Smart Light

Fredrik Haupt

Kevin Sanaee

Md Reaz Morshed

Department of Computer
and Systems Sciences

Degree project 4 HE credits

Computer and Systems Sciences

Degree project at the bachelor level

Autumn term 2021

Supervisor: Ramin Firouzi

Stockholm
University

Table of Contents

Abstract

In this paper we describe our project to develop an application for an Android smart phone to turn a lamp on and off through voice commands. The project is part of a basic course for the Internet of Things (IoT) and the groundwork for the application is made from laboratory work done during the course. During the project we faced several issues that hampered our ability to make progress and fulfill our vision for the project. Because of the many issues we have also dedicated part of this report to the process of solving those issues. To emulate the phone we used a virtual machine in Android Studio which connected to a Raspberry Pi. The Raspberry Pi sent instructions to an “application programming interface” (API) operated by Telldus and the API sent these instructions to a device called a Tellstick that is connected to an actuator with a lamp. The use cases for this are many since the actuator can be connected to many different things. But for this specific project we consider the increasing popularity of smart homes as a convenience and the use of smart functions to make everyday life easier for differently abled as the main reasons for doing this project. This project is of course also an important learning experience for us as students to get into the IoT world.

Keywords: Internet of Things, Raspberry Pi, Android Studio, speech recognition

Introduction

The Internet of Things has been possible to realize during the last few years because of the wide availability of an internet connection in large parts of the developed world. We can connect computers, other devices and different things wirelessly in small and big networks for them to communicate with each other. This communication is then used by us to make smart electronics like phones, TVs, ovens, cars and much more. Even voice assistants like Amazon Alexa and Apple’s Siri are possible because of the Internet of Things.

Smart devices have many different functions but what they all have in common is the connection to other devices that can control then from afar. In a smart phone there are many different utilities and smart phones today are very capable computation machines. Some smart devices are not as advanced but can controlled from different computers, such as smart phones. It is therefore convenient to tie smart systems to smart phone applications. Most people have started carrying their smart phone with them all the time which makes it suitable for controlling smart functions over the internet. Maybe you want to open your smart garage door before even seeing it or find out if a home security system is giving of a false alarm, this is possible as long as you have a connection to the internet on your smart phone.

Background

Smart homes are becoming more and more part of everyday life. The convenience of being able to control your home appliances over the internet is open to everyone with a smart phone and using alternative methods for things such as just turning on or off a light can help people who are differently abled. Our Smart Light project is an example of just that. Anyone with an Android phone and internet connection can use our app to control a lamp without actually being near the lamp. For most people this might not seem like much more than a convenience, but for someone with a disability this could make their everyday life easier. We chose this project to explore what it means to work with this quickly increasing and evolving area in information technology.

Method

In the beginning of the project we had a planned to do more than just the turning a lamp on or off. We were supposed to have that as the first step for creating an app with more features such as dimming the light, controlling more than one lamp and having only recognized users' voices be able to control the lights. But issues with the hardware meant that we could not achieve these goals. We had also planned for working in sprints but with the issues making us fall behind schedule the sprints where not possible to follow. The result was that we just followed a road map for what was most important at the time.

The majority of the project time was spent on making our Raspberry Pi work. We used our knowledge gained from laboratory work preceding the project, asked help from our supervisor and searched the internet to troubleshoot the issues. Eventually with help from our supervisor we figured out that our Raspberry Pi had a file, called "dhcpcd.conf", that was empty. We were then given a new Raspberry Pi that worked. By then there was not enough time for us to expand our project beyond the most basic function since we also had to write this report before the deadline.

At first the issues we faced seemed random. The connection between our devices worked fine one day and the next day it did not. We tried on the school network which worked during the laboratory work, but that only worked sometimes. One time it even stopped working at school after changing room but still being on the same network. Then we tried to set the devices up at home which did not work at first. The most reoccurring problem seemed to be that the Raspberry Pi's IP address, which is required to connect the Raspberry Pi with the computer that we worked from, kept changing and we could not get the correct new address.

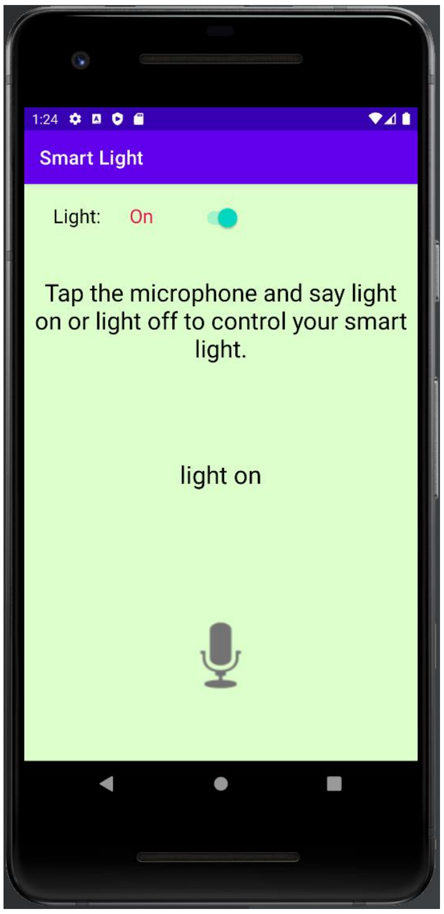
We found several different suggestions to remedy a Raspberry Pi not connecting to other devices. Many were found online and through our supervisor. We also tried a lot ourselves from previous computer network experience and guesswork. Our supervisor helped us solve the issue with finding the correct IP address on the Raspberry Pi. This required the Raspberry Pi to be connected to a monitor and have a mouse and keyboard. If the connection had worked as intended, we could have used a remote terminal emulator to control the Raspberry Pi. This worked for a while but eventually we could not even connect the Raspberry Pi to the internet at all. During a meeting with our supervisor was when we found out that the earlier mentioned file "dhcpcd.conf" was empty.

"dhcpcd" is an abbreviation for "DHCP client daemon" (Arch Linux 2021). The DHCP (Dynamic Host Configuration Protocol) is vital for assigning IP address automatically and is therefore vital when connecting to the internet. It is not strange that we experienced so many connectivity issues when such an important file was out of order. Why dhcpcd.conf was empty we do not know, but we certainly know to look for it if similar problems arise in the future. We were then given another Raspberry Pi by our supervisor, which worked as intended.

The short amount of time that was still left on the project was spent researching and implementing our voice recognition solution. We did so by reading articles and watching video tutorials online, and eventually we settled on a method from a video on Youtube by Sandip Bhattacharya. This included the speech to text method also, which meant that the most basic idea of our project was finished.

Design and development

Smart Light is a fairly simple application and the user interface follows suit. At this stage Smart Light is just a prototype and a final product would be different. In the top left corner a button for turning the light on and off is left as a control during prototype testing but would most likely be removed completely or moved to another menu. Under that there is a text explaining the how to use the app. If the application had more features, these instructions would probably also be moved. In the middle of the screen the recognized speech is posted for the user to know what was picked up by the speech recognizer. Furthest down on the screen a button in the form a microphone enables speech input. If no speech input is available on the device a message will be displayed for a few seconds saying “Speech-to-text not available on this device.”.



Tap the microphone and say light on or light off to control your smart light.

light on

User interface after “light on” has been said.

Hardware

Here we will describe what tools we used and how we used them. We will start with the hardware.

Raspberry Pi

The Raspberry Pi is a small computer that runs the “Raspberry Pi OS”, this operative system is a Linux/GNU distribution. We used the Raspberry Pi to host scripts for communicating with the API that let us control the lamp. It is used to find information about devices connected to a network and run scripts to control these devices.

TELLSTICK ZNET LITE v2

A TELLSTICK ZNET LITE v2, or simply Tellstick, is a gateway that connects to a service called “Telldus Live!” which makes it possible to control different smart devices online. For this project we connected one actuator to switch our lamp on and off. The Tellstick connects to the internet and talks to a Telldus server that hosts an API to control their products, such as the actuator we used.

Software

Python scripts

On the Raspberry Pi we wrote Python scripts, which also were previously used in the laboratory work. These scripts use the API hosted on the Telldus servers to control the actuator. One script is also used to find which devices are available on the same network as the Raspberry Pi. If we had not experienced problems with the connection to the Raspberry Pi, we would have written the scripts by connecting from a Windows computer through the Secure Shell protocol with Putty. Putty is a terminal emulator from which you can control Unix and Windows computers remotely.

Android Studio

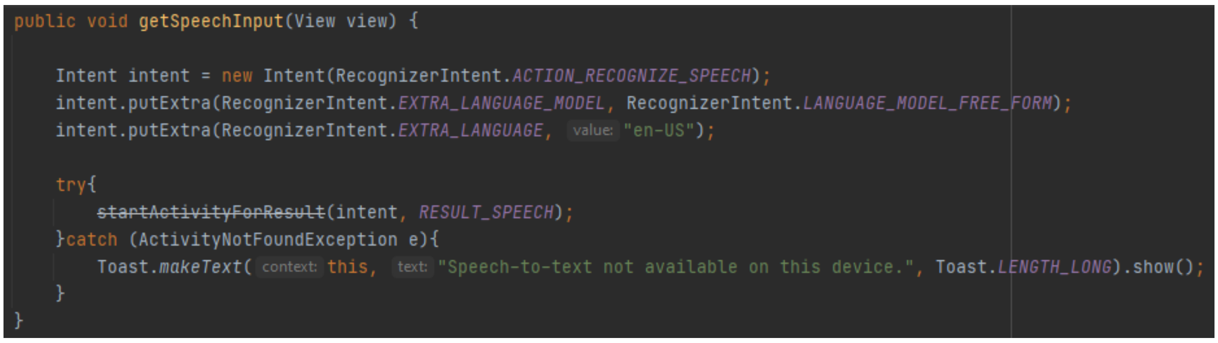
Android Studio is an integrated development environment that is used for easy Android development. Through Android Studio a user can write Java and XML code to run on a virtual Android machine. The virtual machine was the only way we tested the application. It is possible that the application may not work on a real Android phone without some tweaking.

Java and XML

We based our initial Java and XML code on the instructions from the laboratory work done during the course (Firouzi, Ul Alam 2021). This part of the project code handles the starting state when opening the application and connecting to the Raspberry Pi from the machine running the application. It is also the basis for the graphical user interface with much of the backend being the same as in the instructions.

The methods we used to enable speech are based on Sandip Bhattacharya’s video “How to Build a Speech to Text App in Android Studio” (2020). The methods are called “getSpeechInput” and “onActivityResult”.

getSpeechInput is used to process speech input from the user when the microphone button in the application is tapped. In its current form the method can only recognize American English and only using Android Studio on the Windows OS, tried on Windows 10. After speech is recognized the result is saved.



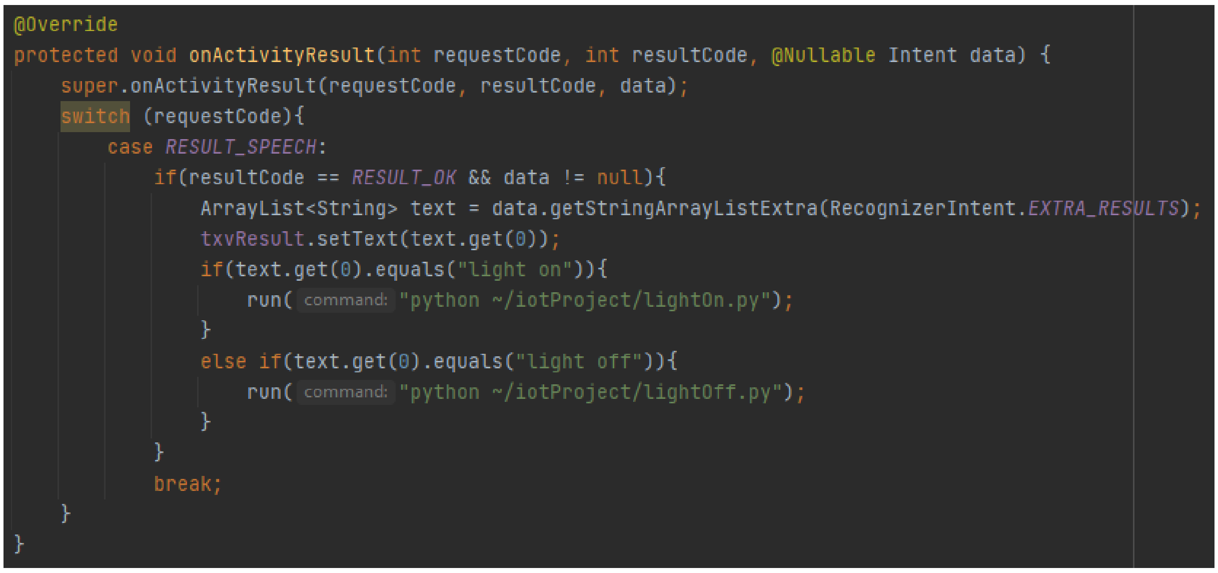
```
public void getSpeechInput(View view) {

    Intent intent = new Intent(RecognizerIntent.ACTION_RECOGNIZE_SPEECH);
    intent.putExtra(RecognizerIntent.EXTRA_LANGUAGE_MODEL, RecognizerIntent.LANGUAGE_MODEL_FREE_FORM);
    intent.putExtra(RecognizerIntent.EXTRA_LANGUAGE,  value: "en-US");

    try{
        ~~startActivityForResult~~(intent, RESULT_SPEECH);
    }catch (ActivityNotFoundException e){
        Toast.makeText( context: this,  text: "Speech-to-text not available on this device.", Toast.LENGTH_LONG).show();
    }
}
```

getSpeechInput

onActivityResult takes the result from getSpeechInput and checks if it matches either “light on” and then runs the script on the Raspberry Pi to turn the light on or “light off” to run the script for turning the light off.



```java
@Override
protected void onActivityResult(int requestCode, int resultCode, @Nullable Intent data) {
    super.onActivityResult(requestCode, resultCode, data);
    switch (requestCode){
        case RESULT_SPEECH:
            if(resultCode == RESULT_OK && data != null){
                ArrayList<String> text = data.getStringArrayListExtra(RecognizerIntent.EXTRA_RESULTS);
                txvResult.setText(text.get(0));
                if(text.get(0).equals("light on")){
                    run( command: "python ~/iotProject/lightOn.py");
                }
                else if(text.get(0).equals("light off")){
                    run( command: "python ~/iotProject/lightOff.py");
                }
            }
            break;
    }
}
```

onActivityResult

Discussion

The complications related to the hardware used during the project made most of our planning moot. It did however show us the kind of issues that may arise during IoT development. We would of course liked to have made a more feature complete application but what we learned from troubleshooting many different issues also provide valuable skills. Development on projects that are made by professionals also run into to problems such as the ones we faced. In those cases it will most likely also result in delays. The nature of school projects mean that deadlines are strict and if we did not have other courses coming up right after this one we could have continued working on the application until it was more in line with our initial vision, just like how projects in the professional world are delayed all the time. We would have liked to discover more about the ease of use for people with disabilities like we had in mind at first, but it is reasonable to say that our solution would be suitable for some disabilities. Having to tap the button in the app is of course not as versatile as using something that is always on like the Alexa smart home system. A good always on system with voice controls is more convenient for both the regular user and disabled people.

IoT development requires more than just writing code. It is important to understand how hardware interacts with other hardware, and how hardware interacts with software, and how software interacts with other software. Without this understanding we cannot create anything for IoT. So, even though we did not achieve all our goals for this project we did learn a lot about IoT development.

We focus on the application of IoT for home use with a lamp in our project. We started out with a vision for our application that would have more features, but many issues with the hardware meant that we had to narrow this vision to just the most basic function of turning the lamp on and off. Though troubleshooting hardware was great learning experience the project as a whole suffered. Expanding our knowledge of integrating software and hardware solutions for a complete project would have been a more rewarding experience and resulted in a more interesting project, but for this report we will have to make do with the small amount of progress that was made. This project is just one small part in the Internet of Things but can still be important for understanding the process of working with IoT.

References

Tsiatsis, V. Höller, J. Mulligan, C. Karnouskos, S. Boyle, D. (2019) *Internet of Things: Technologies and Applications for a New Age of Intelligence (2nd ed).* Elsevier Ltd. Available at: https://www-sciencedirect-com.ezp.sub.su.se/book/9780128144350/internet-of-things#book-description

Sandip Bhattacharya (2020) *How to Build a Speech to Text App in Android Studio*. 7 September. Available at: https://www.youtube.com/watch?v=Mavo98h5wT0 (Accessed: 2 January 2022).

Ramin Firouzi, Mahbub Ul Alam (2021) *Laboratory work 2: Accessing IoT things via an Android application (Version: 1.4)*. Available at: https://ilearn.dsv.su.se/pluginfile.php/273859/mod_resource/content/1/IoTLab2_Manual.pdf

Arch Linux (2021) *dhcpcd*. Available at: https://wiki.archlinux.org/title/dhcpcd (Accessed: 7 January 2022)

Microsoft (2021) *Dynamic Host Configuration Protocol (DHCP)*. Available at: https://docs.microsoft.com/en-us/windows-server/networking/technologies/dhcp/dhcp-top (Accessed: 7 January 2022)