江苏大学

JIANGSU UNIVERSITY

计算机网络实验报告

实验名称：一个典型物联网系统中传输机制的设计与实现

学院名称：计算机科学与通信工程学院

专业班级：物联网工程 2303

学生姓名：邱佳亮

学生学号：3230611072

教师姓名：李峰

报告日期：2024/11/26

目录

1 作业要求

一般一个典型的物联网系统包括感控层（传感器），网络层和应用层组成，而网络层主要用于实现感控对象与应用层的服务对象之间的通信。本次作业就以 TCP/IP 协议栈中传输层协议的应用开发为目标，以 UDP 方式实现一种感控对象与服务对象之间的通信机制，其体系结构如图 1 所示。其中感控对象为一个虚拟路灯对象，在实现过程中用随机数模拟其温度、湿度和环境照度等感知数据，灯作为被控对象，可以通过服务器对其进行打开、关闭控制，且用不同颜色表示其开关状态。每个虚拟路灯都将有一个标识，以示区别。而服务对象可以同时与若干个虚拟路灯对象通信，每个虚拟路灯会定期向服务对象发送其当前状态，服务对象可以对任一个虚拟路灯进行开关控制。

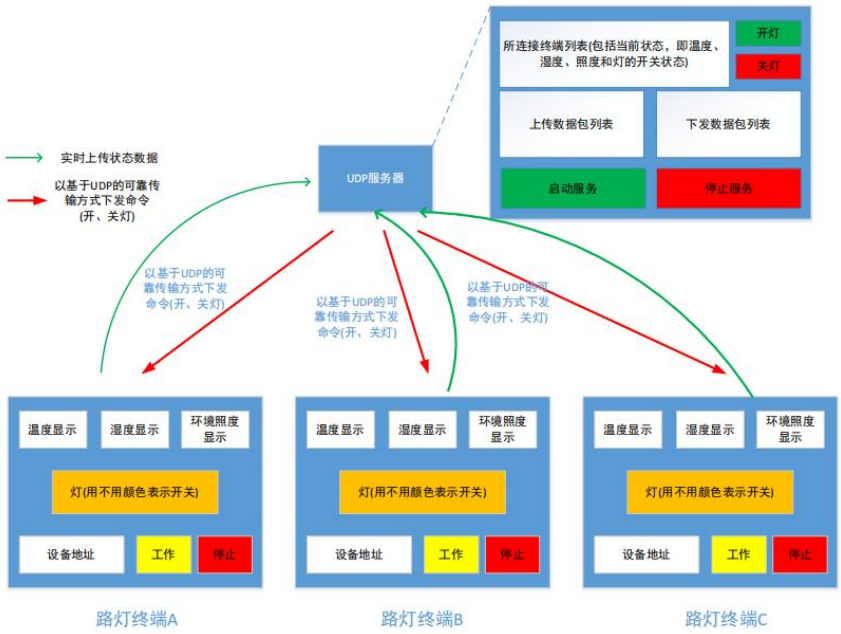

图 1　作业要求

要求：

（1）虚拟路灯状态上传数据直接基于 UDP 实现传输，不考虑可靠性，而服务对象发送给各虚拟路灯的开关命令要利用握手机制实现可靠传输。

（2）基于 UDP 自定义上传、下发数据包格式和传输方式。并实现通信协议的定义、封装和解析。

（3）虚拟路灯界面主要显示当前状态，包括当前温度、湿度和照度，这些都

可以用随机数产生，以及灯的开关状态；每个虚拟路灯皆有一个唯一设备 ID，并可以在界面上显示。

（4）服务器上可以保存各虚拟路灯的历史状态数据，并且可以显示当前各虚拟路灯的相应状态，包括灯的开关状态、温度、湿度和照度。各灯的历史数据可以按指定时段查询。

（5）不限定编程语言。

（6）完成后提交电子版报告（PDF 格式），包括：项目名称、项目目标、设计与实现、测试结果、总结与展望、源代码等内容。同时可以录制 2 分钟左右的演示视频上传课程网站。

2 项目名称

基于 PyQt6 和 socket 的物联网传输机制设计

3 项目实现

3.1 GUI 设计

3.1.1 客户端 UI 设计

利用 Qt-Designer 设计路灯终端，主界面信息包含路灯状态、环境状态（包括温度、湿度、亮度）、设备 ID、设备状态与服务端的开关按钮：

图 2 UI 样式

保存 ui 文件并利用 Qt 编译为 python 文件。

3.1.2 服务端 UI 设计

利用 Qt-Designer 设计服务器端，主界面包含三台设备的 ID 和获取的环境信息、设备开关状态的信息、当前服务器状态和服务器开关机按钮：

图 3 UI 样式

保存 ui 文件并利用 Qt 编译为 python 文件。

3.2 客户端功能实现

3.2.1 生成环境信息

```
1. def randomEnv():
2.     tep=str(np.random.randint(-20,41))
3.     light=str(np.random.randint(300,501))
4.     wet=str(np.random.randint(0,101))
5.     return tep,light,wet
```

通过 randomEnv 函数随机生成环境参数温度（tep）、光照（light）、湿度（wet）并返回三个值。

3.2.2 客户端通信类

```
1. class UdpClient(QtCore.QThread):
2.     message_res=QtCore.pyqtSignal(str)
3.     message_data=QtCore.pyqtSignal(tuple)
4.     socket=None
5.     def __init__(self):
```

```
6.          super().__init__()
7.          self.socket=socket.socket(socket.AF_INET,socket.SOCK_DGRAM)
8.      def __del__(self):
9.          self.wait()
10.     def run(self):
11.        while True:
12.         self.LocalHost="127.0.0.1"
13.         self.id="1"
14.         data=list(randomEnv())
15.         tep=data[0]
16.         light=data[1]
17.         wet=data[2]
18.         self.socket.sendto(tep.encode('utf-8'),(self.LocalHost,8080))
19.         self.socket.sendto(light.encode('utf-8'),(self.LocalHost,808
0))
20.         self.socket.sendto(wet.encode('utf-8'),(self.LocalHost,8080))
21.         self.socket.sendto(self.id.encode('utf-8'),(self.LocalHost,80
80))
22.         self.status,_=self.socket.recvfrom(1024)
23.         self.data_id,_=self.socket.recvfrom(1024)
24.         data.append(self.data_id.decode('utf-8'))
25.         data=tuple(data)
26.         self.message_data.emit(data)
27.         if self.status==b"1":
28.             self.message_res.emit("open")
29.         else:
30.             self.message_res.emit("close")
31.         time.sleep(1000)
32.         #print(tep,light,wet)
```

UdpClient 类为 UDP 客户端，继承自 QtCore.QThread，用于发送和接收 UDP 数据包。其中定义了两个发射信号 message-res 和 message-data，使用 socket(socket.AF_INET,socket.SOCK_DGRAM)创建了基于数据包的 UDP 套接字。在线程的运行函数中，调用 randomEnv 函数生成环境数据 data，将环境数据、设备 id 发送到了 Localhost 的 8080 端口。客户端通过 recvfrom 接收服务器返回状态 status 和 data-id，分别控制路灯的亮灭和 id 显示，如果 status 为 1，发射 open 信号，如果 status 为 0，发射 close 信号。

3.2.3 服务器开关机方法

```
1. def start(self):
```

```
    if (self.label_f_isON.text()=="开启"):
        return
    else:
        self.label_f_isON.setText("开启")
        self.socketThread=UdpClient()
        self.socketThread.message_data.connect(self.display)
        self.socketThread.message_res.connect(self.update_status)
        self.socketThread.start()
```

开机方法 connect 到 UI 界面的开启按钮，按下开启按钮后，检查服务端是否开启，若已经开机，则不响应，若未开机，修改 UI 界面的开启 label，创建新的 UdpClient 对象，将接收到的 data 信号 connect 到 display 函数，res 信号 connect 到 update 函数，启动线程开始通信。

```
def end(self):
    if (self.label_f_isON.text()=="关闭"):
        return
    else:
        if self.socketThread is not None:
            self.socketThread=None
            self.label_f_isON.setText("关闭")
            self.update_status('close')
            self.label.setText("")
            self.label_2.setText("")
            self.label_3.setText("")
```

关机方法 connect 到 UI 界面的关闭按钮，按下按钮后，检查服务端是否关闭，若已关闭，则不响应，若未关闭，将线程关闭，设置状态 label 为关闭，调用 update 函数并传入参数 close，将环境参数 label 均置为空。

3.2.4 UI 显示方法

```
def display(self,data):
    self.label.setText(data[0]+"摄氏度")
    self.label_2.setText(data[1]+"勒克斯")
    self.label_3.setText(data[2]+"%")
    self.label_id.setText(data[3])
def update_status(self, status):
    self.is_ON.setText(status)
    if status=='open':
       self.is_ON.setStyleSheet("color:red")
    else:
       self.is_ON.setStyleSheet("color: black")
```

Display 函数接收 data 信号并根据信号将环境参数和 id 展示在 UI 界面上。Update 函数接收 status 信号，将路灯状态显示在 UI 界面上，如果状态为 open，文字颜色变为红色。

3.3 服务器端方法实现

3.3.1 服务器通信类

```
1. class Thread(QtCore.QThread):
2.     message=QtCore.pyqtSignal(int,str,str,tuple,str)
3.     ID=[]
4.     server_socket=None
5.     def __init__(self):
6.       super().__init__()
7.       print("启动")
8.       self.server_socket=socket.socket(socket.AF_INET,socket.SOCK_DGRAM)
9.       self.server_socket.bind(('',8080))
10.      print("启动成功")
11.    def run(self):
12.       while True:
13.           rece_tep,addr=self.server_socket.recvfrom(1024)
14.           rece_light,addr=self.server_socket.recvfrom(1024)
15.           rece_wet,addr=self.server_socket.recvfrom(1024)
16.           id,addr=self.server_socket.recvfrom(1024)
17.           data_tep=rece_tep.decode('utf-8')
18.           data_light=rece_light.decode('utf-8')
19.           data_wet=rece_wet.decode('utf-8')
20.           data_id=id.decode('utf-8')
21.           print(f"ip 地址为{addr}:温度{data_tep}/光照{data_light}/湿度{data_wet}/ID{data_id}")
22.         if data_light=='#' or data_tep=='#' or data_wet=='#':
23.             break
24.         if data_light<="400":
25.             self.server_socket.sendto(b"1",addr)
26.             self.server_socket.sendto(data_id.encode('utf-8'),addr)
27.             self.bind(data_id,addr,(data_tep,data_wet,data_light),'open')
28.         else:
29.             self.server_socket.sendto(b"0",addr)
30.             self.server_socket.sendto(data_id.encode('utf-8'),addr)
```

```
31.             self.bind(data_id,addr,(data_tep,data_wet,data_light),'c
lose')
32.   def bind(self,id,addr,data,status):
33.       if id in self.ID:
34.           self.message.emit(self.ID.index(id),addr[0],id,data,status)
35.       else:
36.           if len(self.ID)<3:
37.               self.ID.append(id)
38.               self.message.emit(self.ID.index(id),addr[0],id,data,stat
us)
39.           else:
40.               self.message.emit(-1,(),"")
```

Thread 类继承自 QtThread，用于接收客户端发送的数据，并根据数据内容发送响应。其中定义了发射信号 message，使用 socket(socket.AF_INET,socket.SOCK_DGRAM)创建了基于数据包的 UDP 套接字。该类接收温度、亮度、湿度和 ID 数据，并根据光照数据是否大于 400，决定路灯是否开启。Bind 函数用于处理接收数据，如果服务器未接收过该 ID 客户端数据，将环境数据打包为 message 信号并发送。

3.3.2 服务器开机和关机函数

```
1. def start(self):
2.       if self.label_isON.text()=="关机":
3.           self.label_isON.setText("开机")
4.           self.thread=Thread()
5.           self.thread.start()
6.           self.thread.message.connect(self.display)
7.       elif self.label_isON.text()=="开机":
8.           return
9.   def end(self):
10.       if self.label_isON.text()=="开机":
11.           #self.thread.terminate()
12.           self.thread.wait()
13.           self.thread=None
14.           self.label_isON.setText("关机")
15.           self.textBrowser_f1.setText("")
16.           self.textBrowser_f2.setText("")
17.           self.textBrowser_f3.setText("")
18.           self.textBrowser_f4.setText("")
19.           self.label_f1.setText("设备 ID:")
20.           self.label_f2.setText("设备 ID:")
```

```
            self.label_f3.setText("设备 ID:")
            self.ID=[]
        elif self.label_isON.text()=="关机":
            return
```

Start 方法用于启动 Thread 线程，如果当前为关机状态，将 label 更改为开机，创建线程并启动，将信号 message connect 到 display 函数。End 方法用于结束线程，如果服务器为开机状态，将 label 改变为关机，清空文字浏览器内容和 ID 列表。

3.3.3 UI 显示方法

```
def display(self,event,id,data,status):
    if id and id not in self.ID and status=='open':
        self.ID.append(id)
    if event==0:
        self.label_f1.setText("设备 1  "+str(id))
        self.label_res1.setText("id:"+str(id)+"  "+"状态:"+status)
        self.textBrowser_f1.setText("温度:"+str(data[0])+"摄氏度"+'\n'
                                    +"湿度:"+str(data[1])+"%"+'\n'
                                    +"亮度:"+str(data[2]+"勒克斯"))
    elif event==1:
        self.label_f2.setText("设备 2  "+str(id))
        self.label_res2.setText("id:"+str(id)+"  "+"状态:"+status)
        self.textBrowser_f2.setText("温度:"+str(data[0])+"摄氏度"+'\n'
                                    +"湿度:"+str(data[1])+"%"+'\n'
                                    +"亮度:"+str(data[2]+"勒克斯"))
    elif event==2:
        self.label_f3.setText("设备 3  "+str(id))
        self.label_res3.setText("id:"+str(id)+"  "+"状态:"+status)
        self.textBrowser_f3.setText("温度:"+str(data[0])+"摄氏度"+'\n'
                                    +"湿度:"+str(data[1])+"%"+'\n'
                                    +"亮度:"+str(data[2]+"勒克斯"))
```

Display 方法接收 message 信号，从中提取出客户端的环境参数、ID 和状态信息并更新 UI。

4 测试结果

运行服务器和三个设备：

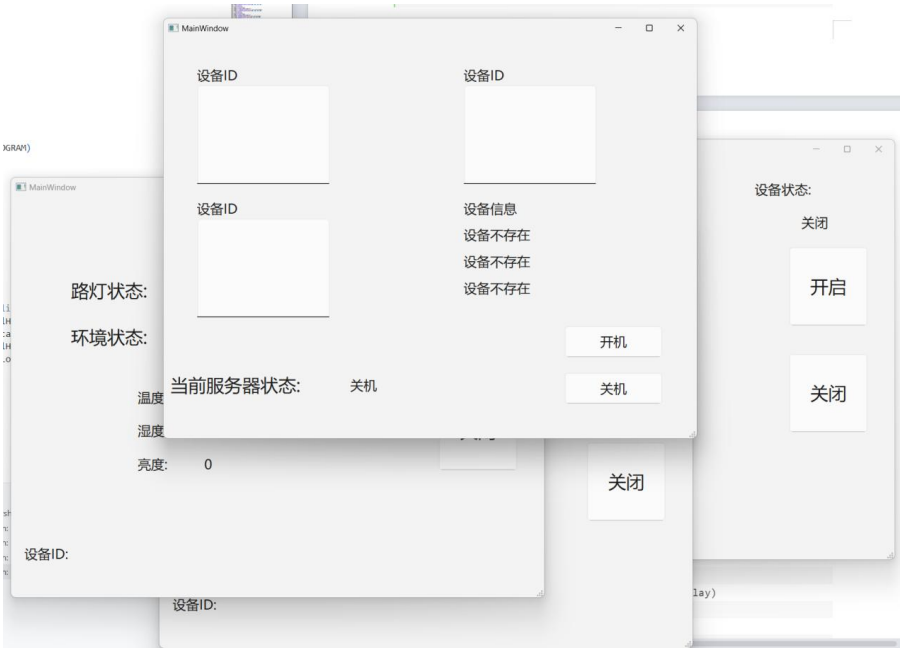

图 4　运行程序

开启服务器和设备 1，看到设备 1 中出现了随机生成的环境信息，由于亮度小于 400，服务器指令路灯打开：

图 5　路灯状态

服务器界面上出现了设备 1 的环境信息，设备信息中显示设备 1 为开启状态：

图 6 服务器界面

控制台也输出了客户端向服务器传输的数据：

```
ip地址为('127.0.0.1', 59736):温度16/光照318/湿度43/ID1
```

图 7 数据

打开其他设备，服务器端展示出所有设备的环境信息：

图 8 打开所有设备

5 总结和展望

项目成功实现了基于 PyQt6 和 socket 的物联网传输机制，包括客户端和服

务端的 GUI 设计，以及客户端功能实现和服务器端方法实现。客户端和服务端的 UI 设计利用 Qt-Designer 完成，并编译为 Python 文件。客户端通过随机数模拟环境参数，并周期性地向服务端发送状态信息。服务端能够接收来自多个虚拟路灯的状态信息，并根据接收到的数据控制路灯的开关状态。测试结果显示，服务器和客户端均能正常运行，客户端能够生成环境信息并发送给服务端。服务端能够正确解析客户端发送的数据，并根据亮度信息控制路灯的开关状态。

系统目前支持三个虚拟路灯的通信，未来可以扩展支持更多设备，同时优化服务器端的数据处理和存储能力。未来可以进一步优化用户界面，使其更加直观和用户友好。未来可以考虑引入加密通信和认证机制，以保护数据传输的安全。

6 源代码

6.1.1 main.py

```
import socket
from PyQt6 import QtCore, QtGui, QtWidgets
from PyQt6.QtCore import *
class Thread(QtCore.QThread):
    message=QtCore.pyqtSignal(int,str,str,tuple,str)
    ID=[]
    server_socket=None
    def __init__(self):
        super().__init__()
         print("启动")
         self.server_socket=socket.socket(socket.AF_INET,socket.SOCK_
DGRAM)
         self.server_socket.bind(('',8080))
         print("启动成功")
     def run(self):
         while True:
             rece_tep,addr=self.server_socket.recvfrom(1024)
             rece_light,addr=self.server_socket.recvfrom(1024)
             rece_wet,addr=self.server_socket.recvfrom(1024)
             id,addr=self.server_socket.recvfrom(1024)
             data_tep=rece_tep.decode('utf-8')
             data_light=rece_light.decode('utf-8')
             data_wet=rece_wet.decode('utf-8')
             data_id=id.decode('utf-8')
```

```
24.             print(f"ip 地址为{addr}:温度{data_tep}/光照{data_light}/湿
度{data_wet}/ID{data_id}")
25.             if data_light=='#' or data_tep=='#' or data_wet=='#':
26.                 break
27.             if data_light<="400":
28.                 self.server_socket.sendto(b"1",addr)
29.                 self.server_socket.sendto(data_id.encode('utf-8'),ad
dr)
30.                 self.bind(data_id,addr,(data_tep,data_wet,data_ligh
t),'open')
31.             else:
32.                 self.server_socket.sendto(b"0",addr)
33.                 self.server_socket.sendto(data_id.encode('utf-8'),ad
dr)
34.                 self.bind(data_id,addr,(data_tep,data_wet,data_ligh
t),'close')
35.     def bind(self,id,addr,data,status):
36.         if id in self.ID:
37.             self.message.emit(self.ID.index(id),addr[0],id,data,stat
us)
38.         else:
39.             if len(self.ID)<3:
40.                 self.ID.append(id)
41.                 self.message.emit(self.ID.index(id),addr[0],id,data,
status)
42.             else:
43.                 self.message.emit(-1,(),"")
44. class Ui_Server(object):
45.     ID=[]
46.     STATUS=[]
47.     def setupUi(self, MainWindow):
48.         MainWindow.setObjectName("MainWindow")
49.         MainWindow.resize(800, 600)
50.         self.centralwidget = QtWidgets.QWidget(parent=MainWindow)
51.         self.centralwidget.setObjectName("centralwidget")
52.         self.label_f1 = QtWidgets.QLabel(parent=self.centralwidget)
53.         self.label_f1.setGeometry(QtCore.QRect(50, 30, 201, 51))
54.         font = QtGui.QFont()
55.         font.setPointSize(15)
56.         self.label_f1.setFont(font)
57.         self.label_f1.setObjectName("label_f1")
58.         self.label_f3 = QtWidgets.QLabel(parent=self.centralwidget)
```

```
59.         self.label_f3.setGeometry(QtCore.QRect(50, 230, 201, 51))
60.         font = QtGui.QFont()
61.         font.setPointSize(15)
62.         self.label_f3.setFont(font)
63.         self.label_f3.setObjectName("label_f3")
64.         self.label_f2 = QtWidgets.QLabel(parent=self.centralwidget)
65.         self.label_f2.setGeometry(QtCore.QRect(450, 30, 201, 51))
66.         font = QtGui.QFont()
67.         font.setPointSize(15)
68.         self.label_f2.setFont(font)
69.         self.label_f2.setObjectName("label_f2")
70.         self.label_total = QtWidgets.QLabel(parent=self.centralwidge
t)
71.         self.label_total.setGeometry(QtCore.QRect(450, 230, 201, 5
1))
72.         font = QtGui.QFont()
73.         font.setPointSize(15)
74.         self.label_total.setFont(font)
75.         self.label_total.setObjectName("label_total")
76.         self.label_statu = QtWidgets.QLabel(parent=self.centralwidge
t)
77.         self.label_statu.setGeometry(QtCore.QRect(10, 500, 250, 40))
78.         font = QtGui.QFont()
79.         font.setPointSize(20)
80.         self.label_statu.setFont(font)
81.         self.label_statu.setObjectName("label_statu")
82.         self.label_isON = QtWidgets.QLabel(parent=self.centralwidge
t)
83.         self.label_isON.setGeometry(QtCore.QRect(280, 500, 50, 40))
84.         font = QtGui.QFont()
85.         font.setPointSize(15)
86.         self.label_isON.setFont(font)
87.         self.label_isON.setObjectName("label_isON")
88.         self.pushButton_ON = QtWidgets.QPushButton(parent=self.centr
alwidget)
89.         self.pushButton_ON.setGeometry(QtCore.QRect(600, 430, 150, 5
0))
90.         font = QtGui.QFont()
91.         font.setPointSize(15)
92.         self.pushButton_ON.setFont(font)
93.         self.pushButton_ON.setObjectName("pushButton_ON")
```

```
94.         self.pushButton_OUT = QtWidgets.QPushButton(parent=self.cent
ralwidget)
95.         self.pushButton_OUT.setGeometry(QtCore.QRect(600, 500, 15
0, 50))
96.         font = QtGui.QFont()
97.         font.setPointSize(15)
98.         self.pushButton_OUT.setFont(font)
99.         self.pushButton_OUT.setObjectName("pushButton_OUT")
100.         self.textBrowser_f1 = QtWidgets.QTextBrowser(parent=self.ce
ntralwidget)
101.         self.textBrowser_f1.setGeometry(QtCore.QRect(50, 70, 200, 1
50))
102.         font = QtGui.QFont()
103.         font.setPointSize(15)
104.         self.textBrowser_f1.setFont(font)
105.         self.textBrowser_f1.setObjectName("textBrowser_f1")
106.         self.textBrowser_f3 = QtWidgets.QTextBrowser(parent=self.ce
ntralwidget)
107.         self.textBrowser_f3.setGeometry(QtCore.QRect(50, 270, 20
0, 150))
108.         font = QtGui.QFont()
109.         font.setPointSize(15)
110.         self.textBrowser_f3.setFont(font)
111.         self.textBrowser_f3.setObjectName("textBrowser_f3")
112.         self.textBrowser_f2 = QtWidgets.QTextBrowser(parent=self.ce
ntralwidget)
113.         self.textBrowser_f2.setGeometry(QtCore.QRect(450, 70, 20
0, 150))
114.         font = QtGui.QFont()
115.         font.setPointSize(15)
116.         self.textBrowser_f2.setFont(font)
117.         self.textBrowser_f2.setObjectName("textBrowser_f2")
118.         self.label_res1 = QtWidgets.QLabel(parent=self.centralwidge
t)
119.         self.label_res1.setGeometry(QtCore.QRect(450, 280, 200, 3
0))
120.         font = QtGui.QFont()
121.         font.setPointSize(15)
122.         self.label_res1.setFont(font)
123.         self.label_res1.setObjectName("label_res1")
124.         self.label_res2 = QtWidgets.QLabel(parent=self.centralwidge
t)
```

```
125.         self.label_res2.setGeometry(QtCore.QRect(450, 320, 200, 3
0))
126.         font = QtGui.QFont()
127.         font.setPointSize(15)
128.         self.label_res2.setFont(font)
129.         self.label_res2.setObjectName("label_res2")
130.         self.label_res3 = QtWidgets.QLabel(parent=self.centralwidge
t)
131.         self.label_res3.setGeometry(QtCore.QRect(450, 360, 200, 3
0))
132.         font = QtGui.QFont()
133.         font.setPointSize(15)
134.         self.label_res3.setFont(font)
135.         self.label_res3.setObjectName("label_res3")
136.         MainWindow.setCentralWidget(self.centralwidget)
137.         self.menubar = QtWidgets.QMenuBar(parent=MainWindow)
138.         self.menubar.setGeometry(QtCore.QRect(0, 0, 800, 26))
139.         self.menubar.setObjectName("menubar")
140.         MainWindow.setMenuBar(self.menubar)
141.         self.statusbar = QtWidgets.QStatusBar(parent=MainWindow)
142.         self.statusbar.setObjectName("statusbar")
143.         MainWindow.setStatusBar(self.statusbar)
144.
145.         self.retranslateUi(MainWindow)
146.         QtCore.QMetaObject.connectSlotsByName(MainWindow)
147.
148.     def retranslateUi(self, MainWindow):
149.         _translate = QtCore.QCoreApplication.translate
150.         MainWindow.setWindowTitle(_translate("MainWindow", "MainWin
dow"))
151.         self.label_f1.setText(_translate("MainWindow", "设备 ID"))
152.         self.label_f3.setText(_translate("MainWindow", "设备 ID"))
153.         self.label_f2.setText(_translate("MainWindow", "设备 ID"))
154.         self.label_total.setText(_translate("MainWindow", "设备信息
"))
155.         self.label_statu.setText(_translate("MainWindow", "当前服务
器状态:"))
156.         self.label_isON.setText(_translate("MainWindow", "关机"))
157.         self.pushButton_ON.setText(_translate("MainWindow", "开机
"))
158.         self.pushButton_OUT.setText(_translate("MainWindow", "关机
"))
```

```
159.         self.label_res1.setText(_translate("MainWindow", "设备不存在
"))
160.         self.label_res2.setText(_translate("MainWindow", "设备不存在
"))
161.         self.label_res3.setText(_translate("MainWindow", "设备不存在
"))
162.         self.pushButton_ON.clicked.connect(self.start)
163.         self.pushButton_OUT.clicked.connect(self.end)
164.     def start(self):
165.         if self.label_isON.text()=="关机":
166.             self.label_isON.setText("开机")
167.             self.thread=Thread()
168.             self.thread.start()
169.             self.thread.message.connect(self.display)
170.         elif self.label_isON.text()=="开机":
171.             return
172.     def end(self):
173.         if self.label_isON.text()=="开机":
174.             if self.socketThread is not None:
175.                 self.thread=None
176.                 self.label_isON.setText("关机")
177.                 self.textBrowser_f1.setText("")
178.                 self.textBrowser_f2.setText("")
179.                 self.textBrowser_f3.setText("")
180.                 self.textBrowser_f4.setText("")
181.                 self.label_f1.setText("设备 ID:")
182.                 self.label_f2.setText("设备 ID:")
183.                 self.label_f3.setText("设备 ID:")
184.                 self.ID=[]
185.         elif self.label_isON.text()=="关机":
186.             return
187.     def display(self,event,ip,id,data,status):
188.         if id and id not in self.ID and status=='open':
189.             self.ID.append(id)
190.         if event==0:
191.                 self.label_f1.setText("设备 1  "+str(id))
192.                 self.label_res1.setText("id:"+str(id)+"  "+"状态:"+
status)
193.                 self.textBrowser_f1.setText("温度:"+str(data[0])+"
摄氏度"+'\n'
194.                                             +"湿度:"+str(data[1])+
"%"+'\n'
```

```
                                                    +"亮度:"+str(data[2]+"
勒克斯"))
        elif event==1:
                self.label_f2.setText("设备 2  "+str(id))
                self.label_res2.setText("id:"+str(id)+"  "+"状态:"+
status)
                self.textBrowser_f2.setText("温度:"+str(data[0])+"
摄氏度"+'\n'
                                                    +"湿度:"+str(data[1])+
"%"+'\n'
                                                    +"亮度:"+str(data[2]+"
勒克斯"))
        elif event==2:
                self.label_f3.setText("设备 3  "+str(id))
                self.label_res3.setText("id:"+str(id)+"  "+"状态:"+
status)
                self.textBrowser_f3.setText("温度:"+str(data[0])+"
摄氏度"+'\n'
                                                    +"湿度:"+str(data[1])+
"%"+'\n'
                                                    +"亮度:"+str(data[2]+"
勒克斯"))
if __name__=="__main__":
    import sys
    app=QtWidgets.QApplication(sys.argv)
    Server=QtWidgets.QMainWindow()
    ui=Ui_Server()
    ui.setupUi(Server)
    Server.show()
    sys.exit(app.exec())
```

6.2 light.py

```
from PyQt6 import QtCore, QtGui, QtWidgets
import socket
import numpy as np
import time
def randomEnv():
    #seed=np.random.randint(1,101)
    #np.random.seed(seed)
    tep=str(np.random.randint(-20,41))
    light=str(np.random.randint(300,501))
     wet=str(np.random.randint(0,101))
     return tep,light,wet
```

```
class UdpClient(QtCore.QThread):
    message_res=QtCore.pyqtSignal(str)
    message_data=QtCore.pyqtSignal(tuple)
    socket=None
    def __init__(self):
        super().__init__()
        self.socket=socket.socket(socket.AF_INET,socket.SOCK_DGRAM)
    def __del__(self):
        self.wait()
    def run(self):
        while True:
            self.LocalHost="127.0.0.1"
            self.id="1"
            data=list(randomEnv())
            tep=data[0]
            light=data[1]
            wet=data[2]
            #self.message.emit(f"Temperature: {tep}, Light: {ligh
t}, Wet: {wet}")
            self.socket.sendto(tep.encode('utf-8'),(self.LocalHost,8
080))
            self.socket.sendto(light.encode('utf-8'),(self.LocalHost,
8080))
            self.socket.sendto(wet.encode('utf-8'),(self.LocalHost,8
080))
            self.socket.sendto(self.id.encode('utf-8'),(self.LocalHo
st,8080))
            self.status,_=self.socket.recvfrom(1024)
            self.data_id,_=self.socket.recvfrom(1024)
            data.append(self.data_id.decode('utf-8'))
            data=tuple(data)
            self.message_data.emit(data)
            if self.status==b"1":
                self.message_res.emit("open")
            else:
                self.message_res.emit("close")
            time.sleep(1000)
            #print(tep,light,wet)
class Ui_MainWindow(object):
    def setupUi(self, MainWindow):
        MainWindow.setObjectName("MainWindow")
        MainWindow.resize(800, 600)
```

```
49.         self.centralwidget = QtWidgets.QWidget(parent=MainWindow)
50.         self.centralwidget.setObjectName("centralwidget")
51.         self.Ludeng_label = QtWidgets.QLabel(parent=self.centralwidg
et)
52.         self.Ludeng_label.setGeometry(QtCore.QRect(90, 110, 170, 6
0))
53.         font = QtGui.QFont()
54.         font.setPointSize(20)
55.         self.Ludeng_label.setFont(font)
56.         self.Ludeng_label.setObjectName("Ludeng_label")
57.         self.Huanjin_label_2 = QtWidgets.QLabel(parent=self.centralw
idget)
58.         self.Huanjin_label_2.setGeometry(QtCore.QRect(90, 180, 17
0, 60))
59.         font = QtGui.QFont()
60.         font.setPointSize(20)
61.         self.Huanjin_label_2.setFont(font)
62.         self.Huanjin_label_2.setObjectName("Huanjin_label_2")
63.         self.Tem_label_3 = QtWidgets.QLabel(parent=self.centralwidge
t)
64.         self.Tem_label_3.setGeometry(QtCore.QRect(190, 280, 120, 4
0))
65.         font = QtGui.QFont()
66.         font.setPointSize(15)
67.         self.Tem_label_3.setFont(font)
68.         self.Tem_label_3.setObjectName("Tem_label_3")
69.         self.Wet_label_4 = QtWidgets.QLabel(parent=self.centralwidge
t)
70.         self.Wet_label_4.setGeometry(QtCore.QRect(190, 330, 120, 4
0))
71.         font = QtGui.QFont()
72.         font.setPointSize(15)
73.         self.Wet_label_4.setFont(font)
74.         self.Wet_label_4.setObjectName("Wet_label_4")
75.         self.Light_label_5 = QtWidgets.QLabel(parent=self.centralwid
get)
76.         self.Light_label_5.setGeometry(QtCore.QRect(190, 380, 120, 4
0))
77.         font = QtGui.QFont()
78.         font.setPointSize(15)
79.         self.Light_label_5.setFont(font)
80.         self.Light_label_5.setObjectName("Light_label_5")
```

```
81.         self.pushButton_ON = QtWidgets.QPushButton(parent=self.centr
alwidget)
82.         self.pushButton_ON.setGeometry(QtCore.QRect(640, 130, 121, 1
21))
83.         font = QtGui.QFont()
84.         font.setPointSize(20)
85.         self.pushButton_ON.setFont(font)
86.         self.pushButton_ON.setObjectName("pushButton_ON")
87.         self.label = QtWidgets.QLabel(parent=self.centralwidget)
88.         self.label.setGeometry(QtCore.QRect(290, 280, 120, 40))
89.         font = QtGui.QFont()
90.         font.setPointSize(15)
91.         self.label.setFont(font)
92.         self.label.setObjectName("label")
93.         self.label_2 = QtWidgets.QLabel(parent=self.centralwidget)
94.         self.label_2.setGeometry(QtCore.QRect(290, 330, 120, 40))
95.         font = QtGui.QFont()
96.         font.setPointSize(15)
97.         self.label_2.setFont(font)
98.         self.label_2.setObjectName("label_2")
99.         self.label_3 = QtWidgets.QLabel(parent=self.centralwidget)
100.         self.label_3.setGeometry(QtCore.QRect(290, 380, 120, 40))
101.         font = QtGui.QFont()
102.         font.setPointSize(15)
103.         self.label_3.setFont(font)
104.         self.label_3.setObjectName("label_3")
105.         self.is_ON = QtWidgets.QLabel(parent=self.centralwidget)
106.         self.is_ON.setGeometry(QtCore.QRect(280, 110, 170, 60))
107.         font = QtGui.QFont()
108.         font.setPointSize(20)
109.         self.is_ON.setFont(font)
110.         self.is_ON.setObjectName("is_ON")
111.         self.pushButton_OUT = QtWidgets.QPushButton(parent=self.cen
tralwidget)
112.         self.pushButton_OUT.setGeometry(QtCore.QRect(640, 290, 12
1, 121))
113.         font = QtGui.QFont()
114.         font.setPointSize(20)
115.         self.pushButton_OUT.setFont(font)
116.         self.pushButton_OUT.setObjectName("pushButton_OUT")
117.         self.label_f_isON = QtWidgets.QLabel(parent=self.centralwid
get)
```

```
118.         self.label_f_isON.setGeometry(QtCore.QRect(660, 80, 60, 3
0))
119.         font = QtGui.QFont()
120.         font.setPointSize(15)
121.         self.label_f_isON.setFont(font)
122.         self.label_f_isON.setObjectName("label_f_isON")
123.         self.label_f = QtWidgets.QLabel(parent=self.centralwidget)
124.         self.label_f.setGeometry(QtCore.QRect(590, 30, 110, 30))
125.         font = QtGui.QFont()
126.         font.setPointSize(15)
127.         font.setUnderline(False)
128.         self.label_f.setFont(font)
129.         self.label_f.setObjectName("label_f")
130.         self.label_ipad = QtWidgets.QLabel(parent=self.centralwidge
t)
131.         self.label_ipad.setGeometry(QtCore.QRect(20, 520, 100, 30))
132.         font = QtGui.QFont()
133.         font.setPointSize(15)
134.         self.label_ipad.setFont(font)
135.         self.label_ipad.setObjectName("label_ipad")
136.         self.label_ip = QtWidgets.QLabel(parent=self.centralwidget)
137.         self.label_ip.setGeometry(QtCore.QRect(120, 520, 100, 30))
138.         font = QtGui.QFont()
139.         font.setPointSize(15)
140.         self.label_ip.setFont(font)
141.         self.label_ip.setText("")
142.         self.label_ip.setObjectName("label_ip")
143.         MainWindow.setCentralWidget(self.centralwidget)
144.         self.menubar = QtWidgets.QMenuBar(parent=MainWindow)
145.         self.menubar.setGeometry(QtCore.QRect(0, 0, 800, 26))
146.         self.menubar.setObjectName("menubar")
147.         MainWindow.setMenuBar(self.menubar)
148.         self.statusbar = QtWidgets.QStatusBar(parent=MainWindow)
149.         self.statusbar.setObjectName("statusbar")
150.         MainWindow.setStatusBar(self.statusbar)
151.         self.retranslateUi(MainWindow)
152.         QtCore.QMetaObject.connectSlotsByName(MainWindow)
153.     def retranslateUi(self, MainWindow):
154.         _translate = QtCore.QCoreApplication.translate
155.         MainWindow.setWindowTitle(_translate("MainWindow", "MainWin
dow"))
```

```
156.         self.Ludeng_label.setText(_translate("MainWindow", "路灯状
态:"))
157.         self.Huanjin_label_2.setText(_translate("MainWindow", "环境
状态:"))
158.         self.Tem_label_3.setText(_translate("MainWindow", "温度:"))
159.         self.Wet_label_4.setText(_translate("MainWindow", "湿度:"))
160.         self.Light_label_5.setText(_translate("MainWindow", "亮度:
"))
161.         self.pushButton_ON.setText(_translate("MainWindow", "开启
"))
162.         self.label.setText(_translate("MainWindow", "0"))
163.         self.label_2.setText(_translate("MainWindow", "0"))
164.         self.label_3.setText(_translate("MainWindow", "0"))
165.         self.is_ON.setText(_translate("MainWindow", "关闭"))
166.         self.pushButton_OUT.setText(_translate("MainWindow", "关闭
"))
167.         self.label_f_isON.setText(_translate("MainWindow", "关闭"))
168.         self.label_f.setText(_translate("MainWindow", "设备状态:"))
169.         self.label_ipad.setText(_translate("MainWindow", "设备 ID:
"))
170.         self.socketThread=None
171.         self.pushButton_ON.clicked.connect(self.start)
172.         self.pushButton_OUT.clicked.connect(self.end)
173.     def end(self):
174.         if (self.label_f_isON.text()=="关闭"):
175.             return
176.         else:
177.             if self.socketThread is not None:
178.                 self.socketThread=None
179.                 self.label_f_isON.setText("关闭")
180.                 self.update_status('close')
181.                 self.label.setText("")
182.                 self.label_2.setText("")
183.                 self.label_3.setText("")
184.     def start(self):
185.         if (self.label_f_isON.text()=="开启"):
186.             return
187.         else:
188.             self.label_f_isON.setText("开启")
189.             self.socketThread=UdpClient()
190.             self.socketThread.message_data.connect(self.display)
```

```
191.             self.socketThread.message_res.connect(self.update_statu
s)
192.             self.socketThread.start()
193.     def display(self,data):
194.         self.label.setText(data[0]+"摄氏度")
195.         self.label_2.setText(data[1]+"勒克斯")
196.         self.label_3.setText(data[2]+"%")
197.         self.label_ip.setText(data[3])
198.     def update_status(self,status):
199.         self.is_ON.setText(status)
200.         if status=='open':
201.             self.is_ON.setStyleSheet("color:red")
202.         else:
203.             self.is_ON.setStyleSheet("color:black")
204. if __name__=="__main__":
205.     import sys
206.     app=QtWidgets.QApplication(sys.argv)
207.     MainWindow=QtWidgets.QMainWindow()
208.     ui=Ui_MainWindow()
209.     ui.setupUi(MainWindow)
210.     MainWindow.show()
211.     sys.exit(app.exec())
212.     app=QtWidgets.QApplication(sys.argv)
213.     Server=QtWidgets.QMainWindow()
214.     ui=Ui_Server()
215.     ui.setupUi(Server)
216.     Server.show()
217.     sys.exit(app.exec())
```