UDC 621.339:316.13 **doi: 10.26906/SUNZ.2023.2.059**

O. Skakalina, A. Kapiton

National University “Yuri Kondratyuk Poltava Polytechnic”, Poltava, Ukraine

IMPLEMENTATION OF THE LIGHTING FUNCTION IN THE SMART HOME CONCEPT

Abstract. The principle of "System of intelligent management of the building" provides a completely new approach in the organization of life support of the building, in which the efficiency of operation and reliability of management of all systems and executive devices of the building increases significantly due to the combination of hardware and software. A smart building should be understood as a system that should be able to recognize specific situations occurring in the building and react accordingly: one of the systems can control the behavior of others according to a previously developed algorithm. The main feature of such a building is the unification of separate subsystems into a single controlled complex. An important feature and property of a smart house is the way of organizing the living space - this is the most progressive concept of human interaction with the living space, when a person sets the desired environment with one command, and the automation, in accordance with external and internal conditions, sets and monitors the modes of operation of all engineering systems and electrical devices. In this case, there is no need to use several remotes when watching TV, many switches when controlling lighting, separate blocks when controlling ventilation and heating systems, video surveillance and alarm systems.

Keywords: smart home, smart lighting, remote control, software, ESP8266, microcontroller, WI-FI, HTTP.

Introduction

The development of the "smart home" concept was largely due to the rapid development of innovative technologies and the improvement of sensors. The concept of "smart light" is one of the components of the global IoT system - the "Internet of things" - the global infrastructure of the information society. The IoT system provides real-time services by connecting physical and virtual things based on existing and evolving interoperable information and communication technologies. The interaction between physical things is implemented through built-in elements with the appropriate software, which is responsible for collecting and analyzing data, device interaction, communication with the cloud. The method of sending and receiving information, as well as communication with external devices, is determined by the choice of communication solution. The system recognizes what is happening around and informs the user about it through IoT - a platform on which the received data is analyzed, then they are converted into a form convenient for the end user.

"Smart light" takes into account the presence of people in the location, the time of day, the presence or absence of natural light. These parameters can be controlled remotely - turn on or change the brightness, simulate the presence of people in the house. Types of lighting devices can be very different - incandescent, gas discharge, diode. The control elements responsible for adjusting the light are microcontrollers, sensors, motion sensors, dimmers. The controller for a smart home with gsm is essentially an intelligent system that coordinates the work of connected devices, makes logical decisions, the algorithms of which are embedded in the corresponding program.

Analysis of recent research and publications. People tend to follow certain patterns in their daily lifestyle. In the context of a smart home, the daily actions of the user generate patterns that play an important role in predicting future events in the smart home. The purpose of the smart home environment is to help the user in daily life; thus, a smart home must find repetitive patterns in user activity and predict user behavior in order to receive assistance [1].

User activity monitoring is used to observe and record human actions in order to achieve the “comfort and efficiency goal” that a smart home can offer. User behavior refers to the range of actions, actions and responses made by the user. Therefore, a smart home must be able to learn and apply the knowledge gained in order to adapt the home to the user's behavior. Since the user generates a pattern, abnormal user behavior can be detected by building a normal user behavior pattern. Typically, smart home sensors and cameras are used to track or identify user activity and perform analysis of human behavior. User behavior can be used to predict and determine future user actions. Thus, the activity recognition method implemented by a smart home should be as accurate as possible to control the system [2].

Context awareness is an important step in the concept of user behavior and behavior. The context-awareness system in this concept is one that "adapts depending on the place of use, the accumulation of nearby people and objects, and changes in these objects over time." Day A.K. [3] defined context as "any information that can be used to characterize a subject's position", and a context-aware system is one that "uses context to provide 8 relevant information and/or services to the user, where appropriateness depends on the user's task ".

Smart home devices can be controlled both through voice assistants built into smart speakers, and through websites and applications connected to the smart home control center. Unfortunately, in the “many home” market there is no single standard for pairing devices, and therefore, large technology manufacturers are forming their own ecosystem with a list of devices that can be connected. To interface many devices without being tied to the manufacturer of the smart home control center, there are open-source DIY solutions based on the Raspberry single-board computer, discussed in the following sections. However, it should be understood

that this type of solution requires technical knowledge from the user and time costs.

Statement of the research problem

The wireless method uses a local network and a data transfer protocol such as HTTP, WebSocket or its own. In this way, the smart lighting connects to the home local network and tells the user its address. In turn, the user needs to take any device with software for interacting with smart lighting and actually control it. Typically, these control devices are smartphones, computers, and laptops. To implement such interaction, you need to create a set of commands, implement software that uses the specified protocol and processes user requests to send these commands to the microcontroller. Commands are sent in most cases using the HTTP protocol.

HTTP is a protocol that allows you to receive various resources, such as HTML documents. The HTTP protocol is the basis of data exchange on the Internet, and is also a protocol of client-server interaction, which means that requests to the server are initiated by the recipient, usually a web browser [11]. The result obtained has its own body. The request or response body has a specific content type, such as text, HTML, JSON. Obviously, a set of commands must contain a certain set of data. In order for the data to be transferred and analyzed conveniently, you need to use one of the known standards, as this eliminates the need to spend time on the implementation of your own type of content. One of the most common options is JSON.

JSON (JavaScript Object Notation) is a text format for exchanging data between computers. JSON is text-based, human-readable. A format allows you to describe objects and other data structures. This format is mainly used to transmit structured information over a network (thanks to a process called serialization).

JSON is built on two structures:

1. A set of name/value pairs. In various languages, this is implemented as an object, record, structure, dictionary, hash table, keyed list, or associative array;

2. Ordered list of values. In many languages, this is implemented as an array, vector, list, or sequence.

These are universal data structures. In theory, all modern programming languages support them in one form or another. Since JSON is used to exchange data between different programming languages, it makes sense to use this format to implement the communication mechanism between the control device and the smart lighting itself [4,5,6].

Android is the most popular OS in the world for smartphones since 2011 and for tablets since 2013. As of May 2021, it has more than two billion monthly active users, the largest installed base of any operating system, and as of January 2022, Google's Play Store features over 2.9 million apps.

The Android source code is released by Google under an open source license, and its open nature encourages a large community of developers and enthusiasts to use the open source code as the basis for community-driven projects that provide updates for older devices, add new features for users, or port Android on devices originally shipped with other operating systems. These releases often provide new features and device updates faster than through official manufacturer channels, with a comparable level of quality [7, 8].

So, the Android OS is a platform with an open architecture, it has convenient tools for creating software. The extremely large number of users of this OS means that almost every smartphone user uses Android. Openness, popularity, support for HTTP, JSON, HTML, JavaScript standards are sufficient arguments for recognizing this platform as appropriate for the development of software for controlling smart lighting devices.

A microcontroller for "smart lighting" must have certain functionality - work with WI-FI, interact with HTTP protocols and be powerful. Therefore, among a large number of microcontrollers, those created on the basis of ESP were chosen. Today, there are two options that satisfy the conditions - ESP8266 and ESP32.

ESP8266 is a popular platform for "Internet of Things" that need to transmit or receive data to the Internet using WI-FI technologies. In addition to WI-FI, the microcontroller differs from others in the absence of flash memory in a single-chip system, and user programs are executed in flash memory via a serial peripheral interface. In turn, the microcontroller attracts with a pleasant price and low power consumption. One of the popular boards based on ESP8266 is NodeMCU V3.

The ESP32 is an inexpensive microcontroller that replaced the ESP8266. Like its predecessor ESP8266, it has low power consumption. It represents a system on a crystal with integrated WI-FI and Bluetooth controllers and antennas. It exists in two types, single-core with a clock frequency of 160 MHz and dual-core, respectively, with a frequency of 240 MHz. Based on ESP32, the DOIT ESP32 DEVKIT V1 board is most often used.

The NodeMCU V3 board was chosen as the board for installing the microcontroller.

The next step is to choose the programming language and development environment - Arduino IDE. The structure of the software (software) of the microcontroller was divided into separate modules:

- ✓ Interaction with a WI-FI access point;
- ✓ Server part;
- ✓ Initialization of controller devices;
- ✓ Description of effects;
- ✓ Effects manager;
- ✓ Utilities.

First of all, the controller must be connected to the WI-FI network. There are a number of means for this. The Arduino IDE suggests using the ESP8266WiFi library. It provides an easy way to interact with access points. Accordingly, we get the ability to connect to the network, obtain an IP address from the code, and also have the ability to monitor the connection status. Thus, when power is supplied to the controller, it starts up, which in turn initiates the connection to the WI-FI access point. Next, it is necessary to find out the status of the connection and, if successful, to display the IP address on the matrix so that the client part can find out which address to send requests to.

The next module is the server part. This refers to the code that implements API access via a defined protocol with a defined content type and content. An available tool for implementing these principles is the ESP8266Web Server library. One of the main possibilities is to create a REST API with request handlers and sending responses using the HTTP protocol. In this way, the developer gets rid of the need to implement his own protocols. So, to ensure client-server interaction, you need to specify which types and how the microcontroller will respond. Regarding the type and format of the content, it should be noted that the body in requests and (or) responses is necessary only in the case of execution of commands for obtaining the current state of the configuration. When using the API to change the state of the controller, you only need to specify the command parameters.

The microcontroller initialization module is a block of code in which variables are described, objects are created, and methods are executed to ensure the start of work. In this case, you need to describe the constants that characterize the numbers of matrix effects, the size of the matrix itself, the number of LEDs, as well as create objects for interaction with WI-FI, HTTP, JSON, FastLED and perform methods for starting the corresponding processes.

The Description of Light Effects module will be organized as a tab in the Arduino IDE. Structurally, it will be a sequence of descriptions of each effect together with its auxiliary methods.

The effects manager in this case is a module with the task of checking the current effect and calling the appropriate method for its execution. The manager will be used when starting the controller, as well as when processing state change commands received from the client part of the smart lighting device.

The utility module is a set of auxiliary classes and methods to simplify the interaction of the use of libraries, as well as to optimize the code as a whole. In this way, large logical chains of code will be placed in separate methods, so they will be available for reuse from anywhere in the controller software, and as a result, the total amount of code will be significantly reduced.

Basic material and results

At the first stage of software implementation, it is necessary to create a reference architecture to ensure a scaling strategy. For this, the most adequate step is to use a modular structure [9]. This architecture provides opportunities for the independence of individual parts of the software in the event of the need for modifications or changes. For Android OS, the most common architectural design pattern is MVVM. According to the conditions of its implementation, the following directories must be created: model, view, viewmodel, repository, utils. Each of them contains a set of classes to perform the corresponding role:

- Model – entities, classes that represent data sets and an interface for interacting with them according to the principles of encapsulation;
- Repository – classes for interaction with the server, database or other sources of information. These classes contain the code for interaction methods with the back-end part and the organization of their work in a multi-threaded environment;
- View – graphic shell screen handlers;
- ViewModel – classes describing the logic of interaction with data for view components;
- Utils – utility classes to facilitate work and also to avoid redundant code duplication.

Strict rules are imposed on the communication between these classes. First of all, to avoid conflicts and memory leaks, you should avoid references to visual elements inside any module other than view. This need exists as each visual element, as well as the viewModel, has its own life cycles, the work of which must be managed by the system without interference.

For the functionality of the command system in the lighting device control software, all operations with the web server must be performed in a separate thread. For this purpose, we will create the corresponding classes and interfaces. Since a separate stream does not have the ability to directly transfer data to the main one, it is necessary to create a "Callback" interface with a callback method of the same name. It is impossible to create an interface object due to the limitation of the JAVA programming language standard. Creating a large number of classes that implement it is not a good solution, because such an approach introduces the permanence of the method's behavior. Therefore, anonymous classes are used for flexibility in the use of the callback mechanism. Thus, when the use of this interface occurs, the developer must create an anonymous class with the implementation of the callback method. This implementation can then be passed to a separate thread. Accordingly, at a certain point in time, this thread will execute a call to this method.

The next stage of development of the mechanism of interaction with the web server is the development of the repository class with the appropriate methods. To do this, we will create the necessary types of requests and response handlers. Based on the microcontroller software, we can perform the following requests:

• Downloading data of the current state of the controller;

• Setting up new settings.

First of all, we will form the general structure of the methods of executing requests. We will distinguish three main components, such as operations on input data, request formation, sending for execution in a separate stream. The first component consists in performing simple type conversion manipulations and (or) extracting values from entity objects. The second component is the creation of a request by creating the "Request" object of the OkHTTP library. Then the processed data is placed in the headers and (or) body of the request, the content type, HTTP method and other settings are specified. The third component is the execution of a request to the web server. To achieve this goal, the implementation of multithreading processors is required. For this, it is worth creating a class that implements the "Runnable" interface, to the input of which Request and Callback objects will be submitted. The inner class will create a new thread, implementing the request execution and callback, after

receiving the response from the server. Next, we implement certain types of operations.

The request to receive the state of the controller as input data involves only the index of the mode whose data needs to be received. According to the web server specification, the request does not need additional headers and bodies. It is enough to form the correct URL. Processing the response consists of converting the JSON into a "Setting" entity object and then saving it to the appropriate storage.

A configuration change request requires 4 fields as input, namely mode index, brightness, speed, and special property value. All interaction, except for processing the response, is similar to the algorithm described above. Since there is nothing in the response except the status code, its processing is reduced to the output of a corresponding message in the debugging log.

Conclusions

An analysis of the technical characteristics of existing analogs presented on the "smart lighting" market was performed. A number of shortcomings were identified - low connection stability, weak interaction speed and high price for not always justified quality and functionality. As a result of the review of the subject area, the main electronic components necessary for the implementation of a smart lighting device were found - a microcontroller, a matrix, a power supply unit.

Considered ways for software implementation, which indicated the availability of appropriate libraries for connecting to a WI-FI network and deploying a web server within the microcontroller software using the ESP8266WebServer library. The development of the logic of the microcontroller, namely the modes of operation, sequence of actions, modularity, was carried out. As a result, the overall structure of the software was divided into three separate modules: initial initialization, interaction with the network, display of effects on the LED matrix.

The "smart lighting" software is implemented in the form of a mobile application. The mobile application includes an architectural design template and basic visual components. To communicate with the web server, mechanisms for building requests and processing responses from the OkHTTP library are used. The implemented structure of the "smart lighting" device has the possibility of further scaling without additional costs, which in turn increases its competitiveness on the commercial market, can be used as a prototype of a hardware and software complex for the implementation of a budget version of "smart lighting".

REFERENCES

1. Adim A.O. Big Sensed Data Meets Deep Learning for Smarter Health Care in Smart Cities / A. O. Adim, B. Kantarci // Journal of Sensor and Actuator Networks. – 2017. – Volume 6. – Issue 4. - PP. 1-22. – URL: https://doaj.org/article/1067f3eaf1d94db38796b306f62c692f .
2. Flowler R.J. Electricity; Principles and Applications / R.J. Flowler - New-York: Delmar Cengage Learning. 2017. 247 p.
3. Haritaoglu I., Ghost: A human body part labeling system using silhouettes / Haritaoglu I., Harwood D., Davis L. S. // Proc. 14th International Conference on Pattern Recognition, Australia. 2015. pp. 16-20.
4. Format JSON [Electronic resourse].URL: https://timeweb.com/ru/community/articles/format-json // (date of access: 15.02.2023).
5. JavaScript Object Notation.URL: https://www.php.net/manual/ru/json.installation.php // (date of access: 15.02.2023).
6. Autonomous JSON Database [Electronic resourse].URL: https://www.oracle.com/autonomous-database/autonomous-json-database/get-started // (date of access: 15.02.2023.)
7. Android 13 Highlights.URL: https://www.android.com/android-13/#a13-beyond-the-phone // (date of access: 15.02.2023.)
8. Economic Benefits of the Global Positioning System (GPS) [Electronic resourse].URL: https://economic-definition.com/Technology/Android_Android__eto.html // (date of access: 15.02.2023.)
9. Froiz-Míguez, I.; Fernández-Caramés, T.M.; Fraga-Lamas, P.; Castedo, L. Design, Implementation and Practical Evaluation of an IoT Home Automation System for Fog Computing Applications Based on MQTT and ZigBee-WiFi Sensor Nodes. Sensors 2018. 18, 2660. doi.org/10.3390/s18082660

Received (Надійшла) 30.03.2023
Accepted for publication (Прийнята до друку) 10.05.2023

Реалізація функції освітлення у концепції «розумного будинку»

О. В. Скакаліна, А. М. Капитон

Анотація. Принцип «Системи інтелектуального управління будинком» передбачає абсолютно новий підхід в організації життєзабезпечення будинку, в якому за рахунок сукупності програмно-апаратних засобів значно зростає ефективність функціонування і надійність управління всіх систем і виконавчих пристроїв будівлі. Під розумним будинком слід розуміти систему, яка повинна вміти розпізнавати конкретні ситуації, що відбуваються в будівлі, і відповідним чином на них реагувати: одна з систем може управляти поведінкою інших за заздалегідь виробленим алгоритмом. Основною особливістю такої будівлі є об'єднання окремих підсистем в єдиний керований комплекс. Важливою особливістю і властивістю розумного будинку є спосіб організації життєвого простору - це найбільш прогресивна концепція взаємодії людини з житловим простором, коли людина однією командою задає бажану обстановку, а вже автоматика відповідно до зовнішніх і внутрішніх умов задає і відстежує режими роботи всіх інженерних систем і електроприладів. У цьому випадку виключається необхідність користуватися кількома пультами при перегляді ТБ, багатьма вимикачами при управлінні освітленням, окремими блоками при управлінні вентиляційними і опалювальними системами, системами відеоспостереження та сигналізації.

Ключові слова: розумний дім, розумне освітлення, віддалене керування, програмне забезпечення, ESP8266, мікроконтролер, WI-FI, HTTP.