(data)
                print 'now temp
hum:',now_temp_hum['temp'],now_temp_hum['hum']
                print 'set temp
hum:',set_temp_hum['min_temp'],set_temp_hum['min_hum']
                if flag==True:
                    th.judge_temp_hum(ser,now_temp_hum,set_temp_hum)

            #光敏模块
            if data[6:8]=='02':
                light=pr.photores_store(data)
                #智能开关打开且不是睡眠模式

```



```

        if flag==True and sleep_flag==False:
            pr.dimming(ser,light)

#触摸模块
#有触摸时
if flag==True and data[6:8]=='0e' and data[10:12]=='01':
    tc.touch(ser,data[8:10],flag,sleep_flag)
    if sleep_flag==True:
        #关闭全部灯
        pwm.led_light(ser,'00')
        #关窗帘
        mt.time_motor(ser,'03',3)

#电机模块
if flag==True and data[6:8]=='09':
    if data[8:9]=='dd':
        mt.status_motor(ser,data)
'''*****'''
*函数:  main
*主要功能:  程序主入口函数
*****'''
if __name__ == '__main__':
    #多线程处理数据函数
    thread.start_new_thread(dealdata,())
    #图形界面
    thread.start_new_thread(windows,())
    time.sleep(10000)

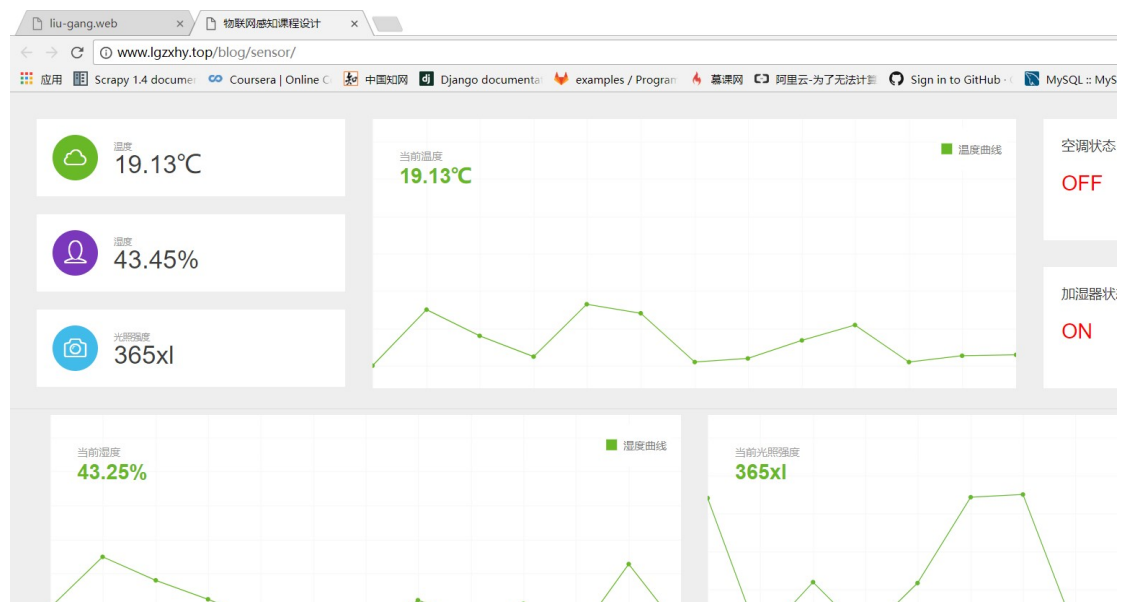
```

## 6. 界面设计

PC 端界面图



WEB 端界面图



本地图形界面设计源程序

```
#coding=utf8
'''
*****
*模块： 图形界面
*主要功能： 1.实时显示各个传感器的信息
              2.实时刷新温湿度信息
              3.实时显示传感器曲线
*****
'''
import matplotlib
matplotlib.use('TkAgg')
from matplotlib.backends.backend_tkagg import FigureCanvasTkAgg
from matplotlib.figure import Figure
```

```

from Tkinter import *
import mysqlcon as mc
import pandas as pd

'''*****
*函数： flush
*主要功能： 1.从数据库中读取各传感器状态
            2.将数据进行实时刷新
*相关参数： flag,sleep_flag 触摸开关
            set_temp,set_hum 设定温湿度
            temp,hum,ill 温湿度光照强度
            mo1_st,mo2_st 空调加湿器
*****'''
def
flush(flag,sleep_flag,set_temp,set_hum,temp,hum,ill,mo1_st,mo2_st,root)
:
    con=mc.mysqlconnect()
    realtime_data_sql='select * from sensorinfo'
    realtime_data=pd.read_sql(realtime_data_sql,con)
    con.close()
    flag.set(realtime_data['flag'][0])
    sleep_flag.set(realtime_data['sleep_flag'][0])
    set_temp.set(realtime_data['set_temp'][0])
    set_hum.set(realtime_data['set_hum'][0])
    temp.set(realtime_data['temp'][0])
    hum.set(realtime_data['hum'][0])
    ill.set(realtime_data['ill'][0])
    mo1_st.set(realtime_data['motor1_status'][0])
    mo2_st.set(realtime_data['motor2_status'][0])
    #500ms 刷新数据

root.after(500,flush,flag,sleep_flag,set_temp,set_hum,temp,hum,ill,mo1_
st,mo2_st,root)

'''*****
*函数： draw
*主要功能： 绘制并刷新温湿度和光照强度曲线
*相关参数： root 图形根界面
*****'''
def draw(root):
    #读取温湿度光照强度数据 10 条
    con=mc.mysqlconnect()
    recent_10_temphum_data_sql='select temp,hum,addtime from temp_hum
group by id desc limit 10'

```

```

recent_10_temphum_data=pd.read_sql(recent_10_temphum_data_sql,con)
recent_10_ill_data_sql='select illumination,addtime from photores
group by id desc limit 10'
recent_10_ill_data=pd.read_sql(recent_10_ill_data_sql,con)
con.close()
#创建画板并进行图形绘制
f = Figure(figsize=(10,4), dpi=100)
temp_photo=f.add_subplot(221)
hum_photo=f.add_subplot(222)
ill_photo=f.add_subplot(223)

th_x=recent_10_temphum_data['addtime']

temhum_x=[th_x[9],th_x[8],th_x[7],th_x[6],th_x[5],th_x[4],th_x[3],th_x[
2],th_x[1],th_x[0]]

ty=recent_10_temphum_data['temp']
temp_y=[ty[9],ty[8],ty[7],ty[6],ty[5],ty[4],ty[3],ty[2],ty[1],ty[0]]

hy=recent_10_temphum_data['hum']
hum_y=[hy[9],hy[8],hy[7],hy[6],hy[5],hy[4],hy[3],hy[2],hy[1],hy[0]]

ix=recent_10_ill_data['addtime']
i_x=[ix[9],ix[8],ix[7],ix[6],ix[5],ix[4],ix[3],ix[2],ix[1],ix[0]]

iy=recent_10_ill_data['illumination']
i_y=[iy[9],iy[8],iy[7],iy[6],iy[5],iy[4],iy[3],iy[2],iy[1],iy[0]]

#设置横纵坐标
temp_photo.plot(th_x,temp_y)
hum_photo.plot(th_x,hum_y)
ill_photo.plot(i_x,i_y)

temp_photo.set_xlabel('time')
temp_photo.set_ylabel('temp')
hum_photo.set_xlabel('time')
hum_photo.set_ylabel('hum')
ill_photo.set_xlabel('time')
ill_photo.set_ylabel('illumination')

#图像展示
temp_photo.grid()
hum_photo.grid()
ill_photo.grid()

```

```

dataPlot = FigureCanvasTkAgg(f, master=root)
dataPlot.show()
dataPlot.get_tk_widget().pack()
#root.after(500,draw,root)

'''*****'''
*函数: windows
*主要功能: 主窗口
*****'''
def windows():
    root=Tk()
    root.title('windows')
    #root.geometry("400x300")

    #设置状态更新
    flag=StringVar()
    sleep_flag=StringVar()
    set_temp=StringVar()
    set_hum=StringVar()
    mo1_st=StringVar()
    mo2_st=StringVar()
    temp=StringVar()
    hum=StringVar()
    ill=StringVar()

    Label(root, text='物联网感知课程设计',font=('宋体', 20)).pack()

    fm_flag_set=Frame(root)

    flaglabel_name=Label(fm_flag_set,text='总开关',font=('宋体', 15))
    flaglabel_name.pack(side=LEFT,pady=10)
    flaglabel_val=Label(fm_flag_set,textvariable=flag,font=('宋体', 15))
    flaglabel_val.pack(side=LEFT,padx=30)

    sleep_flaglabel_name=Label(fm_flag_set,text='睡眠模式开关',font=('宋
体', 15))
    sleep_flaglabel_name.pack(side=LEFT,pady=10)

    sleep_flaglabel_val=Label(fm_flag_set,textvariable=sleep_flag,font=('宋
体', 15))
    sleep_flaglabel_val.pack(side=LEFT,padx=30)

    set_templabel_name=Label(fm_flag_set,text='设定温度',font=('宋体',
15))

```

```

set_templabel_name.pack(side=LEFT,pady=10)
set_templabel_val=Label(fm_flag_set,textvariable=set_temp,font=('宋体', 15))
set_templabel_val.pack(side=LEFT,padx=30)

set_humlabel_name=Label(fm_flag_set,text='设定湿度',font=('宋体', 15))
set_humlabel_name.pack(side=LEFT,pady=10)
set_humlabel_val=Label(fm_flag_set,textvariable=set_hum,font=('宋体', 15))
set_humlabel_val.pack(side=LEFT,padx=30)

mo2_stlabel_name=Label(fm_flag_set,text='空调状态',font=('宋体', 15))
mo2_stlabel_name.pack(side=LEFT,pady=10)
mo2_stlabel_val=Label(fm_flag_set,textvariable=mo2_st,font=('宋体', 15))
mo2_stlabel_val.pack(side=LEFT,padx=30)

mo1_stlabel_name=Label(fm_flag_set,text='加湿器状态',font=('宋体', 15))
mo1_stlabel_name.pack(side=LEFT,pady=10)
mo1_stlabel_val=Label(fm_flag_set,textvariable=mo1_st,font=('宋体', 15))
mo1_stlabel_val.pack(side=LEFT,padx=30)
fm_flag_set.pack(side=TOP,pady=10)

#温度-湿度-光照强度模块实时展示
fm1=Frame(root)
templabel_name=Label(fm1,text='温度',font=('宋体', 15))
templabel_name.pack(side=LEFT,pady=10)
templabel_val=Label(fm1,textvariable=temp,font=('宋体', 15))
templabel_val.pack(side=LEFT,padx=30)

humlabel_name=Label(fm1,text='湿度度',font=('宋体', 15))
humlabel_name.pack(side=LEFT,pady=10)
humlabel_val=Label(fm1,textvariable=hum,font=('宋体', 15))
humlabel_val.pack(side=LEFT,padx=30)

illlabel_name=Label(fm1,text='光照强度',font=('宋体', 15))
illlabel_name.pack(side=LEFT,pady=10)
illlabel_val=Label(fm1,textvariable=ill,font=('宋体', 15))
illlabel_val.pack(side=LEFT,padx=30)
fm1.pack(side=TOP,pady=15)

```

```
#刷新各传感器状态和图像

flush(flag,sleep_flag,set_temp,set_hum,temp,hum,ill,mo1_st,mo2_st,root)
draw(root)
#root.update_idletasks()
root.mainloop()
```

## 四. 系统说明

本系统搭建了一个智能家居环境,系统由两个触摸传感器,一个温湿度传感器,一个光敏传感器,两个电机传感器,一个LED调光传感器来模拟智能家居设备。

触摸传感器一个为智能家居的总开关,一个作为睡眠模式的开关。当关闭智能总开关时,智能操作全部关闭,当睡眠模式开启时,自动通过电机来关闭窗帘,并关闭所有灯光。

温湿度传感器采集到的数据通过智能处理,自动智能地开启空调和加湿器

光敏传感器采集到的数据通过智能处理,自动进行室内的调光

## 五. 设计总结

通过对传感器的数据读取,从端口读取数据到数据库,再从数据库中读取数据,并进行相应的界面设计,学到了很多知识,也提高了自己解决问题的能力,收获颇多。

## 六. 附录

源程序文件名清单:

Main.py 主程序入口

Mysqlcon.py 数据库连接

Coordinator.py 协调器程序

Tempandhum.py 温湿度传感器程序

Photoresistor.py 光敏传感器程序

Touch.py 触摸传感器程序

PWM.py 数字调光程序

Motor.py 电机程序

Display.py 本地界面程序

部分Web端页面程序:

```
<!DOCTYPE html>
<html lang="en">
<head>
```

```

<meta charset="utf-8">
<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1.0"
/>
<meta name="description" content="Xenon Bootstrap Admin Panel" />
<meta name="author" content="" />

<title>物联网感知课程设计</title>

</head>

        toastr.info("", "欢迎来到物联网感知课程设计 web 界面", opts);

        }, 3000);

        <div class="row">
            <div class="col-sm-3">

                <div class="xe-widget xe-counter" data-count=".num"
data-from="15" data-to="19.13" data-suffix="°C" data-duration="2">

                    <div class="xe-icon">
                        <i class="linecons-cloud"></i>
                    </div>
                    <div class="xe-label">
                        <span>温度</span>

                        <strong class="num">0.0°C</strong>
                    </div>
                </div>

                <div class="xe-widget xe-counter xe-counter-purple"
data-count=".num" data-from="42" data-to="43.45" data-suffix="%"
data-duration="3" data-easing="false">
                    <div class="xe-icon">
                        <i class="linecons-user"></i>
                    </div>
                    <div class="xe-label">
                        <span>湿度</span>

                        <strong class="num">43.45</strong>
                    </div>
                </div>
            </div>

```



```

        <div class="xe-widget xe-counter xe-counter-info"
data-count=".num" data-from="300" data-to="365" data-suffix="x1"
data-duration="4" data-easing="true">
            <div class="xe-icon">
                <i class="linecons-camera"></i>
            </div>
            <div class="xe-label">
                <span>光照强度</span>
                <strong class="num">356</strong>
            </div>
        </div>

</div>
<div class="col-sm-6">

        <div class="chart-item-bg">
            <div class="chart-label">
                <span class="text-medium text-muted">当前
温度</span>

                <div class="h3 text-secondary text-bold"
data-count="this" data-from="0.00" data-to="19.13" data-suffix="°C"
data-duration="1">0.00%</div>

            </div>
            <div id="pageviews-visitors-chart" style="height:
298px;"></div>
        </div>

</div>
<div class="col-sm-3">

        <div class="chart-item-bg">
            <div class="chart-label chart-label-small">
                <span>空调状态</span>
                <h3 ><font color="#FF0000">OFF</font> </h3>
            </div>
            <div id="server-uptime-chart" style="height:
134px;"></div>
        </div>

        <div class="chart-item-bg">
            <div class="chart-label chart-label-small">
                <span>加湿器状态</span>

```

```

        <h3><font color="#FF0000">ON</font></h3>
    </div>
    <div id="server-uptime-chart" style="height:
134px;"></div>

    </div>

    </div>
</div>

    <div class="col-sm-6">

        <div class="chart-item-bg">
            <div class="chart-label">
                <span class="text-medium text-muted">当前
湿度</span>

                <div class="h3 text-secondary text-bold"
data-count="this" data-from="0.00" data-to="43.25" data-suffix="%"
data-duration="1">0.00%</div>

            </div>
            <div id="pageviews-visitors-chart2" style="height:
298px;"></div>
        </div>

    </div>
    <div class="col-sm-6">

        <div class="chart-item-bg">
            <div class="chart-label">
                <span class="text-medium text-muted">
当前光照强度</span>

                <div class="h3 text-secondary text-bold"
data-count="this" data-from="0.00" data-to="365" data-suffix='x1'
data-duration="1">0.00%</div>

            </div>
            <div id="pageviews-visitors-chart3"
style="height: 298px;"></div>
        </div>

    </div>

    <!-- Main Footer -->

```

```

        <!-- Choose between footer styles: "footer-type-1" or
"footer-type-2" -->
        <!-- Add class "sticky" to always stick the footer to the end
of page (if page contents is small) -->
        <!-- Or class "fixed" to always fix the footer to the end of
page -->

        <footer class="main-footer sticky footer-type-1">

            <div class="footer-inner">

                <!-- Add your copyright text here -->
                <div class="footer-text">
                    &copy; 2018-物联网感知课程设计 by

                        <a href="www.lgzxhy.top" target="_blank"
class='mysite' ><strong>liugang</strong> </a>
                </div>

                <!-- Go to Top Link, just add rel="go-top" to any link
to add this functionality -->
                <div class="go-up">

                    <a href="#" rel="go-top">
                        <i class="fa-angle-up"></i>
                    </a>

                </div>

            </div>

        </footer>
    </div>
</div>

<div class="page-loading-overlay">
    <div class="loader-2"></div>
</div>
<!-- Bottom Scripts -->
<script src="assets/js/bootstrap.min.js"></script>
<script src="assets/js/TweenMax.min.js"></script>
<script src="assets/js/resizeable.js"></script>
<script src="assets/js/joinable.js"></script>

```

```

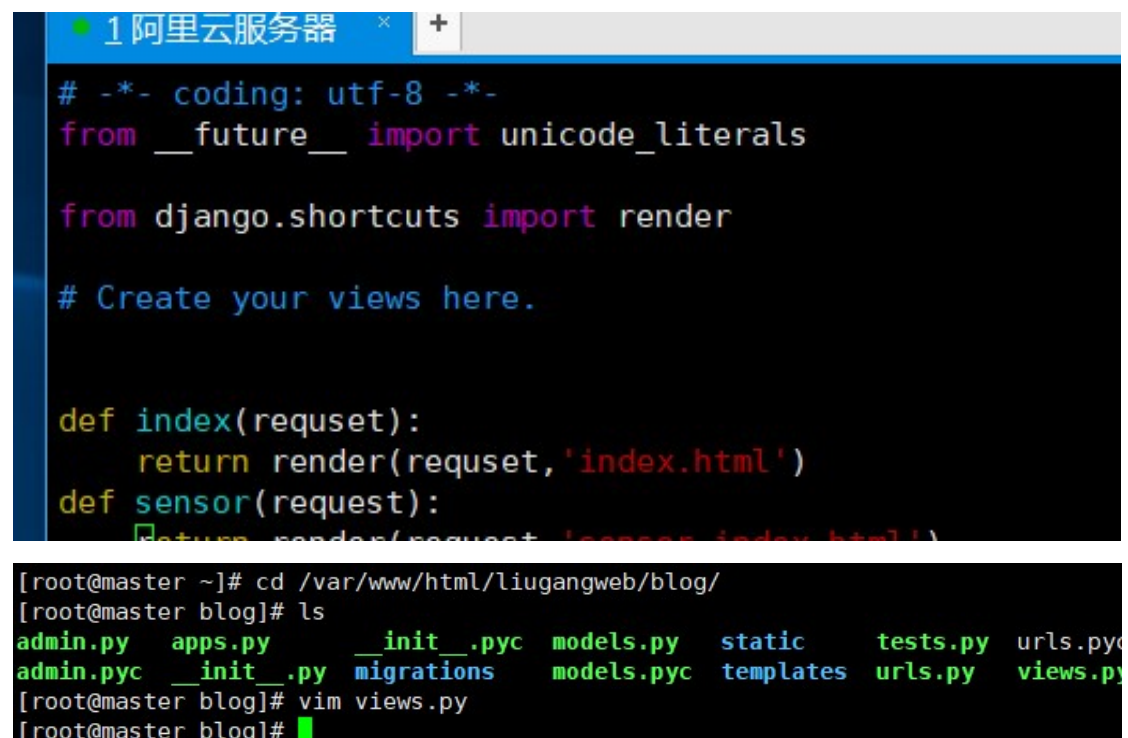
<script src="assets/js/xenon-api.js"></script>
<script src="assets/js/xenon-toggles.js"></script>

<!-- Imported scripts on this page -->
<script src="assets/js/xenon-widgets.js"></script>
<script
src="assets/js/devexpress-web-14.1/js/globalize.min.js"></script>
<script
src="assets/js/devexpress-web-14.1/js/dx.chartjs.js"></script>
<script src="assets/js/toastr/toastr.min.js"></script>
<!-- JavaScripts initializations and stuff -->
<script src="assets/js/xenon-custom.js"></script>

</body>
</html>

```

云端服务器代码



The screenshot shows a terminal window with a tab labeled '1 阿里云服务器'. The terminal displays Django code for a view named 'index' and 'sensor'. Below the code, shell commands are executed to navigate to the project directory, list files, and open the 'views.py' file in vim.

```

# -*- coding: utf-8 -*-
from __future__ import unicode_literals

from django.shortcuts import render

# Create your views here.

def index(request):
    return render(request, 'index.html')
def sensor(request):
    return render(request, 'sensor.html')

```

```

[root@master ~]# cd /var/www/html/liugangweb/blog/
[root@master blog]# ls
admin.py  apps.py  __init__.pyc  models.py  static  tests.py  urls.py
admin.pyc  __init__.py  migrations  models.pyc  templates  urls.py  views.py
[root@master blog]# vim views.py
[root@master blog]#

```