WIRTSCHAFTS
UNIVERSITÄT
WIEN VIENNA
UNIVERSITY OF
ECONOMICS
AND BUSINESS

EFMD
EQUIS
ACCREDITED

Seminar Paper

Controlling sockets based on humidity

Georg Prohaska

Student ID: 0325904

Course: Information Systems Engineering Course III

Lecturer: Univ.Prof. Dr. Gustaf Neumann

Department of Information Systems and Operations, Vienna University of Economics and Business, Welthandelsplatz 1, 1020 Vienna, Austria

DEPARTMENT FÜR INFORMATIONS-VERARBEITUNG UND PROZESS-MANAGEMENT DEPARTMENT OF INFORMATION SYSTEMS AND OPERATIONS

1 Introduction

Humidifiers are a common tool to control air humidity and they can be found in many regular households. A large variety of devices exist and many of them, especially the cheaper ones, provide very limited functionality, i.e. turning it on, off and regulating intensity. Naturally, we control such a device based on the current humidity in the room, i.e. switching it on when the current level is low and switching it off when a comfortable level is reached. Clearly, this could be done automatically based on a configurable threshold, a functionality that is offered by more expensive devices.
In this project we solved this problem using a few simple and cheap IoT components including an Arduino UNO, a humidity sensor and a remote-control socket. Furthermore, we developed a web-interface that displays chronological humidity and temperature data and allows configuring the threshold. In that way, the humidifier can even be controlled from anywhere in the world over the internet.

2 Goals and Achievements

The primary goal of this project was to create a device that remote controls sockets based on the current humidity in the room. A key aspect was to keep the needed components cheap while providing the same functionality as existing pricy all-in-one solutions. As an additional usability feature we decided to implement a web-interface.
Overall, all of the mentioned goals have been achieved. The main components of the project are a standard 433Mhz remote control socket, which is connected to a humidifier, a DHT-22 humidity/temperature sensor, a 433Mhz sender unit and an Arduino UNO that controls these parts. A sketch for the Arduino has been developed that provides an interface for the serial port, i.e. signals for turning the socket on and off can be sent to Arduino from any software application on the machine connected to it via USB. Furthermore, current humidity and temperature data is provided periodically also via the serial port. A python script handles communication with this interface and enforces the humidity threshold set by the user. A NodeJS server provides the front-end to display data and forwards user input to the python script.

3 Hardware

As a humidity sensor we used the DHT-22 / AM2302 sensor. It is available for around 7 € and works with Arduino as well as most common boards. It works with 3.3 as well as 5 Volts. This is a very common sensor with a simple wiring. Figure 1 shows the setup used in this project.

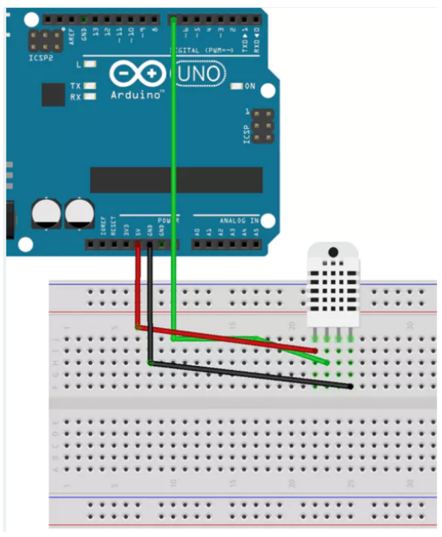


Figure 1: Wiring for the DHT-22 sensor.

We used remote control sockets of the brand Brennenstuhl (Figure 2). Three sockets plus a remote-control cost around 30 €. The signal codes of the remote control, called tristate codes, are fixed; Hence, once the code for a socket is known it can always be used to control it.

Figure 2: Remote-control sockets used in this project.

For communication with the remote-control socket we used a 433Mhz sender and a 433Mhz receiver module. Together they are around 4 €. The signal strength for the sender is somewhat weak. To improve that antennas may be added to increase the range of the transmitted signal. The wiring for these components is also rather simple (Figure 3). The receiver was only used to discover the signal code of the remote control belonging to the socket. Hence, for the final setup the module on the left was removed.

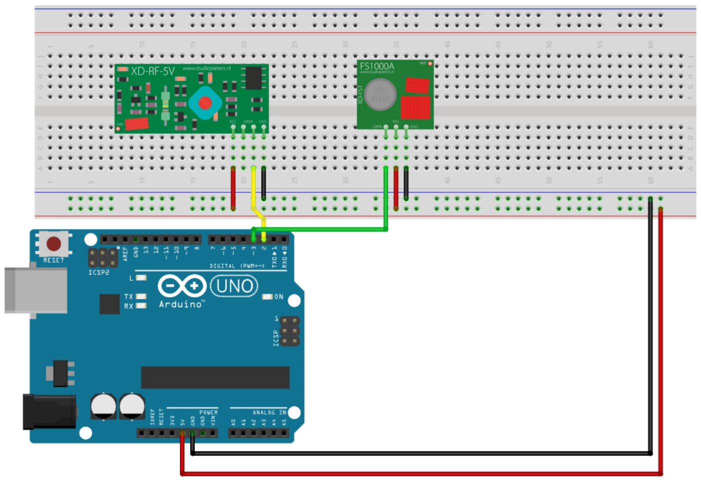


Figure 3: Wiring for 433Mhz sender and receiver.

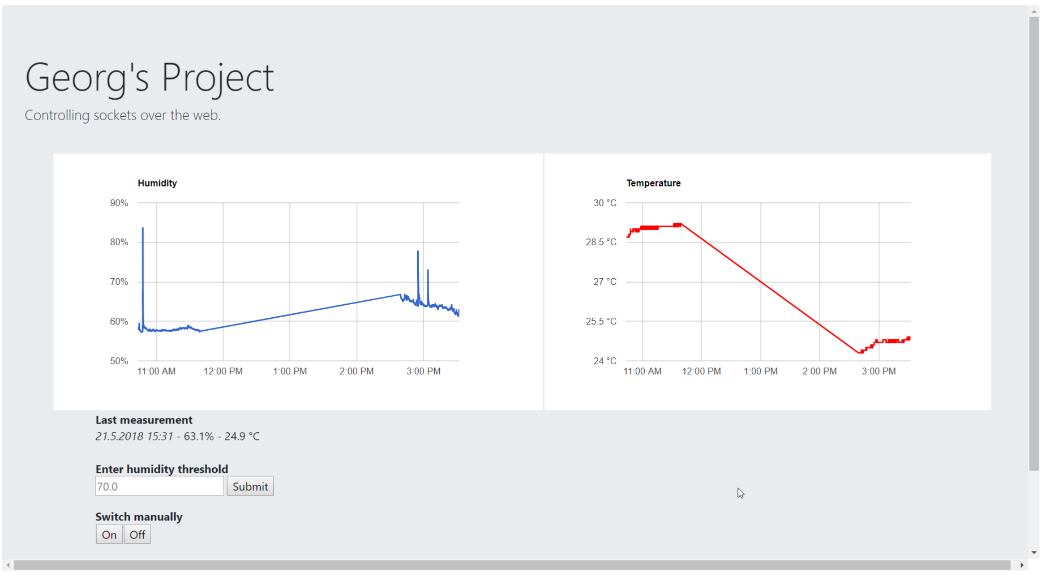
For the humidifier that is plugged into the socket the only crucial point is that it is controlled mechanically, i.e. it can be turned on and off solely by connecting it to an electric circuit. This is the case for many of the cheaper models.
Initially, an Intel Galileo was provided as a central control unit, however, we ran into problems with the 433Mhz sender/receiver components. The commonly used library for using these parts is not compatible with the Intel Galileo, which is why we decided to switch to the Arduino UNO, which costs around 20 €. The processing power of the Arudino UNO is more than enough for the requirements of this project.

4 Software

For the DHT-22 sensor we used the standard library from Adafruit[1]. Reading the current humidity and temperature values are two simple function calls, after initialising the sensor in the setup phase. Every few seconds, after a

[1]`https://github.com/adafruit/DHT-sensor-library`

simple sanity check, the current values are written to the serial port
Another common library was used to control the 433Mhz sender/receiver modules[2]. To discover the tristate codes of the remote control an example sketch of the library was used[3]. These codes were noted down and hardcoded in the sketch of the project. There, the program listens for a serial event and when a predefined byte is received sends an on/off signal to the socket.
A python script communicates with the Arduino via the serial port. It gathers data and saves it to a csv. Furthermore, it manages the threshold, i.e. it sends the appropriate on/off signals once the humidity threshold is reached. This script provides an interface via stdin to receive commands itself for setting the threshold and turning the socket on and off regardless of the threshold.
The last piece of the puzzle is the front-end. We used a Node JS server that provides a web interface for display of the data and user input (Figure 4). The temperature and humidity data of the last 24 hours is displayed in two separate graphs. Furthermore, the user can set the threshold here and directly turn the socket on and off.


Georg's Project

Controlling sockets over the web.

Last measurement
21.5.2018 15:31 - 63.1% - 24.9 °C

Enter humidity threshold
70.0

Figure 4: Remote-control sockets used in this project.

[2]https://github.com/sui77/rc-switch
[3]https://github.com/sui77/rc-switch/tree/master/examples/ReceiveDemo_Advanced

5 Conclusion

This project presents a relatively simple way to control a humidifier based on the current humidity and even interact with it over the internet. This concept can of course be applied to different appliances as well, e.g. switching lights on and off based on a timer. A major drawback of our solution is of course that a computer has the be running to control the Arduino respectively the socket. The threshold could have been managed in the sketch itself rendering the Arduino able to work independently, however, since we wanted to provide the web interface anyway we decided to keep that logic on the server side. Another point that has not been addressed is security. In the current state, anyone that has access to the web interface can control the socket. For future work some form of cryptographic measures should be implemented to prohibit unwanted access.
Overall, we showed that with a few cheap components advanced and useful functionality can be achieved. This is just a glimpse of what is possible in the world of IoT.