**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ**

**РОССИЙСКОЙ ФЕДЕРАЦИИ**

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ**

**УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**«КАБАРДИНО-БАЛКАРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ**

**им. Х. М. БЕРБЕКОВА»**

**Институт информатики, электроники и робототехники**

**Кафедра компьютерных технологий и информационной безопасности**

Допущен к защите

«\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2021 г.

Зав. кафедройкомпьютерных технологий и информационной безопасности, д.т.н.

Т.Ю. Хаширова

(подпись зав. кафедрой)

**ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА**

**Разработка программного обеспечения для передачи данных по стеку протоколов TCP/IP между устройствами с операционной системой Android**

|  |  |  |
| --- | --- | --- |
| Выполнил:  студент 4 курса  направления 09.03.01 Информатика и вычислительная техника, (Автоматизированные системы обработки информации и управления) | Подпись | Мунчаев О.М. |
| Руководитель:  старший преподаватель кафедры компьютерных технологий и информационной безопасности института информатики, электроники и робототехники КБГУ  Научный консультант:  к.ф.-м.н., доцент, доцент кафедры компьютерных технологий и информационной безопасности института информатики, электроники и робототехники КБГУ | Подпись  Подпись | Акбашева Г.А.  Бозиев О.Л. |

НАЛЬЧИК 2021

Оглавление

[Оглавление 2](#_Toc74567073)

[ВВЕДЕНИЕ 3](#_Toc74567074)

[1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ 5](#_Toc74567075)

[1.1. Обзор аналогичных проектов 5](#_Toc74567076)

[1.2. Обзор существующих средств разработки для платформы Android 7](#_Toc74567077)

[1.3. Роль стека протоколов TCP/IP в развитии сетей 10](#_Toc74567078)

[1.4. Обзор технологий для создания сетевого приложения 11](#_Toc74567079)

[1.4.1. Технология WiFi 12](#_Toc74567080)

[1.4.2 Технология Bluetooth 13](#_Toc74567081)

[2. АНАЛИЗ ТРЕБОВАНИЙ 15](#_Toc74567082)

[2.1. Функциональные требования к проектируемой системе 15](#_Toc74567083)

[2.2. Нефункциональные требования к проектируемой системе 15](#_Toc74567084)

[2.3. Диаграмма вариантов использования 16](#_Toc74567085)

[3. АРХИТЕКТУРА 19](#_Toc74567086)

[3.1. Cпособ хранения полученных данных 19](#_Toc74567087)

[3.2. Компоненты мобильного приложения 19](#_Toc74567088)

[3.3. Описание функциональности 22](#_Toc74567089)

[4. РЕАЛИЗАЦИЯ 25](#_Toc74567090)

[4.1. Реализация системы хранения файлов 25](#_Toc74567091)

[4.2. Реализация файлового менеджера 26](#_Toc74567092)

[4.3. Реализация поддержки Bluetooth 30](#_Toc74567093)

[4.4. Реализация поддержки WiFi 34](#_Toc74567094)

[4.5. Передача файлов по сети 35](#_Toc74567095)

[ЗАКЛЮЧЕНИЕ 41](#_Toc74567096)

[ЛИТЕРАТУРА 42](#_Toc74567097)

ВВЕДЕНИЕ

**Актуальность темы**

Сегодняшние реалии жизни требуют от человека быть в курсе всех последних событий, новостей финансового и политического мира, а также незамедлительно реагировать на любые изменения, происходящие в мире. Человек нуждается в постоянном обмене данными. Ярким примером такой зависимости от информационных каналов связи можно назвать трейдерство. Человек, играющий на бирже, должен владеть всеми сведениями, которые влияют на котировки акций. Больше того, ему нужен Интернет, чтобы вовремя внести изменения в свои фишки, иначе он не получит прибыль. Благодаря тому, что сейчас активно развиваются кабельные, спутниковые и мобильные линии связи, такой человек может иметь постоянно работающий канал, а нередко даже и резервный, на всякий случай. Этот пример доказывает актуальность темы исследования.

В наше время все больше и больше смартфонов, коммуникаторов, планшетных ПК и других видов устройств, удобных для использования как в повседневной жизни, так и в заграничных поездках в частности, выпускаются на базе ОС Android. Каковы же причины распространения данной операционной системы?

Во-первых, Android поддерживает большое количество устройств разных производителей. Во-вторых, Android характеризуется высокой доступностью средств разработки. Средства разработки для платформы Android бесплатны, в то время как разработка, к примеру, под iPhone (от компании Apple) требует немалых начальных финансовых вложений. Кроме всего вышеперечисленного, преимуществом ОС Android является наличие бесплатных библиотек для работы со сторонними ресурсами (YandexMapKit, Google Map API, др.), в то время как для Windows Phone Mobile такие библиотеки не распространены [1]

**Цель и задачи**

Целью работы является разработка программного обеспечения для передачи данных между двумя устройствами. Для этого в рамках дипломной работы должны были быть решены следующие задачи:

1) осуществить постановку задачи, выделить требования к приложению

2) изучить особенности разработки программного обеспечения на стеке протоколов tcp/ip;

3) изучить современные средства разработки приложений для Android;

6) определить требования и спроектировать программное обеспечение;

7) реализовать программный продукт.

**Структура и объем работы**

Работа состоит из введения, четырех разделов, заключенияи библиографии. Объем работы составляет 40 страниц, объем библиографии – 31 источник.

Раздел «Анализ предметной области» описывает анализ предметной области, в рамках которой проводилась работа.

Раздел «Анализ требований» описывает этап определения требований к разрабатываемой системе. Он состоит из анализа функциональных и нефункциональных требований к системе, а также вариантов эксплуатирования системы.

Раздел «Архитектура» описывает архитектуру системы. В данной главе подробно описана архитектура всего приложения

Раздел «Реализация» посвящен описанию всех компонентов системы,

В заключении описываются основные результаты работы.

1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

Целью данной работы является разработка программного обеспечения для передачи данных, в частности для передачи файлов с файловой системы устройства. В рамках текущей работы предлагается следующее решение поставленной задачи: приложение, в функционал которого входит выбор необходимых файлов на одном устройстве и прием этих файлов на другом.

1.1. Обзор аналогичных проектов

Проведен анализ уже используемых приложений для платформы Android, созданных для передачи данных между устройствами. Для анализа было выбрано несколько наиболее популярных приложений

**ShareMe**

Это приложение Xiaomi.  Чтобы отправить файл на другой телефон, другому человеку также необходимо установить приложение. Для обмена просто отсканируйте QR-код, чтобы обмениваться данными через Bluetooth или Wi-Fi.

Между тем, опция Webshare создает специальную точку доступа. Затем другие устройства, включая ПК, могут получить доступ к нему для получения файлов. Это немного неудобно, так как вам нужно каждый раз генерировать новый SSID и пароль, но это работает нормально.

ShareMe работает с фото (одиночными и папками фотографий), аудиозаписями (отдельные файлы и альбомы), видео, приложениями, опциями памяти. Любые выбранные файлы через это приложение могут быть отправлены как на компьютер, так и на смартфон.

**Xender**

Xender – приложение для беспроводной передачи данных с одного устройства на другое. Оно достаточно популярное наряду с похожими программами Pushbullet, AirDroid, Chirp.

Xender в своей работе использует протоколы беспроводной передачи данных, которые достигают скорости, в 50 раз превышающей скорость Bluetooth. Единственное условие – программа должна устанавливаться на отправляющем и принимающем устройствах. При помощи неё можно перекидывать музыкальные и видео файлы, обмениваться фото, приложениями и телефонными контактами.

В одно и то же время без повторного подключения можно работать с 5-ю устройствами. One Key Phone Change – функция, предусмотренная в этом приложении, позволяющая пользователю легко переносить все данные со старого смартфона на новый. А опция Slide позволяет перетягивать изображения с одного устройства на иное при помощи движения пальцем.

[**SHAREit**](https://www.ushareit.com/)

Эта программа является предустановленной на многих устройствах Lenovo на Android. Но ее можно скачать и отдельно. SHAREit работает, создавая точку доступа (WiFi direct), а затем передает с ее помощью файлы. Помимо этого есть возможность сканирования QR-кодов для отправки или принятия файлов.У приложения больше миллиарда пользователей, и все они доверяют передачу данных по всему миру.

Здесь есть возможность использования кода для соединения и обмена файлами. К тому же, можно обмениваться файлами сразу с несколькими людьми, общаться в чате, использовать удалённую камеру для того, чтобы наблюдать, что происходит в телефоне того или иного человека.

**DuktoR6**

Dukto - это простой инструмент для передачи файлов, разработанный для использования в локальной сети. Используйте его в повседневной работе для передачи файлов с одного ПК на другой, не беспокоясь о пользователях, разрешениях, операционных системах, протоколах, клиентах, серверах и т. д. Просто запустите dukto на двух ПК и перенесите файлы и папки, перетащив их в окно. Это действительно очень быстро.

1.2. Обзор существующих средств разработки для платформы Android

На данный момент существует несколько платформ для создания приложений для Android:

1) AndroidStudio [25], основанная на языке Java и AndroidSDK [23];

2) Eclipse [27], поддерживающийязык Java и Android Development Tools [1];

3) Xamarin [30], основанная на языке программирования C# и платформе .NET.

4) IntellijIDEA [31], основанная на языке Java и AndroidSDK[23].

**Android SDK**

Android SDK состоит из разнообразных библиотек, документации и инструментов, которые помогают разрабатывать программное обеспечение для платформы Android.

1) API Android SDK – API библиотеки Android, предоставляемые для разработки программного обеспечения;

2) Документация SDK – включает обширную справочную информацию, описывающую, что включено в каждый пакет и класс и как этим пользоваться при разработке программного обеспечения;

3) AVD (Android Virtual Device) – интерактивный эмулятор мобильного устройства Android. Используя этот эмулятор, можно запускать и тестировать приложения без использования реального Androidycтpoйcтвa;

4) Development Tools – SDK включает несколько инструментальных средств для разработки, которые позволяют компилировать и отлаживать проектируемое ПО;

5) Sample Code – Android SDK предоставляет стандартные приложения, которые демонстрируют многие возможности Android, и простые программы, которые показывают, как использовать индивидуальные особенности API в вашем коде

**IntelliJIDEA**

IntelliJIDEA - интегрированная среда разработки программного обеспечения для многих языков программирования, в частности Java, JavaScript, Python, разработанная компанией JetBrains.

Первая версия появилась в январе 2001 года и быстро приобрела популярность как первая среда для Java с широким набором интегрированных инструментов для рефакторинга[6], которые позволяли программистам быстро реорганизовывать исходные тексты программ. Дизайн среды ориентирован на продуктивность работы программистов, позволяя сконцентрироваться на функциональных задачах, в то время как IntelliJ IDEA берёт на себя выполнение рутинных операций.

**AndroidStudio**

AndroidStudio- интегрированная среда разработки (IDE) для работы с платформой Android, анонсированная 16 мая 2013 года на конференции Google I/O.

Данная IDE находилась в свободном доступе начиная с версии 0.1, опубликованной в мае 2013, а затем перешла в стадию бета-тестирования, начиная с версии 0.8, которая была выпущена в июне 2014 года. Первая стабильная версия 1.0 была выпущена в декабре 2014 года, тогда же прекратилась поддержка плагина Android Development Tools (ADT) для Eclipse.

Android Studio, основанная на программном обеспечении IntelliJ IDEA от компании JetBrains, — официальное средство разработки Android приложений[2]. Данная среда разработки доступна для Windows, macOS и GNU/Linux[3]. 17 мая 2017, на ежегодной конференции Google I/O, Google анонсировал поддержку языка Kotlin, используемого в Android Studio, как официального языка программирования для платформы Android в дополнение к Java и С++.

**Eclipse**

Eclipse - свободная интегрированная среда разработки модульных кроссплатформенных приложений. Развивается и поддерживается EclipseFoundation.

Наиболее известные приложения на основе EclipsePlatform — различные «EclipseIDE» для разработки ПО на множестве языков (например, наиболее популярный «JavaIDE», поддерживавшийся изначально, не полагается на какие-либо закрытые расширения, использует стандартный открытый API для доступа к EclipsePlatform).

**Xamarin**

Xamarin- это фреймворк для кроссплатформенной разработки мобильных приложений (iOS, Android, Windows Phone) с использованием языка C#. Это инструмент для создания приложений на языках семейства .NET (C#, F#, Visual Basic).

Он позволяет писать единый код (Xamarin.Forms), который будет работать на Android, iOS и UWP (Universal Windows Platform — технология создания приложений для Windows 10), или нативные программы для этих платформ (Xamarin).

1.3. Роль стека протоколов TCP/IP в развитии сетей

В настоящее время TCP/IP является самым популярным стеком протоколов. В состав этого стека входят наиболее часто используемые протоколы, которые обсуждаются в данной книге. В большинстве приложений не реализована поддержка нескольких стеков протоколов, поэтому чаще всего приложение может работать с одним конкретным стеком.

Одна из причин популярности стека протоколов TCP/IP — его гибкость. Протоколы TCP/IP являются маршрутизируемыми протоколами, т.е. пакеты TCP/IP могут передаваться из одной локальной сети в другую. Для передачи пакетов между различными сетями не нужна единая карта Internet; при маршрутизации используется распределенная информация о структуре сети, хранящаяся на различных маршрутизаторах. Число допустимых адресов в сетях TCP/IP достаточно велико (в IPv4 адрес представляется 32 битами, а в IPv6 используются 128-битовые адреса; подробно IP-адреса рассматривались в главе 2), кроме того, в этих сетях поддерживается иерархическая структура имен. Эти положительные качества стали причиной того, что протоколы TCP/IP были выбраны в качестве основы для создания глобальной сети Internet.

Впервые протоколы TCP/IP были использованы в UNIX; система Linux "унаследовала" их. Как в Linux, так и в UNIX средства TCP/IP используются для обеспечения работы различных компонентов системы. Сеть, в состав которой входят только компьютеры, работающие под управлением UNIX или Linux, может быть создана на основании TCP/IP, без использования других стеков протоколов.

В состав семейства TCP/IP входят HTTP, FTP, SMTP (SimpleNetworkMailProtocol — простой протокол передачи почтовых сообщений), NFS (NetworkFileSystem — сетевая файловая система), Telnet, SSH (SecureShell — защищенная оболочка), NNTP (NetworkNewsTransferProtocol — протокол передачи сетевых новостей), XWindow и многие другие протоколы.

Широкое использование TCP/IP привело к тому, что в инструментах, изначально ориентированных на работу с другими стеками протоколов, была реализована поддержка TCP/IP. Например, несмотря на то, что в системе Windows используется стек протоколов NetBEUI (NetBIOSExtended User Interface — расширенный пользовательский интерфейс NetBIOS), средства поддержки протоколов SMB (Server Message Block — блок сообщений сервера) / CIFS (Common Internet Filesystem — общая межсетевая файловая система) могут взаимодействовать с TCP/IP через NetBIOS (Network Basic Input/Output System — базовая сетевая система ввода-вывода). Начиная с Windows 95 все версии Windows поддерживают TCP/IP. Аналогично, протоколы Apple, предназначенные для разделения файлов, могут работать не только с AppleTalk, но и с TCP/IP.

Несмотря на свои достоинства и популярность, стек TCP/IP не позволяет решить все задачи, возникающие при создании сетей. Так, например, в некоторых сетях могут присутствовать компьютеры, не поддерживающие TCP/IP. В частности, старые компьютеры Macintosh обеспечивают обмен файлами только средствами AppleTalk, а некоторые машины под управлением DOS или Windows могут быть сконфигурированы лишь для работы с IPX или NetBEUI. Поэтому поддержка альтернативных средств протоколов является положительным качеством системы Linux, котороя в свою очередь является частью Android.

1.4. Обзор технологий для создания сетевого приложения

1.4.1. Технология WiFi

WiFi - технология беспроводной локальной сети с устройствами на основе стандартов IEEE 802.11. Логотип Wi-Fi является торговой маркой Wi-Fi Alliance. Под аббревиатурой Wi-Fi (от английского словосочетания Wireless Fidelity[2], которое можно дословно перевести как «беспроводная точность») в настоящее время развивается целое семейство стандартов передачи цифровых потоков данных по радиоканалам.

Основными диапазонами Wi-Fi считаются 2.4 ГГц /12 см (2412 МГц-2472 МГц) и 5 ГГц /5 см (5160-5825 МГц). Сигнал Wi-Fi может передаваться на километры даже при низкой мощности передачи, но для приема Wi-Fi сигнала с обычного Wi-Fi маршрутизатора на далеком расстоянии нужна антенна с высоким коэффициентом усиления (например параболическая антенна или Wi-Fi пушка).

Обычно схема сети Wi-Fi содержит не менее одной точки доступа и не менее одного клиента. Также возможно подключение двух клиентов в режиме точка-точка (Ad-hoc), когда точка доступа не используется, а клиенты соединяются посредством сетевых адаптеров «напрямую». Точка доступа передаёт свой идентификатор сети (SSID) с помощью специальных сигнальных пакетов на скорости 0,1 Мбит/с каждые 100 мс. Поэтому 0,1 Мбит/с — наименьшая скорость передачи данных для Wi-Fi. Зная SSID сети, клиент может выяснить, возможно ли подключение к данной точке доступа. При попадании в зону действия двух точек доступа с идентичными SSID приёмник может выбирать между ними на основании данных об уровне сигнала. Стандарт Wi-Fi даёт клиенту полную свободу при выборе критериев для соединения. Более подробно принцип работы описан в официальном тексте стандарта.

Однако стандарт не описывает всех аспектов построения беспроводных локальных сетей Wi-Fi. Поэтому каждый производитель оборудования решает эту задачу по-своему, применяя те подходы, которые он считает наилучшими с той или иной точки зрения. Поэтому возникает необходимость классификации способов построения беспроводных локальных сетей.

По способу объединения точек доступа в единую систему можно выделить:

1) Автономные точки доступа (называются также самостоятельные, децентрализованные, умные)

2) Точки доступа, работающие под управлением контроллера (называются также «легковесные», централизованные)

3) Бесконтроллерные, но не автономные (управляемые без контроллера)

По способу организации и управления радиоканалами можно выделить беспроводные локальные сети:

1) Со статическими настройками радиоканалов

2) С динамическими (адаптивными) настройками радиоканалов

3) Со «слоистой» или многослойной структурой радиоканалов

1.4.2 Технология Bluetooth

Bluetooth - производственная спецификация беспроводных персональных сетей (Wireless personalareanetwork, WPAN). Bluetooth обеспечивает обмен информацией между такими устройствами, как персональные компьютеры (настольные, карманные, ноутбуки), мобильные телефоны, интернет-планшеты, принтеры, цифровые фотоаппараты, мыши, клавиатуры, джойстики, наушники, гарнитуры и акустические системы на надёжной, бесплатной, повсеместно доступной радиочастоте для ближней связи. Bluetooth позволяет этим устройствам сообщаться, когда они находятся друг от друга в радиусе около 100 м в старых версиях протокола и до 1500 м начиная с версии Bluetooth 5[4]. Дальность сильно зависит от преград и помех, даже в одном помещении.

Принцип действия основан на использовании радиоволн. Радиосвязь Bluetooth осуществляется в ISM-диапазоне (англ. Industry, Science andMedicine), который используется в различных бытовых приборах и беспроводных сетях. Частоты Bluetooth: 2.402-2.48 ГГц (в мегагерцах 2402—2480 МГц).[12][13]. В Bluetooth применяется метод расширения спектра со скачкообразной перестройкой частоты[14] (англ. FrequencyHoppingSpreadSpectrum, FHSS). Метод FHSS прост в реализации, обеспечивает устойчивость к широкополосным помехам, а оборудование недорогое.

В ходе анализа предметной области были рассмотрены аналогичные проекты, а также средства и технологии разработки для ОС Android. В результате было решено использовать Android Studio, Android SDK и язык программирования Java для реализации программного обеспечения. Для реализации сопряжения двух устройств были выбраны технологии WiFi и Bluetooth

2. АНАЛИЗ ТРЕБОВАНИЙ

2.1. Функциональные требования к проектируемой системе

Функциональные требования определяют функциональность ПО, которую нужно построить, чтобы пользователи смогли выполнить свои задачи. В ходе анализа были выявлены следующие требования:

1) приложение OmarSender должно предоставить возможность пользователю выбрать сторону коммуникации: отправитель или получатель;

2) приложение OmarSender должно управлять Bluetooth адаптером устройства;

3) приложение OmarSender должно управлять WiFi адаптером устройства;

4) приложение OmarSender должно показывать все доступные устройства с включенным Bluetooth;

5) приложение OmarSender должно подключаться к любым доступным устройствам;

6) приложение OmarSender должно принимать запросы на подлючение;

7) приложение OmarSender должно отображать файловую систему устройства

8) приложение OmarSender должно отправлять все выбранные файлы на указанное устройство

9) приложение OmarSender должно принимать все отправленные файлы от устройства

2.2. Нефункциональные требования к проектируемой системе

Нефункциональные требования описывают свойства и ограничения, накладываемые на систему. Для реализации приложения были выявлены следующие требования:

1) приложение OmarSender должно быть написано на языке Java с использованием Android SDK;

2) приложение OmarSender должно работать на платформе Android

версии 7.1 и выше;

3) приложение OmarSender должно соединять два устройства с помощью технологий Bluetooth и WiFi.

2.3. Диаграмма вариантов использования

В ходе анализа был выделено два основных актера системы – Получатель и Отправитель (рис. 1). Получатель может использовать функции системы, которые представлены ниже:

1) Выполнить вход: Получатель может осуществить вход в приложение;

2) Стать получателем данных: получатель может ожидать запроса на подключение

3) Принять запрос на подключение: получатель может принять запрос на подключение от любого устройства

4) Получить данные: получатель может получить данные от подключенного устройства

5) Посмотреть результат: получатель может посмотреть принятые файлы в отдельной папке

Отправитель, в свою очередь может использовать такие функции как:

1) Выполнить вход: отправитель может осуществить вход в приложение;

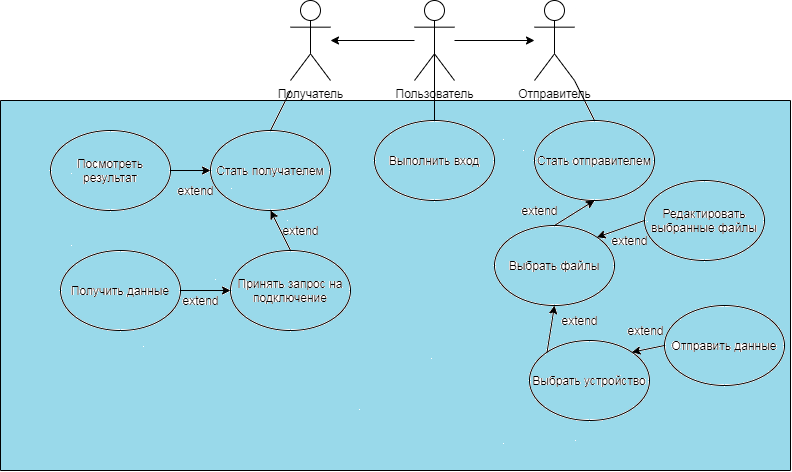
2) Стать отправителем данных: отправитель может начать выбор файлов

3) Выбрать файлы: отправитель может выбирать любые доступные файлы на устройстве

4) Редактировать выбранные файлы: отправитель может удалить из набора любые файлы;

5) Выбрать устройство для сопряжения: отправитель может выбрать любое доступное устройство с включенным Bluetooth

6) Отправить данные: отправитель может начать отправлять данные на выбранное устройство

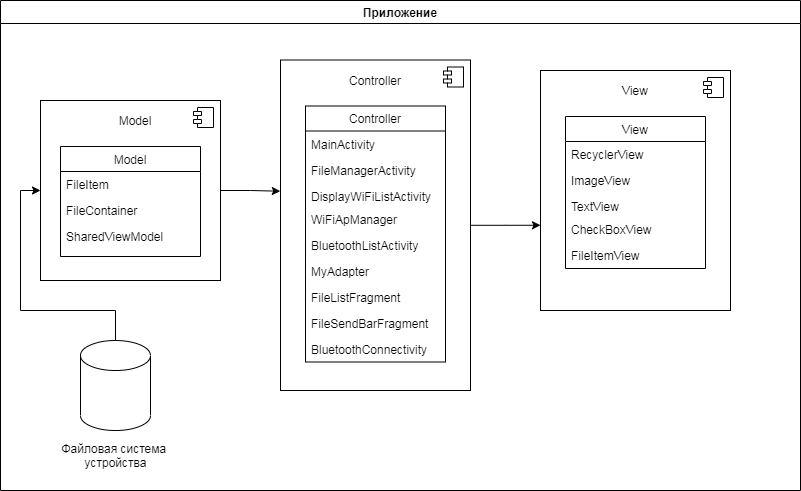


**Рис.1**. Диаграмма вариантов использования

В ходе анализа были установлены основные функциональные и нефункциональные требования к системе, определены пользователь и варианты использования системы.

3. АРХИТЕКТУРА

На рис. 2 представлена архитектура системы.



**Рис.2.** Архитектура приложения

Разрабатываемое программное обеспечение состоит из программных компонентов и директории для хранения полученных файлов

3.1. Cпособ хранения полученных данных

В ходе разработки было рассмотрено два потенциальных способа хранения полученных файлов:

1) Хранить файлы в базе данных

2) Создать специальную папку в файловой системе устройства и сохранять все файлы в нее.

Было принято второе решение, так как использование БД посчиталось избыточным для поставленной задачи.

3.2. Компоненты мобильного приложения

Архитектура приложения будет использовать паттерн проектирования Model-View-Controller (MVC)

Приложения Android строятся на базе архитектуры, называемой «Модель-Представление-Контроллер», или сокращенно MVC (Model-View-Controller). Согласно канонам MVC, каждый объект приложения должен быть объектом модели, объектом представления или объектом контроллера.

Объект модели содержит данные приложения и «бизнес-логику». Классы модели обычно проектируются для моделирования сущностей, с которыми имеет дело приложение, пользователь, продукт в магазине, фотография на сервере, вопрос «да/нет» и т. д. Объекты модели ничего не знают о пользовательском интерфейсе; их единственной целью является хранение данных и управление ими.

В приложениях Android классы моделей обычно создаются разработчиком для конкретной задачи. Все объекты модели в вашем приложении составляют его уровень модели.

Объекты представлений умеют отображать себя на экране и реагировать на ввод пользователя, например касания. Простейшее правило: если вы видите что-то на экране, значит, это представление.

Android предоставляет широкий набор настраиваемых классов представлений. Разработчик также может создавать собственные классы представлений. Объекты представления в приложении образуют уровень представления.

Объекты контроллеров связывают объекты представления и модели; они содержат «логику приложения». Контроллеры реагируют на различные события, инициируемые объектами представлений, и управляют потоками данных между объектами модели и уровнем представления.

В Android контроллер обычно представляется субклассомActivity, Fragment или Service

В разрабатываемом приложении компонент View включает следующие экраны:

1) RecyclerView – отображает списки доступных файлов и устройств для сопряжения

2) ImageView – отображает различные картинки

3) ChekBoxView – отображает флажок на чекбоксе

4) TextView – отображает текстовые данные

5) FileItemView – отображает элементы файлов

Компонент Controller состоит из:

1) MainActivity – Основная активность приложения

2) FileManagerActivity – активность файлового менеджера

3) BluetoothListActivity – активность для отображения Bluetoothустройст

4) MyAdapter – адаптер для списков

6 DisplayWiFiListActivity – активность для отображения доступных WiFi точек

7) FileListFragment – фрагмент отображения файловой системы

8) FileSendBarFragment – фрагмент счетчика файлов и отправки

9) BluetoothConnectivity – отвечает за подключение к Bluetooth устройству

10) WiFiApManager – отвечает за подключение к указанной точке доступа WiFi

Компонент Model состоит из:

1) FileItem – отвечает за хранение информации о файле

2) FileContainer – отвечает за хранение выбранных пользователем файлов

3) SharedViewModel – отвечает за передачу и хранение информации во время общения фрагментов

3.3. Описание функциональности

Специфика приложений для передачи данных заключается в возможности выбора необходимых файлов, поиска доступных устройств, сопряжения с этими устройствами и передаче соответствующих данных.

С учетом перечисленных и других требований к системам для передачи данных и на основе анализа некоторых существующих систем, были поставлены следующие цели разработки приложения:

разработка подсистемы для возможности выбора соответствующих данных с устройства;

разработка подсистемы для возможности поиска доступных устройств;

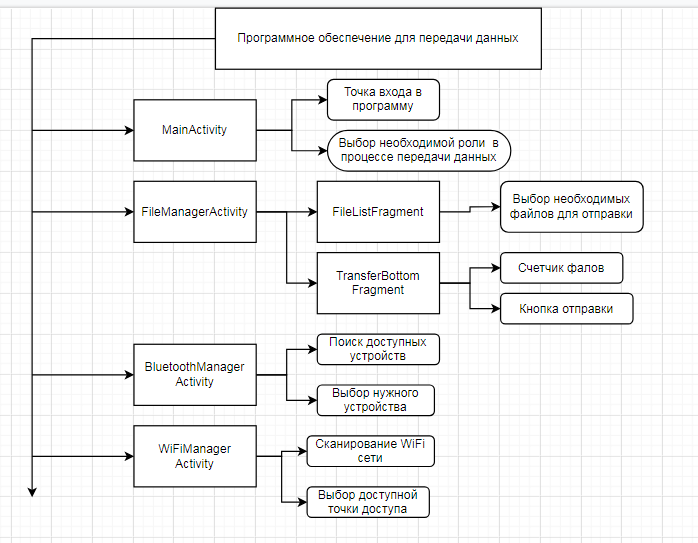
В соответствии с данными целями сформирован перечень основных задач, которые должна решать проектируемая система:

выбор необходимых файлов из файловой системы устройства;

управление Bluetooth соединениями;

управление WiFi соединениями;

прием и отправка файлов по сети;



**Рис. 3**. Функциональная схема приложения

В процессе проектирования системы необходимо составить описание состава действий, а именно совокупности взаимосвязанных мероприятий и работ, направленных на создание конечного продукта. Для примера коротко опишем пять первых пунктов процесса передачи файлов с одного устройства на другое

1. Пользователь нажимает на кнопку “SEND” в MainActivity приложения. Приложение создает новую активность FileManagerActivity

2. В FileManagerActivity помещается два фрагмента ListFragment и SendBarFragment

3. Пользователь выбирает необходимые файлы перемещаясь по директориям устройства, при нажатии кнопки назад состояние выбранных файлов не меняется.

4. Пользователь нажимает на кнопку “Transfer” и переходит в BluetoothManagerActivity, где должен выбрать нужное ему устройство

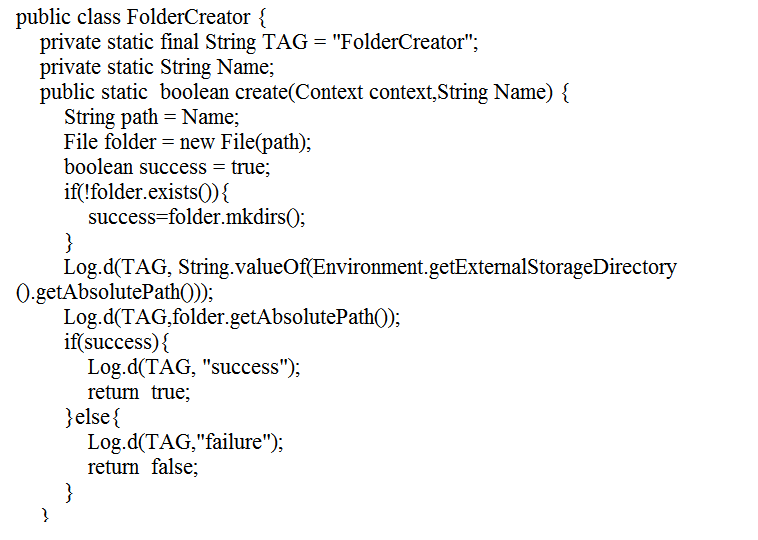
5. После нажатия кнопки, на второй телефон отправляется уведомление о попытке сопряжения, при соглашении сопряжения начинается передача данных.

В результате была спроектирована архитектура программного обеспечения. Приложение включает в себя компоненты Model, View и Controller, а также папку для хранения полученных файлов.

4. РЕАЛИЗАЦИЯ

4.1. Реализация системы хранения файлов

Для создания папки хранения был написан класс FolderCreator. Он имеет основной метод create, который создает папку с соответствующим именем в файловой системе, если папка еще не создана. Результат работы метода create изображен на рисунке 4.



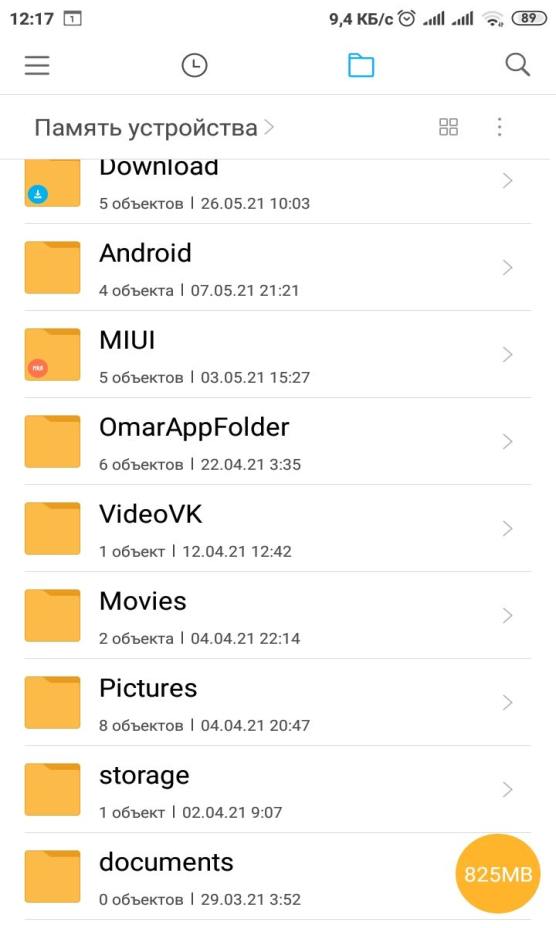


Рис. 4. Папка для хранения файлов

4.2. Реализация файлового менеджера

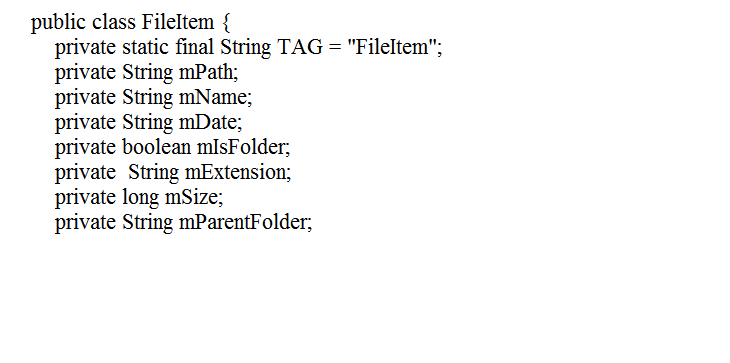
**Классы хранения данных о файлах**

Для работы с файлами было реализовано три класса:

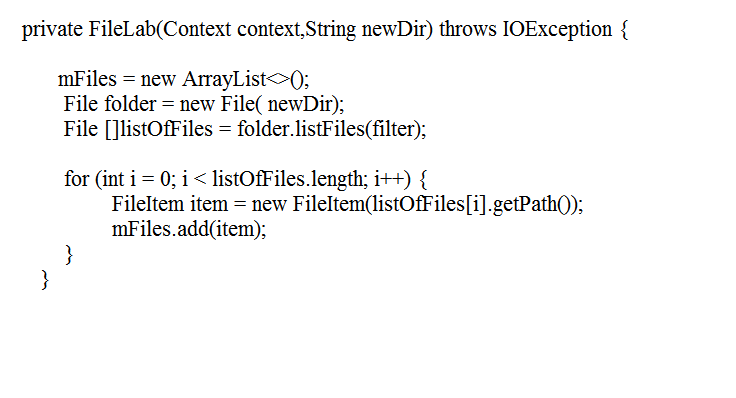
1) FileItem – класс для хранения информации об одном файле или директории

2) FileContainer – класс для хранения всех выбранных пользователем файлов

3) FileLab – класс, который получает информацию о всех файлах в директории



Класс FileLab содержит список из всех файлов в директории, который заполняется в конструкторе.

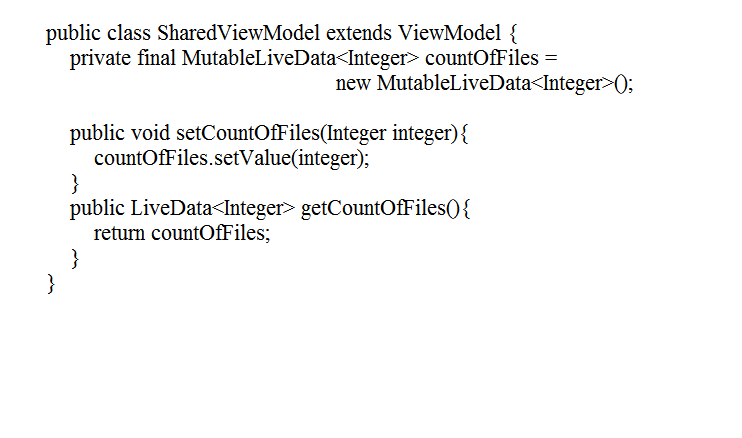


**Главная активность файлового менеджера**

FileManagerActivity состоит из двух фрагментов: FileListFragment и FileSendBarFragment. При создании активности мы заполняем каждый фрагмент.

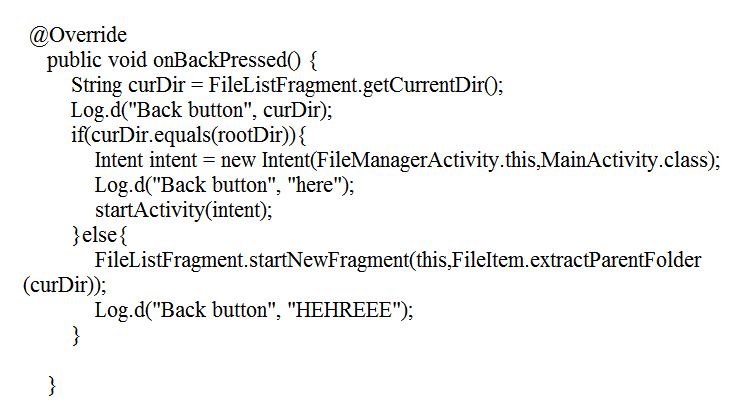
Затем, когда пользователь нажимает на нужную директорию FileListFragment создается еще раз с новым значением корневой папки.

При добавлении или удалении очередного файла из контейнера, с помощью класса SharedViewModel в FileSendBarFragment отправляется текущее число файлов.



**Проблема сохранения состояния**

В ходе разработки было выявлено, что при нажатии на кнопку назад, в списке не отмечались ранее выбранные файлы. Для устранения этой проблемы был реализован метод onBackPressed.



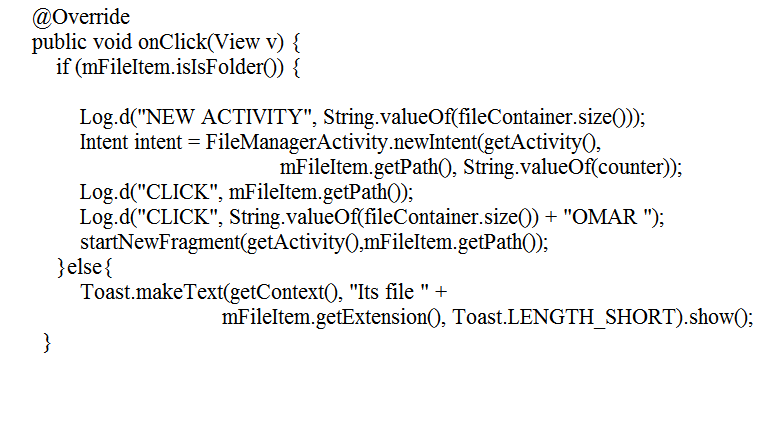
**Вывод списка файлов**

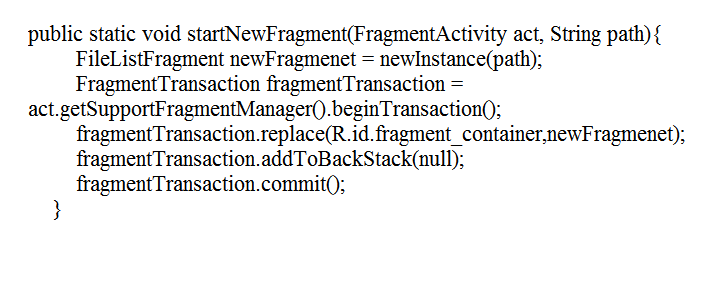
Класс FileListFragment содержит в себе два вспомогательных внутренних класса:

1) FileHolder – класс для хранения данных об элементе списка

2) FileAdapter – класс для заполнения списка

Оба наследуются от полей класса RecyclerView и реализованы для корректного отображения элементов списка. Для того, чтобы перемещаться по файловой системе устройства был реализован метод OnClick в классе FileHolder. Он проверяет является ли выбранный элемент списка директорией, и при успешной проверке запускает фрагмент FileListFragment с новым значением корневой директории.

 Запуск нового фрагмента осуществляется с помощью метода startNewFragment. Вторым аргументом этого метода является выбранная пользователем директория.



На рисунке 5 изображена активность файлового менеджера.

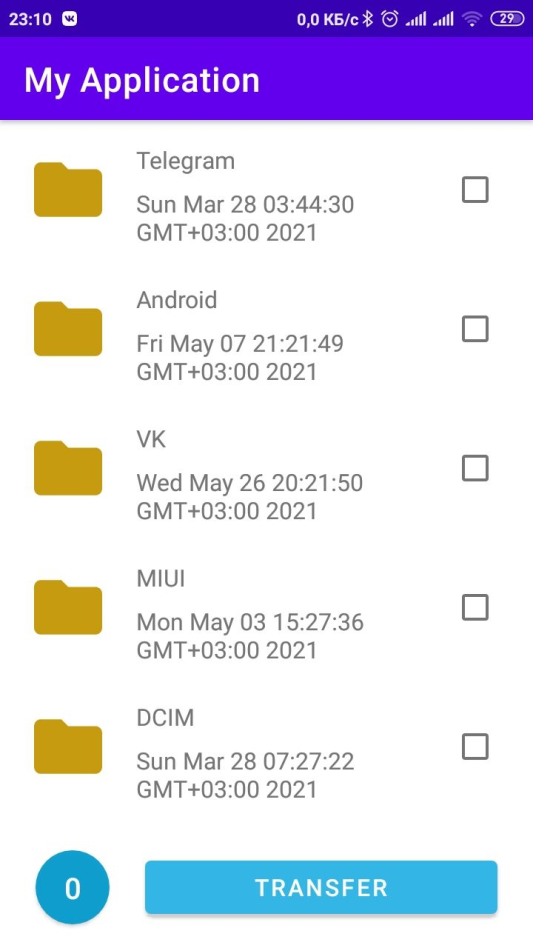


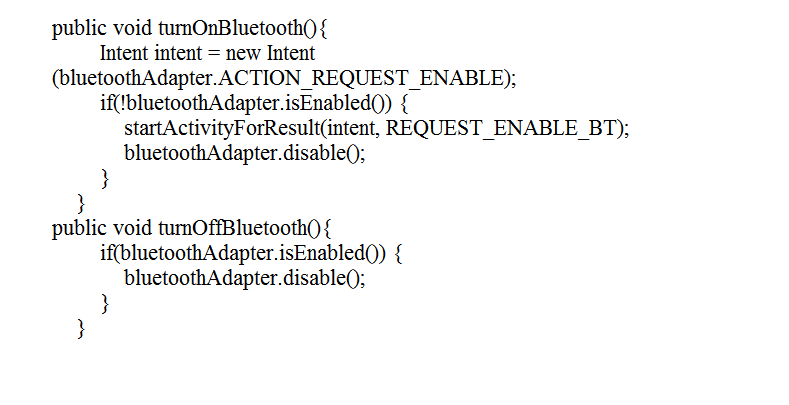
Рис.5. FileManagerActivity

4.3. Реализация поддержки Bluetooth

После выбора всех необходимых файлов и нажатия кнопки «Transfer» пользователь должен выбрать устройство для сопряжения.

**Включение/выключение Bluetooth адаптера**

Для управлением Bluetooth адаптером реализованы две функции turnOnBluetooth и turnOffBluetooth. Эти функции использую поле класса типа BluetoothAdapter. Функция включения запускает новую активность, в которой пользователю предлагается включить адаптер.



Результат работы функции turnOnBluetooth изображен на рисунке 6.

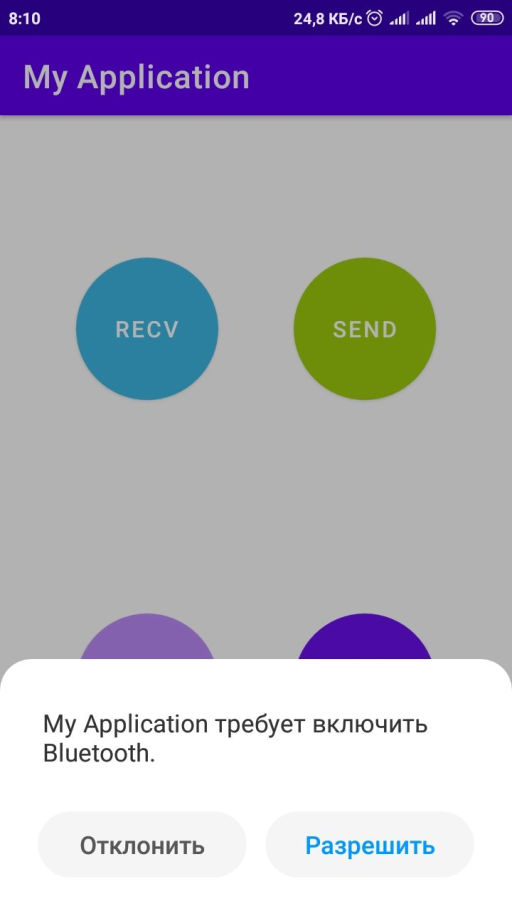
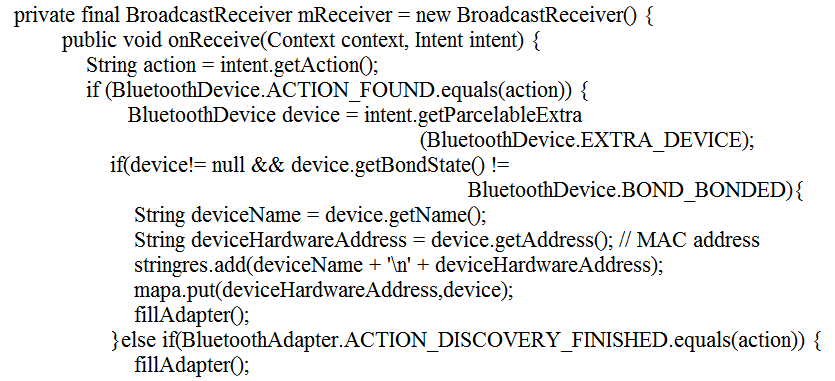
****

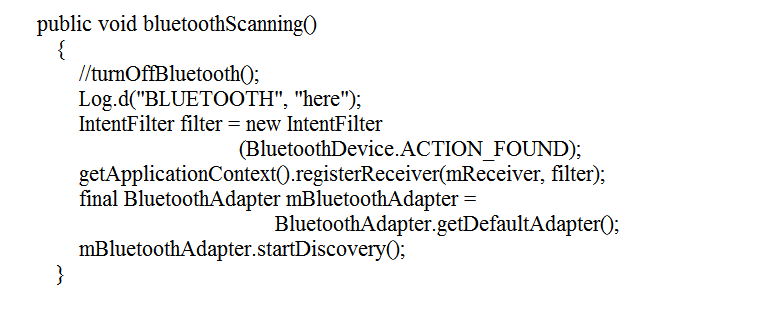
Рис. 6. Включение Bluetooth.

**Сканирование доступных устройств**

Для того, чтобы получить список всех устройств, нам необходим широковещательный приемник, который нужно инициализировать реализуя метод onReceive.



Функция bluetoothScanning выполняет всю оставшуюся работу по сканированию устройств.



Список всех доступных устройств изображен на рисунке 7.

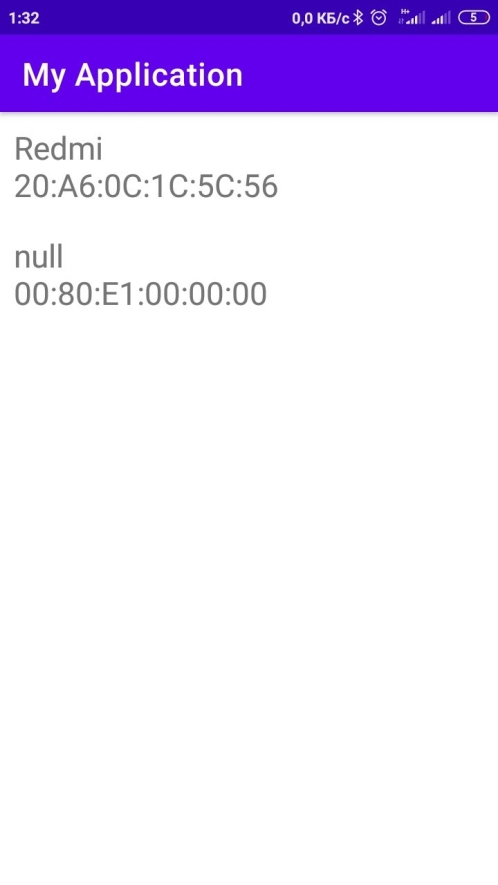


Рис. 7. Список доступных устройств.

**Сопряжение устройств и передача WiFi пароля**

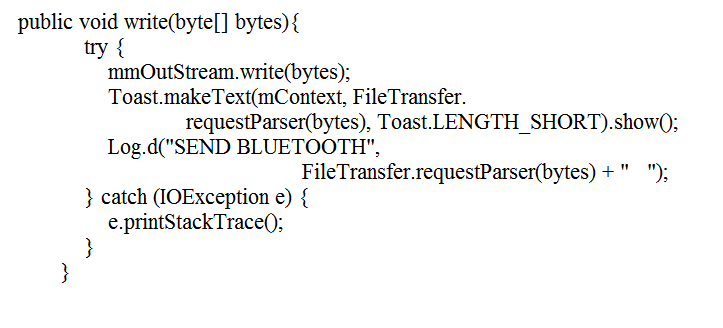
Для сопряжения двух устройств по Bluetooth был написан класс BluetoothConnectivity. Этот класс содержит в себе реализацию нескольких внутренних вспомогательных классов:

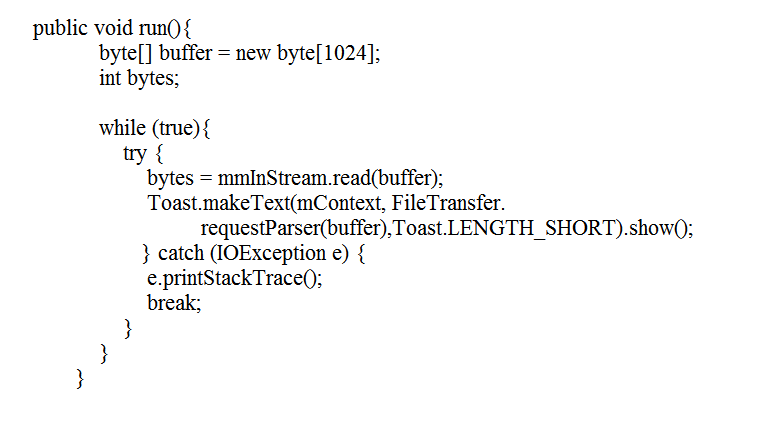
1) AcceptThread – класс для приема соединений.

2) ConnectThread – класс для подключения.

3) ConnectedThread – класс для передачи данных между подключенными устройствами.

Каждый из вышеописанных классов наследуется от класса Thread, соответственно было необходимо реализовать методы run и cancel. Для отправки SSID и пароля был создана функция write в классе ConnectedThread. Прием данных был осуществлен в методе run.

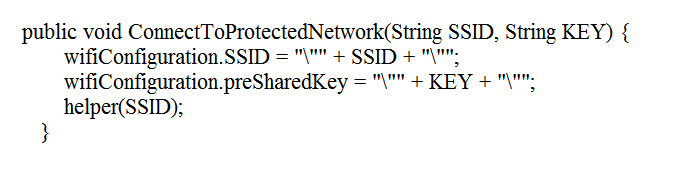




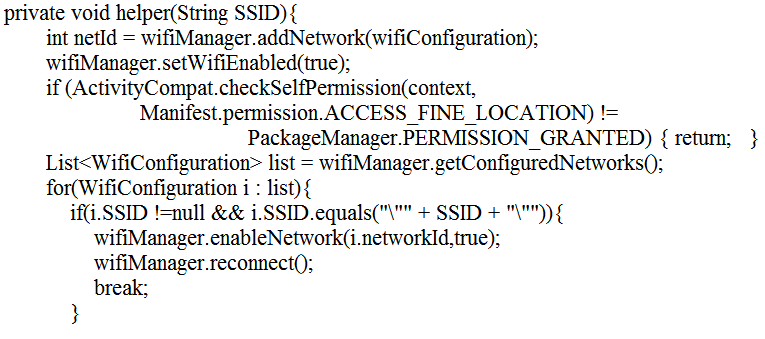
4.4. Реализация поддержки WiFi

**Соединение с указанной точкой доступа**

После передачи SSID и пароля устройству клиента, мы можем выполнить подключение к точке доступа по полученным данным. Для этого был реализован класс WifiConnect. Он имеет несколько методов для подключения к указанной сети. Метод ConnectToProtectedNetwork принимает SSID и пароль, заполняет конфигурацию и вызывает вспомогательную функцию helper.



Метод helper добавляет указанную конфигурацию, включает WiFi и подключается к сети.



4.5. Передача файлов по сети

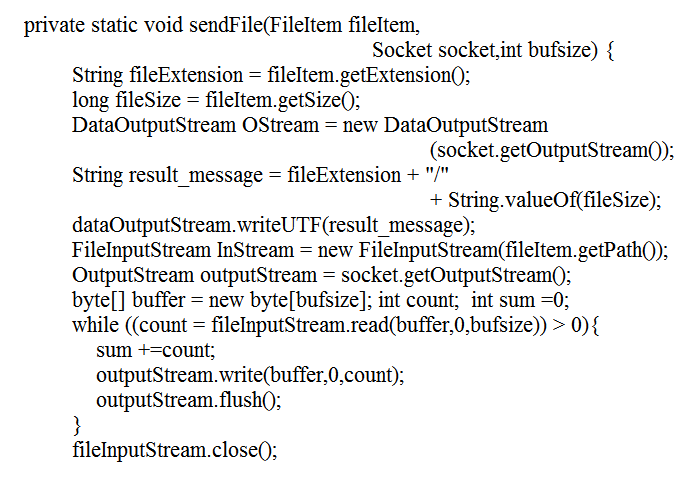
Для передачи файлов был написан класс FileTransfer. Все члены этого класса были сделаны статическими, так как нет необходимости создавать объект этого класса при отправке. В таблице 1 приведены соглашения для передачи

Таблица 1.

|  |  |
| --- | --- |
| Порт для прослушивания соединений  (TRANSFER\_PORT) | 9999 |
| Размер буфера  (bufsize) | 8192байт |

**Передача одного файла**

Для передачи одного файла на устройство был реализован метод sendFile.



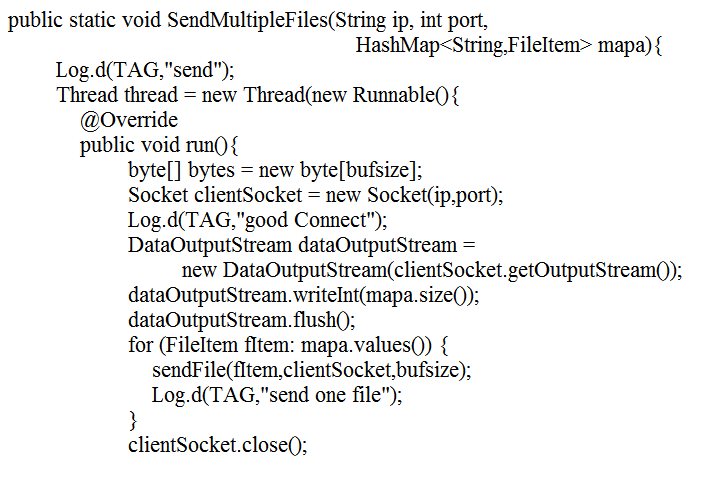
**Прием одного файла**

Прием одного файла осуществляется в методе recvFile.



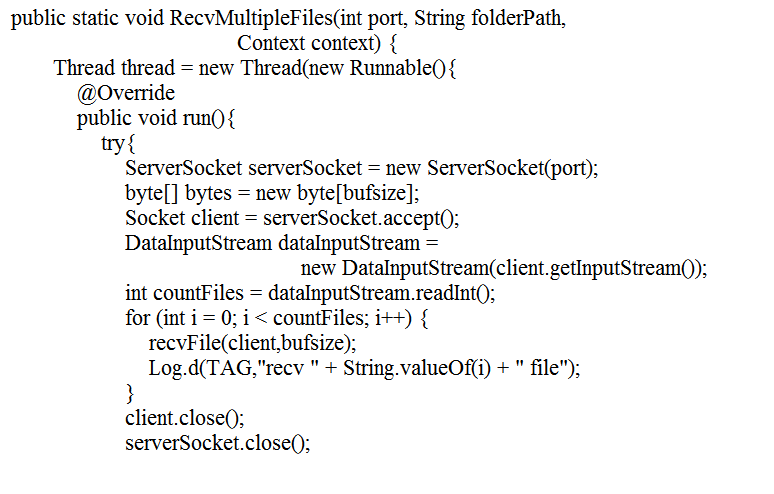
**Отправка нескольких файлов**

Для отправки всех выбранных файлов был реализован метод SendMultipleFiles. Так как запись в сокет является блокирующей операцией, было решено запускать цикл отправки в отдельном потоке. Функция SendMultipleFiles принимает Ip адрес, порт и хэш-таблицу с выбранными файлами.



**Прием файлов на устройстве**

Прием файлов осуществляется в методе RecvMultipleFiles. Этот метод принимает номер порта и директорию для сохранения полученных файлов. Эта функция, также как и в случае с отправкой, выполняет все блокирующие операции в отдельном потоке.



Результат работы функции RecvMultipleFiles изображен на рисунке 8.

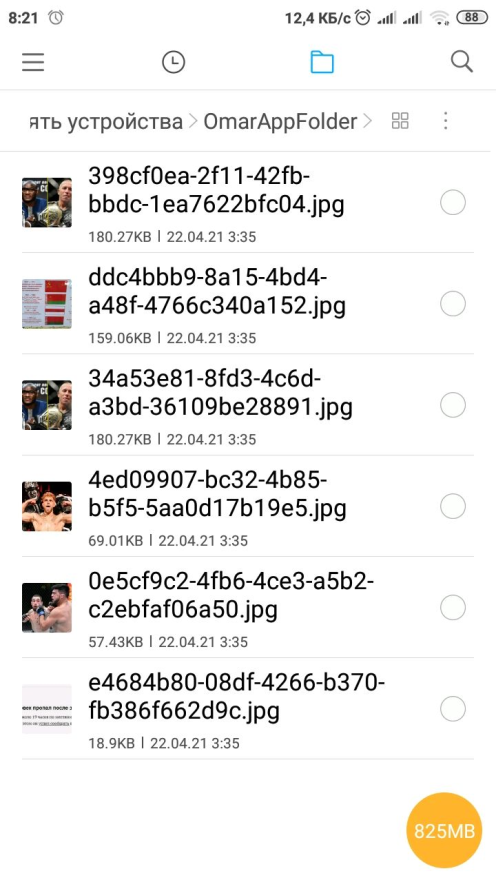


Рис. 8. Результат приема файлов от отправителя.

В результате была выполнена реализация всех компонентов системы в соответствии со всеми требованиями к системе.

ЗАКЛЮЧЕНИЕ

В рамках работы было разработано приложение для Android. Для достижения данной цели были решены следующие задачи:

1) осуществлена постановку задачи, выделены требования к приложению;

2) произведен обзор существующих решений для передачи данных;

3) изучены современные средства разработки приложений для Android;

4) определены требования и спроектировано программное обеспечение;

5) реализовано приложение.

Все поставленные задачи были решены, цель достигнута.

ЛИТЕРАТУРА

1. Android Studio Features. [Электронный ресурс] URL: https://developer.android.com/studio/features.html.

2. Appery.io: Enterprise Mobile App Builder &MBaaS. [Электронный ресурс] URL: https://appery.io/.

3. Download Android Studio and SDK Tools | Android Studio. [Электронный ресурс] URL: https://developer.android.com/studio/index.html.

4. Eclipse - The Eclipse Foundation open source community website. [Электронныйресурс] URL: https://www.eclipse.org/downloads.

5. Fork of Tesseract Tools for Android. [Электронный ресурс] URL: https://github.com/rmtheis/tess-two.

6. IntelliJ IDEA the Java IDE – JetBrains. [Электронный ресурс] URL: https://www.jetbrains.com/idea/.

7. Introduction to Material Design. [Электронныйресурс] URL: https://material.google.com/.

8. Kaner, Falk, Nguyen. Testing Computer Software. – USA: Wiley Computer Publishing, 1999.

9. MVP and MVC Architectures in Android. [Электронный ресурс] URL: https://www.techyourchance.com/mvp-mvc-android-1/.

10. Shoutem - Make an App - Build Apps with Easy Application Creator. [Электронный ресурс] URL: www.shoutem.com/.

11. Арлоу Дж., Нейштадт А. UML 2 и Унифицированный процесс. Практический объектно-ориентированный анализ и проектирование. 2-е издание. – М.: Издательство «Символ-Плюс», 2007. – 624 с.

12. Архитектура Android-приложений. Часть II – архитектурные стили и шаблоны. [Электронный ресурс] URL: https://habrahabr.ru/post/140655/.

13. Бурнет Э. Привет, Android! Разработка мобильных приложений. 2-е издание. – СПб.: Издательство «Питер», 2012. – 256 с.

14. Буч Г., Рамбо Дж., Джекобсон А. Язык UML. Руководство пользователя. – СПб.: Издательство «Питер», 2003. – 432 с.

15. ВЕДОМОСТИ – Интернет-аудитория России растет за счет мобильных устройств. [Электронный ресурс] URL: https://www.vedomosti.ru/technology/articles/2016/01/28/625779-internet-auditoriya-rossiirastet-schet-mobilnih-ustroistv/.

16. Дейтел П., Дейтел Х., Уолд А. Android для разработчиков. 3-е издание. – СПб.: Издательство «Питер», 2016. – 512 с.

17. НильсенЯ., БудиуР. MobileUsability. Как создавать идеально удобные приложения для мобильных устройств. – М.: Эксмо, 2013. – 256 с.

18. Операции. [Электронный ресурс] URL: https://developer.android.com/guide/components/activities.html

19. Основы создания приложений. [Электронный ресурс] URL: https://developer.android.com/guide/components/fundamentals.html.

20. Дэвис, Джозеф Microsoft Windows Server 2003. Протоколы и службы TCP/IP. Техническое руководство / Джозеф Дэвис , Томас Ли. - М.: Эком, 2019. - 752 c.

21. Паркер TCP/IP. Для профессионалов / Паркер, Сиян Тим; , К.. - М.: СПб: Питер; Издание 3-е, 2021. - 859 c.

22. Фейт, С. TCP / IP. Архитектура. Протоколы. Реализация / С. Фейт. - Москва: ИЛ, 2019. - 424 c.

23. Фейт, С. TCP/IP. Архитектура, протоколы, реализация (включая IPv6 и IP Security) / С. Фейт. - М.: ЛОРИ, 2018. - 554 c.

24. Хант, К. TCP/IP. Сетевое администрирование / К. Хант. - М.: СПб: Символ-Плюс; Издание 3-е, 2018. - 816 c.

25. Хант, К. TCP/IP. Сетевое администрирование / К. Хант. - М.: Символ-плюс, 2018. - 693 c.

26. Бертсекас Д., Галлагер Р. Сети передачи данных. - М., Мир, 1989. - 542с. .

27. Блэк Ю. Сети ЭВМ: протоколы, стандарты, интерфейсы. - М., Мир, 1990. .

28. Бройдо В.Л. Вычислительные системы, сети и телекоммуникации. - Спб., Питер, 2002. - 688с. .

29. Вендров А.М. Проектирование программного обеспечения экономических информационных систем, - М.: Финансы и статистика, 2002. .

30. Гук М. Аппаратные средства локальных сетей. СПБ., Питер, 2000. - 576с. .

31. М. Гук. Аппаратные средства PC. Энциклопедия. - М. 2003. .