UDC 004.42

M. Kliuba, T. Likhouzova

SOFTWARE TOOLS FOR CREATING INTERFACES FOR INTERACTION WITH ARDUINO VIA BLUETOOTH

Abstract: The problem of communication between microcontrollers of the AVR family and other devices via Bluetooth using special modules is considered. A comprehensive solution is proposed, including a modified interaction protocol, an Arduino library that implements interaction on the microcontroller side, and a mobile application for creating interfaces that allows a user to interact with Arduino via Bluetooth.

Keywords: Arduino, Bluetooth, graphical user interface, software interface on the microcontroller side.

Introduction

The development of the microcontroller market in recent years has led to the fact that they have become more affordable and convenient to use, allowing developers to create more complex and powerful electronic devices in less time. With the spread of microcontrollers, the field of amateur robotics and microelectronics also developed. One of the most popular such platforms is Arduino [1].

Arduino is a software and hardware computing platform for developing electronic devices and systems. It is based on the AVR family of microcontrollers and has open-source code, which enables developers to quickly and easily create electronic devices using ready-made modules and libraries. Arduino supports communication with other devices via Bluetooth using special modules. Moreover, it is possible to send and receive data from a smartphone. There are specialized applications for this. However, the functionality of the latter is limited and does not fully satisfy the needs of users. Often there is a need to customize the application interface for your specific needs or create it yourself. It is also not always obvious how to implement interaction with the application on the Arduino side.

Therefore, there is a need to develop complex software that will allow the creation of a graphical user interface for interacting with Arduino via Bluetooth and providing a convenient software interface on the microcontroller side.

The purpose of the work is to improve tools for creating interfaces that provide user interaction with Arduino via Bluetooth, and the object of research is software tools for creating such interfaces.

Analysis of existing solutions

Designing user interfaces is an important component of software development. Its purpose is to create a convenient and effective interface that will facilitate user interaction with the software product [2]. This includes creating intuitive controls, logical information placement, and cross-platform adaptation.

The user interface design process includes analysis and planning, prototyping, testing, and updating the interface [3]. It is necessary to create an interface that will be attractive, convenient, and intuitive for users. When designing the interface, such factors as appearance, placement of elements, color palette, typography, and animation are taken into account [4]. Various methods and tools are used to determine the optimal placement of elements and ensure easy access to functions.

Therefore, you need to pay attention to the following theses:

– goal: to ensure convenient and effective interaction with the software product;

– aspects: intuitive controls, logical placement of information, adaptation to different platforms;

– the role of accessibility: to ensure accessibility for all users, including persons with disabilities;

– aesthetics: an attractive and aesthetic interface that attracts users and improves their interaction;

– adaptation: flexibility and constant improvement of design to changes in technology and market requirements.

Among the currently available software solutions, several of the most popular and most functional within the framework of the work topic can be singled out, namely:

– a mobile application for the Android operating system (API 23+) from the developer Kai Morich – Serial Bluetooth Terminal [5];

– an application for the Android OS (API 23+) developed by Kenneth Njoma – Bluetooth Controller for Arduino [6];

– a mobile application for Android OS (API 23+) developed by Shemanuev Evgeny – RemoteXY: Arduino control [7].

Based on the analysis, the following features and requirements for the software were determined:

– the software should consist of a mobile application and a library for Arduino to ensure reliable interaction;

– the Arduino library must support work with the most popular Bluetooth modules;

– the mobile application must work on the Android 6 operating system or higher (API level 23+);

– the mobile application must support both regular Bluetooth and BLE with the most popular chips;

– the interface designer should provide an opportunity to create an adaptive interface with various types of control elements: buttons, switches, input fields, sliders, voice input, and others;

– the ability to export and import the created interface;

– appropriate functionality for searching, scanning, and connecting to new devices;

– in general, the mobile application should correspond to the style and general idea of the Arduino platform.

 ISSN 1560-8956

Materials and methods

Arduino is an open hardware and software environment for prototyping electronic devices. The Arduino platform consists of two main components: the Arduino hardware board and the Arduino IDE (integrated development environment) [8]. After connecting the Arduino board to the computer, you can use the Arduino IDE to write program code in the C/C++ language [9]. The platform provides access to various functions and libraries that simplify hardware programming, reading data from sensors, managing outputs, and much more. The Arduino platform also has various additional components such as touch screens, wireless modules (e.g. Wi-Fi, Bluetooth), relays, motor drivers, and many others. These components can be connected to Arduino boards to expand their capabilities and functionality.

Bluetooth technology is a wireless data transmission protocol that allows devices to exchange information over a short distance via radio wave communication [10]. Bluetooth can be used to connect different types of devices such as mobile phones, smartphones, laptops, tablets, headsets, keyboards, mice, car audio systems, and many others. It allows you to wirelessly exchange data and control various devices.

User interface (UI) design technologies include a set of tools, methods, and approaches that help developers create effective, user-friendly, and attractive user interfaces. To simplify the creation of user interfaces, there are special libraries that contain design recommendations and standards. One of these design systems is Material Design, developed by Google [11]. It is also available in Jetpack Compose. Using Material Design helps create stylish, user-friendly, and intuitive apps that help maintain a cohesive and familiar experience for Android users. Using Material Design when designing user interfaces for Android applications helps to create a modern and attractive design that meets Google's standards and recommendations. These principles are intended to facilitate the use of the application by users and improve their experience.

Results and discussion

Development of an interaction protocol

To implement the transfer of information between the Arduino and the smartphone via Bluetooth, it is necessary to develop an interaction protocol. The most optimal option is to use a modified version of the Serial Port Profile (SPP) protocol.

The SPP protocol is used to establish a virtual serial connection (COM port) between two Bluetooth devices. One device acts as a server and the other as a client. The server listens on a specific port, while the client connects to the server using a MAC address or other identification value. After the connection is established, the client and server can transfer data to each other via SPP. This allows devices that support SPP to exchange data directly over a Bluetooth connection. This protocol allows writing a sequence of bytes of data that can encode both digital and string data into a virtual port.

For convenient work with data, the following modification was developed for this protocol, called BM-SPP (BluetoothMe SPP):

– to divide the general data stream into separate commands, a terminal symbol was used - the LF new line symbol - "\n";

– each command consists of a tag and a value, which are separated by the tag terminator symbol - "/";

– a tag is a subject, title, identifier, or other data to recognize and identify this command. For example, "led" for the command to control the LED, "brightness" for controlling the brightness of the lighting or other intuitive tags from the context;

– the value is the meaningful load of the command. For example, for an LED, these can be values of "0" and "1" for the on and off states, respectively.

Creating a program library for Arduino

To implement user interaction with the application on the Arduino side, the BluetoothMe library was developed. This library, using the IBluetoothAdapter interface, encapsulates work with the Bluetooth adapter and provides convenient functions for sending and receiving data. Thanks to this approach, the user gets the opportunity to use any Arduino-compatible Bluetooth module. Figure 1 shows the class diagram of the developed library.

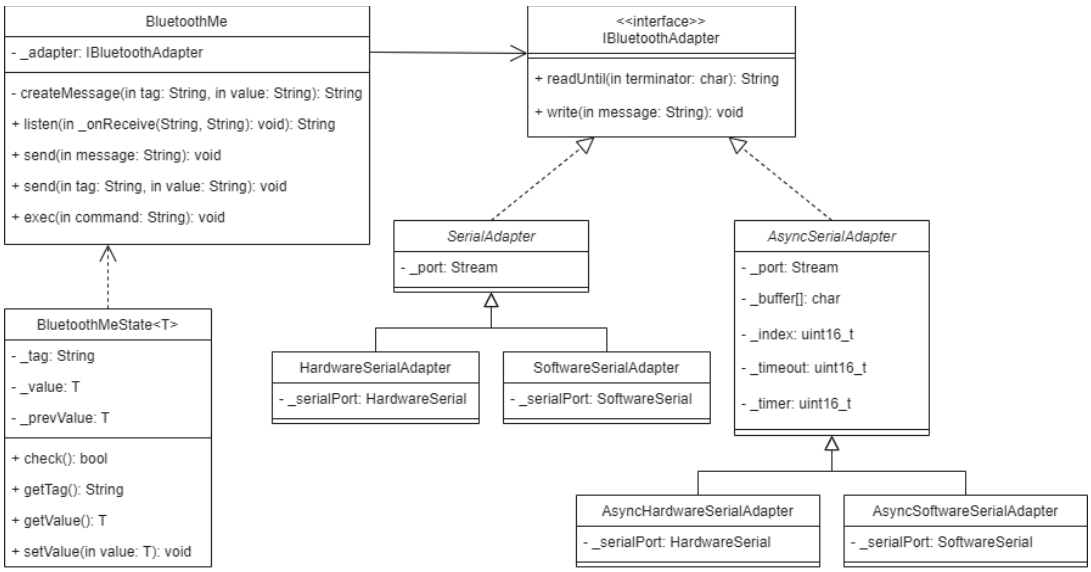

Fig. 1. BluetoothMe library class diagram

Mobile application development

Clean Architecture is a popular approach to software development, including mobile applications for the Android platform. Its main idea is to separate the application code into different levels of dependencies to ensure code transparency, make it easy to understand and testable. The program must contain at least two layers: the user interface layer (UI Layer), which

 ISSN 1560-8956

provides data display on the screen and interaction with the user, and the data layer (Data Layer), which contains business logic and provides access to data. You can also add a domain layer that implements the interaction between the UI and data layers. This architectural approach was used to develop the BluetoothMe mobile application. Figure 2 shows the general structure of the project.

At the domain level are use cases, models, and repository interfaces. All the main elements of the program are described here, written only in pure Kotlin code and without the use of external frameworks and libraries.

The data layer covers the description of different data sources: internal storage and services for data exchange over Bluetooth. It also contains implementations of repositories.

The presentation layer is responsible for the code for user interaction with the various screens of the application. All screens and activities are divided into packages and contain ViewModel code and Composable functions for building the user interface.

Also, to facilitate testing and scaling of the project, the Dagger Hilt library was used, which automates dependency injection (DI).

The interaction of all application layers in the context of data exchange is implemented using Kotlin Flow, a component of the Kotlin programming language that provides asynchronous processing of data streams with the ability to send and receive sequential values. Flow is based on the concept of reactive programming and provides the ability to work with asynchronous data streams.

One of the most important components of the proposed application is the implementation of Bluetooth interaction. For this, several classes and interfaces were created, which are at the domain level. The implementation of these interfaces allows you to create a program module responsible for working with Bluetooth, which has a simple and stable structure.

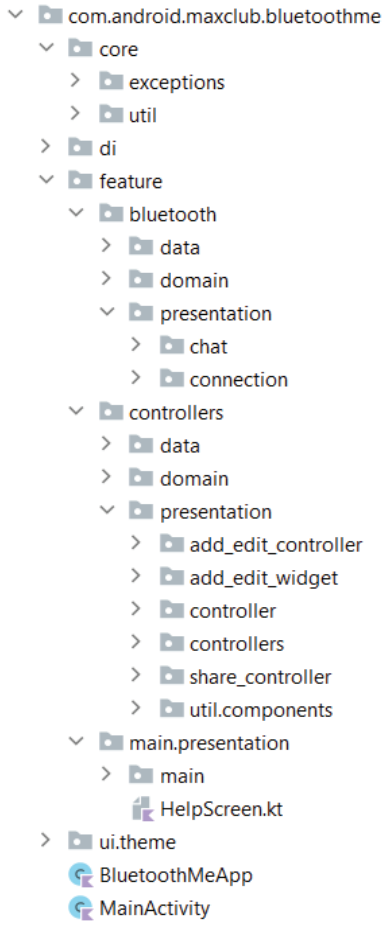

Fig. 2. Project structure

Another aspect worth mentioning in the context of working with Bluetooth when developing a mobile application for the Android operating system is user permissions for using Bluetooth. They depend on the current version of Android on the user's device and all of them must be written in the application manifest.

Another important component of the application is the Controller and Widget entities, which are used to save the created interfaces to the database. Controller is a wrapper, a workspace that contains various control elements - widgets. Widgets can be of different types and are used for direct interaction with the user (for example, buttons, switches, etc.). During the design process, the necessary attributes of these entities were determined and an ER-diagram was constructed in Chen's notation (Figure 3). The relationship between entities is one-to-many.

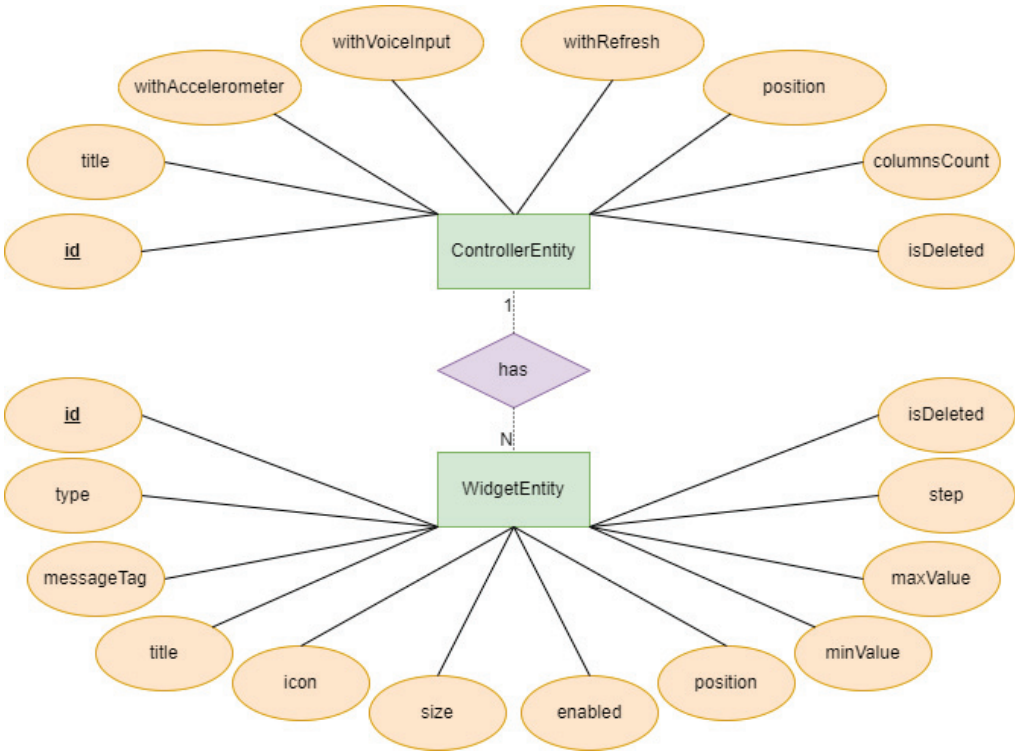

Fig. 3. ER diagram

The designer of user interfaces is implemented in the form of a visual editor. Each user-created interface is stored and displayed in the application as a controller. The controller can contain a conditionally unlimited number of control elements - widgets. Widgets can be of different types and with different functionality. Table 1 describes the available widget types.

Table 1

Name	Description
BUTTON	A non-latching button that has two states: pressed and released.
SWITCH	Two-position switch with locking.
SLIDER	An element for changing values in a certain range. Similar to a scroll bar.
TEXT	A field for entering text from the keyboard.
EMPTY	An empty widget with no control. Can be used as a plug.

 ISSN 1560-8956

In addition to the widgets in the controllers, it is also possible to activate additional tools that will be displayed on the toolbar. Table 2 shows their description.

Table 2

Name	Description
Voice input option (withVoiceInput)	A button for entering text using the voice recognition function.
Refresh option to get current state (withRefresh)	Refresh button to get the current state. When clicked, sends a message with a tag "get/".

Since widgets have different functionality that depends on their type and state, the Widget class was made a generic class. This makes it easy to scale the app and provides the ability to add new types of widgets in the future.

The created controllers are displayed in the application on the "My Controllers" screen (Figure 4). Here you can move them, delete them and go to the editor screen.

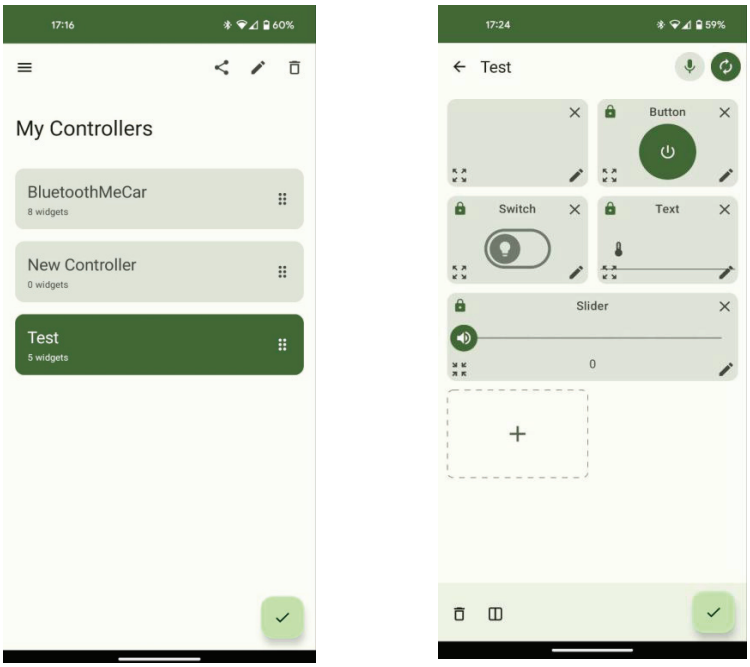

Fig. 4. The process of designing the controller interface

To create a new controller or edit an existing one, go to the controller designer screen. Here you can add new widgets, move and delete them, change the size of the controller grid, activate additional tools and make other settings (Figure 4). From the "My Controllers" screen, you can also open a selected controller by simply clicking on it in the list. The created interface will open on a new screen and will be completely ready for use and interaction with the user.

Each controller can be imported and exported as a custom JSON markup file. Therefore, separate ControllerJson and WidgetJson model classes are used for serialization and deserialization, which are combined in the ControllerWithWidgetJson class. For quick and convenient distribution, it is possible to share interface files in the form of a QR code (this option is available only if the file size is small). In other cases, you can use the JSON file directly. To import the interface, you can use the function of scanning a QR code or selecting a file from the device's storage.

The Jetpack Compose technology and the Material Design 3 style system were used to develop the user interface of the application. The Navigation library is used to navigate between different screens. The ModalNavigationDrawer side menu takes the central place in the application navigation. The elements of this menu are described by objects of the NavDrawerItem class. Figure 5 shows the application navigation map.

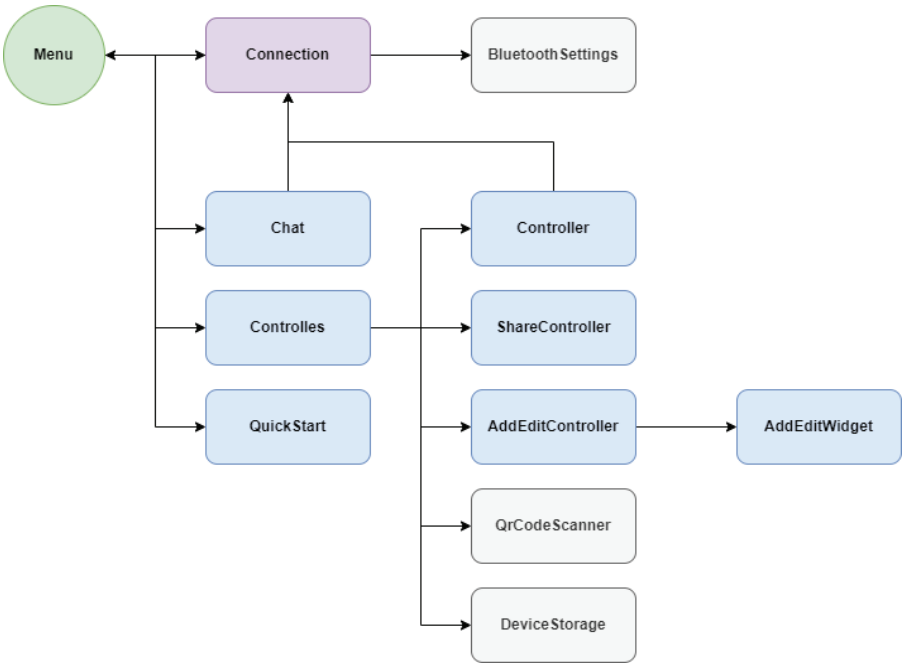

Fig. 5. Application navigation map

To use this application, it is necessary to include the BluetoothMe library on the microcontroller side. To acquaint the user with important information and provide the library code with examples, the "Quick Start" item was added to the menu. When clicking on it, a screen with relevant instructions opens to the user.

Conclusion

The purpose was to improve tools for creating interfaces that provide user interaction with Arduino via Bluetooth. To achieve it:

– a Bluetooth interaction protocol has been developed;

ISSN 1560-8956

– a software library for Arduino has been developed;

– a mobile application in accordance with official standards and conventions has been designed;

– a service for working with Bluetooth and BLE has been created in the application, with which you can search, scan and connect to new devices, send and receive data, monitor the connection status;

– local temporary storage for sent and received data (messages) has been created;

– a database for storing created user interfaces has been designed;

– an adaptive interface designer has been developed for adding, configuring and editing various control elements (widgets);

– export and import of created user interfaces have been implemented;

– a convenient application interface has been designed;

All this together makes it possible to create a graphical user interface for interaction with Arduino via Bluetooth and provide a convenient software interface on the microcontroller side.

References

1. Arduino Documentation. URL: https://docs.arduino.cc/.

2. Bodker, S. (2021). Through the interface: A human activity approach to user interface design. CRC Press.

3. Johnson, J. (2020). Designing with the mind in mind: simple guide to understanding user interface design guidelines. Morgan Kaufmann.

4. Amershi, S., Weld, D., Vorvoreanu, M., Fourney, A., Nushi, B., Collisson, P., ... & Horvitz, E. (2019, May). Guidelines for human-AI interaction. In Proceedings of the 2019 chi conference on human factors in computing systems (pp. 1-13).

5. Serial Bluetooth Terminal. URL: https://play.google.com/store/apps/ details?id=de.kai_ morich.serial_bluetooth_terminal.

6. Bluetooth Controller for Arduino. URL: https://play.google.com/store/apps/details? id=com.don.user.arduino_programmer_pro_fr.

7. RemoteXY: Arduino control. URL: https://play.google.com/store/apps/details?id=com. shevauto.remotexy.free.

8. Banzi, M., & Shiloh, M. (2022). Getting started with Arduino. Maker Media, Inc.

9. Arduino, S. A. (2015). Arduino. Arduino LLC, 372.

10. Bhagwat, P. (2001). Bluetooth: technology for short-range wireless apps. IEEE Internet computing, 5(3), 96-103.

11. Material Design. URL: https://m3.material.io/.