ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ БЮДЖЕТНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «ФИНАНСОВЫЙ УНИВЕРСИТЕТ ПРИ ПРАВИТЕЛЬСТВЕ РОССИЙСКОЙ ФЕДЕРАЦИИ»

Кафедра «Информационных технологий и анализа больших данных»

Курсовая работа по дисциплине Технологии разработки приложений для мобильных устройств на тему «Разработка мобильного приложения для измерения силы удара (с тренажером)»

Студентки группы ПИ20-2

Факультета Прикладной информатики

Зорихиной Майи Витальевны

(подпись)

Научный руководитель: Клочков Е.Ю.

(подпись)

Москва 2022

СОДЕРЖАНИЕ

|  |  |
| --- | --- |
|  |  |
| ВВЕДЕНИЕ………………………………………………………………………….. | 3 |
| ГЛАВА 1. ТЕОРЕТИЧЕСКАЯ ЧАСТЬ……………………………………………. | 6 |
| ГЛАВА 2. ПРАКТИЧЕСКАЯ ЧАСТЬ…………………………………………….. | 15 |
| ЗАКЛЮЧЕНИЕ……………………………………………………………………... | 23 |
| СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ И ИНТЕРНЕТ-РЕСУРСОВ.. | 25 |

ВВЕДЕНИЕ

Можно с уверенностью сказать, что сегодня сфера физического здоровья стала крайне популярна и необходима человечеству. Практически каждый следил за своим весом, нагрузками и параметрами тела, записывались в спортзал, подбирали для себя подходящие тренировки, корректировали питание и т.д. Эта тема стала культовой по всему миру, ведь то, как выглядит человек – здоровым или нет – отражается на личной жизни и карьере. Многим хочется выглядеть стройными и подтянутыми, и для этого необходимы занятия спортом и правильное питание.

Что касается питания, то любой интересующийся отслеживанием своего рациона будет поражен количеством и разнообразием приложений для данных нужд. Подсчет калорий, возможность вносить свои приемы пищи, трекер выпитой воды, количество употребленных жиров, белков и углеводов, количество потраченных калорий, а также специально подобранные планы питания и диеты - все это можно найти в обычном приложении для питания. Сегодня стало крайне удобно следить за своей диетой, достаточно ввести рост, возраст, текущий и желаемый вес, и приложение само вам скажет сколько и что вам нужно съесть, чтобы похудеть или набрать вес, а трекер воды не даст вам быть обезвоженным. Он так же рассчитает необходимое вам количество воды, которое нужно выпить за день. Многие приложения имеют функцию подпора персональных диет, в которых описывается польза определенных продуктов, какие витамины в них содержатся и т.д. Поэтому в наши дни стало намного проще отслеживать свой рацион, питаться правильно и разнообразно. Лично из моего опыта хочу отметить трекер воды, так как я всегда забывала пить воду в течение дня, но приложения всегда высылало мне уведомления-напоминания о воде, и теперь я считаю эту функцию крайне полезной.

Кроме того, есть люди, у которых занятия спортом и отслеживание питания переросло в профессию, которой они отдаются каждый день. Фитнес-тренеры помогают другим правильно использовать те или иные тренажеры, следить за своим дыханием и частотой сердечных ударов, планируют тренировки специально для каждого человека, учитывая особенности тела.

Слово фитнес появилось от английского слова "to fit" и переводится как "пребывать в хорошей форме". Эта тенденция физического развития была создана в XX веке, а сейчас является одной из самых развивающихся.

Однако кто-то предпочитает сделать из этого карьеру и самосовершенствоваться в спорте, чтобы участвовать в соревнованиях. Для таких людей отслеживание своего результата является наиважнейшей функцией, чтобы видеть рост или спад прогресса. Например, для боксера важными параметрами является сила и техника удара.

Прогресс в сфере мобильных технологий перевернул наши взгляды на то, как мы следим за нашим здоровьем, чтобы оставаться активными. Сегодня большинство людей имеют смартфоны, поэтому возросло число тех, кто использует фитнес-трекеры для тренировок. На рынке электроники можно найти бесчисленное количество фитнес-браслетов, смарт-часов и т.д., которые обладают важным и полезным функционалом как для начинающих, так и для профессиональных спортсменов. К примеру, только благодаря измерению пульса с помощью специальных светодиодов стало возможным отслеживать частоту сердечных ударов в реальном времени. Если пользователь видит, что пульс слишком высокий для его массы тела, он должен замедлить темп или отдохнуть. Таким образом, данная технология позволяет не только отслеживать пульс, но и записать кардиограмму, и если часы обнаружат аритмию, они предупредят человека.

Рассматривая данную тему с точки зрения информатики и программирования, она затрагивает вопрос совместимости между физическим прибором и мобильным приложением, что также является актуальным в наши дни. Научная новизна курсовой работы состоит в том, что в разработке мобильных приложений для измерения каких-либо параметров с помощью различных приспособлений стало возможным собирать и анализировать данные о физических характеристиках человека, такие как частота сердечных ударов, скорость ходьбы, качество сна, активность в течении дня и т.д.

Тема моей курсовой работы является актуальной по многим причинам: в первую очередь, измеряемые данные важны для профессиональных спортсменов в сфере бокса для самосовершенствования, также и простым людям будет интересно узнать свою силу удара или тем, кто следит за состоянием своего здоровья и не хочет терять силу. Для этого будет необходимо мобильное приложение, которое предоставит эти данные пользователю. Объектом данной курсовой работы является разработка приложения для мобильных устройств. Предметом исследования является создание приложения, соединение, сбор и анализ данных о силе удара с помощью смартфона и тренажера. В своей работе я попытаюсь объединить мобильное приложение и тренажер для измерения силы удара.

ГЛАВА 1. ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

На сегодняшний день платформа android является наиболее популярной операционной системой. На статистике, она установлена у 86% всех проданных смартфонов в мире. На OS Android сегодня работают не только мобильные и планшетные устройства, а также всевозможные часы, телевизоры и некоторые компьютеры Не смотря на огромное множество устройств вы можете практически под все из них программировать через одну программу – Android Studio, которая является наиболее популярной средой для разработки приложений для android. Конечно, для каждой платформы будут свои особенности – размер экрана, характеристики устройства и т.д. Тем не менее общий процесс создания будет схожим. Данная программа оснащена визуальным и текстовым редактором. Внутри визуального редактора мы может прописывать весь дизайн программ – задавать кнопки, всплывающие окна, текстовые поля и много другое. Текстовый редактор отвечает за создание функционала внутри программы.

Разработка для операционной системы android позволяет превратить ваши идеи в реальность. Для создания приложения для измерения силы удара будет использована среда разработки Android Studio.

В проекте файлы группируются по определенным признакам. Папка .gradle служебная, автоматически генерируется, используется менеджером зависимости gradle. Например, мы указываем для какой версии андроида мы создаем приложение, какие дополнительные библиотеки подключаем и т.д. Она содержит всевозможные файлы, которые относятся к Android Studio. Папка app - папка модуля, в которой содержатся все файлы или код, который присутствует в нашем приложении. Приложение может состоять из одного модуля или из нескольких. В ней папка build автоматически генерируется, там находятся промежуточные файлы, создается apk файл, который содержит приложение и который устанавливается на устройство. Различные библиотеки помещаются в папку libs. Src – главная папка, в которой находятся весь код и все остальные файлы проекта. Папка main содержит основные составляющие приложения – папка java. Ней в главном пакете находится основной класс MainActivity, с которой начинается выполнение программы.

Папка res является крайне важной частью проекта, в ней хранятся различные ресурсы: drawable – изображения, layout – activity, которые отвечают за расположение элементов; mipmap – иконки для приложения, values – различные константы для цветов, строк и стилей.

В папке manifests находится файл AndroidManifest.xml. Данный файл представляет собой список основной информации о самом приложении, может содержать большое количество настроек. В начале находится корневой элемент manifest, который содержит xmlns и название основного пакета. Элемент application содержит информацию о самом приложении. AllowBackup - говорит о том, нужно ли нам при резервировании данных самого андроида сохранять данные самого приложения. Далее icon – иконка, которая отображается на экране смартфона, в панели уведомлений. Указывается путь к ресурсу. Label – название приложения, которое будет отображаться, например6 на рабочем столе или в меню приложений. Theme – ссылка на тему (набор стилей), которая будет устанавливаться по умолчанию. Далее идет список activity. Для каждой activity будет свой элемент в application. Activity имеет имя и intent-filter. Часто AndroidManifest используется для запроса доступа к различным разрешениям, например, получить доступ на чтение смс, доступ к геолокации и т.д, а также могут добавляться различные сервисы.

Первая часть приложения – макет – описывает как приложение должно выглядеть, описывает структуру пользовательского интерфейса. Все элементы в макете построены с использованием иерархии, которая включается в себя view и ViewGroup. В отличие от ViewGroup, View обычно отображает то, что видит пользователь, в то время как ViewGroup является невидимым объектом, который определяет структуру макета для View и других ViewGroup. Объекты View, называемые виджетами, могут быть частью подклассов, таких как Button или TextView. В разрабатываемом приложении понадобятся такие компоненты как TextView и ListView. ListView представляет собой список определенных элементов, которые можно очень удобно прокручивать. Наполнением информацией будет заниматься специальный адаптер данных – ArrayAdapter. Адаптер – это часть программного обеспечения, с которой придется столкнуться каждый раз, когда нужно работать со списками или любыми раскрывающимся меню. ListView необходимо создать и заполнить данными. Для этого нужен список или коллекция, в которой будут храниться элементы. С помощью метода list.add() его можно заполнить. Для ArrayAdapter указывается context – this, ресурс – android.R.layout.simple\_list\_item\_1 – готовая системная разметка (какой тип адаптера мы собираемся построить) и list.Объекты ViewGroup обычно называются layout или слой, с помощью которых могут быть созданы LinearLayout, FrameLayout, TableLayout, RelativeLayout и ConstraintLayout. За счет ConstraintLayout мы можем размещать все объекты в любой точке рамки. В LinearLayout объекты могут быть расположены рядом друг с другом либо вертикально, либо горизонтально. FrameLayout позволяет добавить определенный фрейм. TableLayout создает таблицу объектов. RelativeLayout является довольно удобным слоем, если нам необходимо создать некое адаптивное приложение, в котором каждый элемент будет прикреплен к предыдущему элементу, у каждого элемента ширина и высота будет растягиваться в зависимости от того, на каком устройстве будет запущено приложение. Помимо данных слоев существуют и другие.

Используя XML словарь андроида мы можем быстро создавать layout’ы и элементы, которые они содержат. Данный процесс схож с созданием веб-страницы на HTML при создании ряда вложенных элементов. После того, как мы создали макет в XML, его необходимо сохранить с расширением .xml в каталоге res/layout в проекте, что бы он был правильно скомпилирован.

Когда мы компилируем приложение, каждый xml файл компилируется в View ресурс.

Activity – страница в приложении. Их может быть любое количество, и у каждой страницы будут свои определенные состояния. При открытии MainActivity можно увидеть подключение activity в setContentView(), описание самого класса и метод onCreate(). Данный метод описывает определенное состояние, он будет срабатывать, когда activity запускается т.е. весь код, который записан в методе onCreate() будет работать в момент запуска страницы. Помимо onCreate() существуют другие состояния, которые вызываются в определенные моменты жизни самой activity.

Для примера можно взять любую activity т.е. любую страницу в приложении. Для каждой страницы будут срабатывать определенные события. Как только activity запускается, всегда будет срабатывать onCreate(). После этого метода будет запущен onStart(), в котором происходит запуск activity. Далее срабатывает метод onResume() – процесс отображения activity пользователю. После этих шагов activity будет запущена, и пользователь сможет с ней работать. В случае, если будет открыта верхняя шторка уведомлений, страница встанет на паузу – вызов метода onPause(). Это необходимо, когда, например, пользователь вышел из игры, и требуется сохранить счет или поставить игру на паузу, или выключать фонарик при выходе из приложения. Чтобы возобновить весь процесс работы, когда пользователь вернулся в приложение, нужно обратиться к onResume(). Если он не вернулся, будет вызван метод onStop(), когда приложение полностью закрыто – onDestroy(). На рис. 1 изображен жизненный цикл приложения.

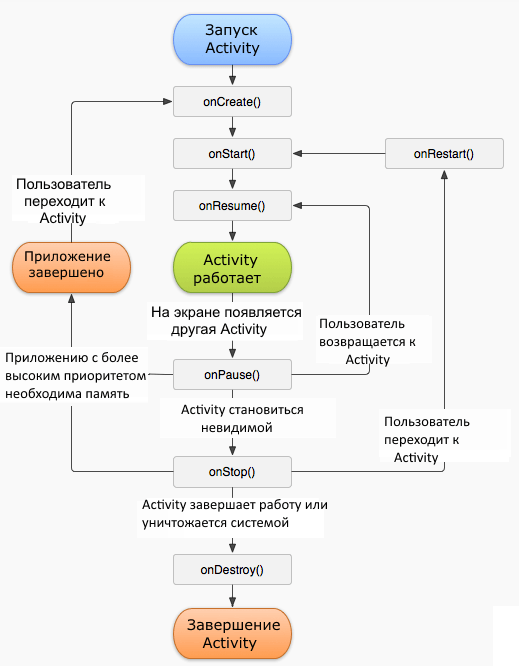


Рис. 1 – жизненный цикл приложения.

Чтобы разработать функционал приложения, нужно перейти в MainActivity. В первую очередь, необходимо создать переменные, которые будут ссылаться на определенный объект или виджет из дизайна. При создании переменной указывается ее тип. Чтобы узнать, какой класс нам надо использовать при объявлении переменной, нужно посмотреть, как называется объект. Например, при создании переменной для объекта TextView мы обратимся к классу TextView и назовем переменную, при этом классы импортируются в начале документа. Названия могут отличаться от идентификаторов в activity. В методе onCreate() можно присвоить созданным переменным какие-либо значения с помощью findViewById() – находится объект по идентификатору, при этом мы обращаемся к классу R. Класс R работает со всей папкой res, за счет него существует возможность находить идентификаторы объектов, файлы с дизайном, картинки, цвета и т.д. Чтобы добавить обработчик события, нужно обратиться к setOnClickListener(), с помощью которого мы создадим метод – что будет срабатывать при нажатии на кнопку. Внутри него помещаем создание класса, автоматически создается onClick(). Весь код внутри данного метода будет срабатывать при нажатии на кнопку. Метод setText() позволяет установить определенный текст.

Чтобы создать всплывающее окно, нужно обратиться к встроенному классу Toast и импортировать его в начале файла. Через него можно к методу makeText(), он принимает несколько параметров: контекст (тот класс, внутри которого мы показываем всплывающее окно), текст, длительность показа (LENGTH\_SHORT – будет показано на секунду, LENGTH\_LONG – 3 или 4 секунды). Далее необходимо указать метод show(), который позволяет указать само всплывающее окно. В случае, если мы хотим передать тот текст, что на кнопке, необходимо обратиться к параметру v в методе showText().

Платформа Android включает в себя возможность поддержки сетевого стека Bluetooth, который позволяет устройствам иметь беспроводную связь и обмениваться данными с другими устройствами. Чтобы устройства с поддержкой Bluetooth могли передавать данные друг от друга, они должны сначала сформировать канал связи с помощью процесса сопряжения. Одно обнаруживаемое устройство становится доступным для входящих запросов на подключение. Другое устройство находит это обнаруживаемое устройство с помощью процесса нахождения службы. После того, как устройство примет запрос на подключение, два смартфона завершают процесс соединения, в котором они обмениваются ключами безопасности, после чего обмениваются информацией. Для разработки приложения для измерения силы удара понадобится информация о работе с Bluetooth, чтобы приложение могло получать данные от тренажера. Для начала необходимо добавить разрешение user-permission в файле AndroidManifest.xml. Чтобы получить доступ к Bluetooth нам нужен BluetoothAdapter – это класс, который отвечает за доступ к нему. Также должна быть проверка, подключено ли устройство или нет.

Всем известно, что ресурсы ограничены, и их нужно использовать с умом. Процессор и память в смартфоне – это главные помощники в увеличении производительности. За эффективность процессора отвечает многопоточность. Процесс – закрытая среда выполнения, которая имеет полный набор runtime ресурсов и также имеет свою область памяти. Процесс может состоять из одного или нескольких потоков. Потоки могут существовать только в рамках какого-либо процесса.

Приложение это процесс, оно состоит из нескольких компонентов: Activity, Service, BroadcastReciever и ContentProvider. Если один из компонентов приложения запускается, и операционная система видит, что это первый компонент, то под него создастся новый процесс с потоком. Такой поток обычно называют главным или UI Thread и в нем выполняются базовые компоненты приложения. При желании можно изменить процесс в файле AndroidManifest в атрибуте android:process. Если какой-то другой компонент уже был запущен в системе, то он будет присоединен к процессу, который был создан для первого компонента.

В андроид приложении может быть несколько потоков, которые будут выполняться одновременно, при этом операционная система либо распределит различные потоки на различные ядра, либо будет выполнять потоки на одном ядре, при этом одно ядро будет быстро перемещаться между разными потоками, и будет создаваться видимость, что потоки выполняются одновременно. В любом android приложении есть главный поток, который ответственен за прорисовку экранов, переходов, анимации, получении данных от пользователя, нажатия на кнопку и т.д. То есть главный поток ответственен за основные функции в приложении и этот поток запускается при запуске приложения. Чтобы приложение работало плавно и не замирало мы должны выносить всю ресурсоемкую или долгую работу в отдельные потоки и не нагружать главный поток.

В андроиде для того, чтобы реализовать многопоточные программы существует специальный паттерн или абстракция, которая называется AsyncTask. Она необходима для того, что делать ресурсоемкую работу в отдельном потоке и после этого обновлять пользовательский интерфейс в главном потоке. При создании необходимо указать три параметра: Params(тип входных данных параметров для выполняемого процесса), Progress (тип данных, возвращаемых в ходе работы процесса), Result (тип данных, возвращаемых процессом). Можно переопределить следующие методы: doInBackground, onProgressUpdate, onPostExecute, onPreExecute. В doInBackground описывается главная логика того, что мы хотим делать в отдельном потоке. OnProgressUpdate – то, что мы хотим выполнять в момент выполнения процесса. OnPostExecute – то, что будет выполняться после совершения процесса. OnPreExecute – выполняется самым первым, например, инициализация переменных. На рис. 2 изображен принцип работы AsyncTask. Главный поток изображен зеленой стрелкой – Main/UI thread. Процесс, который выполняется на заднем плане, в отдельном потоке, указан синей стрелкой. Когда мы запускаем AsyncTask, появляется второй поток. Изначально выполняется onPreExecute(), в нем обычно инициализируются переменные, производится какая-либо начальная конфигурация. Затем выполняется метод doInBackground() в отдельном потоке, и во время его выполнения мы можем отправлять обратно в пользовательский интерфейс прогресс, например, строку загрузки, с помощью publishProgress. После того, как процесс в doInBackground() выполнится, запустится onPostExecute(), который, например, может закрывать соединения, чистить ресурсы и т.д.

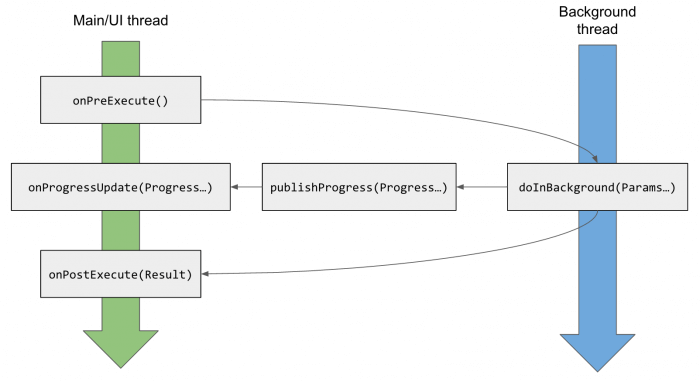


Рис. 2 - принцип работы AsyncTask.

Интерфейс Bluetooth сокетов схож с TCP сокетами – Socket и ServerSocket. Со стороны сервера используется BluetoothServerSocket чтобы создать прослушивание серверного сокета. Когда соединение принимается BluetoothServerSocket, оно возвращает новый BluetoothSocket чтобы управлять соединением. Со стороны клиента используется один BluetoothSocket чтобы инициировать входящее и исходящее соединение, а также управлять им. Наиболее часто используемым Bluetooth сокетом является RFCOMM. RFCOMM - это ориентированный на соединение потоковый транспорт через Bluetooth. Он также известен как Serial Port Profile (SPP). Чтобы создать BluetoothSocket для соединения с известным устройством необходимо использовать BluetoothDevice.createRfcommSocketToServiceRecord(). Затем вызывается connect() чтобы попытаться подключиться к удаленному устройству. Этот вызов будет блокироваться до тех пор, пока соединение не будет установлено или соединение не выйдет из строя.

При написании приложения понадобится Handler – это механизм последовательного принятия и обработки сообщений, т.к. может возникать проблема многопоточности, ведь основной поток MainThread отвечает только за отрисовку экрана и View. В android к потоку может быть привязана очередь сообщений. Мы можем помещать туда сообщения, а система будет следить за очередью и отправлять сообщения на обработку, при этом мы можем указать, чтобы сообщение ушло на обработку не сразу, а спустя определенное количество времени. Handler – это механизм, который позволяет работать с очередью сообщений. Он привязан к одному потоку и работает с его очередью. Задача Handler – создавать и отправлять сообщения, а также опционально обрабатывать их. Он дает нам несколько возможностей: реализовать отложенное по времени выполнение кода и выполнение кода не в своем потоке.

Также в приложении будет необходим хронометр – это управляемый таймер, в котором есть три основных метода – start, stop и setFormat.

ГЛАВА 2. ПРАКТИЧЕСКАЯ ЧАСТЬ

Главная задача приложения для измерения силы удара – получить данные от тренажера, обработать их и в качестве результата вывести значение. В моем проекте содержатся такие файлы: PowerActivity, SettingsActivity в папке java; activity\_settings, activity\_power в папке res/layout. В файле PowerActivity будут происходить основные процессы, связанные с приемом данных и расчётом. SettingsActivity предназначен для соединения смартфона к тренажеру с помощью Bluetooth и выводу списка подключенных устройств. Activity\_power – главная страница приложения, которую видит пользователь, на ней выводятся значения силы удара, скорости удара, реакции и таймера. Activity\_settings представляет собой экран, на котором изображает список подключенных устройств.

В SettingsActivity в методе onCreate() мы связываем данный файл с файлом activity\_settings. Activity\_settings представляет собой экран, на котором расположены два LinearLayout и кнопка Button. Первый LinearLayout использован в качестве заголовка, он говорит нам подключиться к нужному устройству, во втором находится ListView, в котором будут показаны устройства, доступные для подключения. Через ListView также будет реализована возможность подключения к устройству через нажатие по объекту. С помощью кнопки button будет запускаться поиск устройств и добавление их в список для адаптера, который выводит ListView. В SettingsActivity объявляем переменные: Button btnPaired – кнопка для поиска устройств, ListView devicelist – список доступных устройств, Animation anim – анимация при нажатии на кнопку, TextView txtViewPairDev – текст над списком. Для того, чтобы использовать функционал Bluetooth в Android Studio, прежде всего необходимо его импортировать. Методы использования Bluetooth имеют класс BluetoothAdapter, который предназначен для соединения с устройством. Чтобы установить данный класс, необходимо прописать следующий код в методе onCreate(): myBluetooth = BluetoothAdapter.getDefaultAdapter(). В случае, если то устройство, которое мы ходим присоединить, не поддерживает Bluetooth, то BluetoothAdapter будет принимать значение null. Далее будет происходить проверка, можно ли работать с подключенным устройством, т.е. равен ли BluetoothAdapter null или нет. Если равен – выводим сообщение «устройство недоступно». Иначе проверяем включен ли Bluetooth на устройстве. Чтобы включить Bluetooth, посылаем интент – запрос на подключение BluetoothAdapter.ACTION\_REQUEST\_ENABLE. Далее передаем созданный интент startActivityForResult. Обычно для подключения к устройству мы заходим в настройки смартфона и ищем нужное устройство. Добавим функцию поиска устройств сразу из приложения. Для этого необходимо добавить специальное разрешение ACCESS\_FINE\_LOCATION в манифесте. Оно связано с GPS модулем, так как Bluetooth и GPS модули связаны. Следующим шагом будет обработка нажатия кнопки btnPaired, добавление анимации при нажатии. Цель данной кнопки – вывести список, который содержит доступные для подключения устройства, чтобы подключиться к тренажеру, при этом используется метод pairedDevicesList().

В pairedDevices помещаются набор объектов BluetoothDevice, которые связаны или связываются с локальным адаптером, а также создается пустой список. Происходит проверка есть ли устройства в pairedDevices, и, если они есть, с помощью цикла проверяем все устройства и в новый пустой список помещаются имя и адрес устройства, этот список будет передаваться в адаптер. Если доступных устройств нет - приложение выводит «no paired Bluetooth devices found».

Далее создается ArrayAdapter для работы с ListView, в котором будут содержаться элементы с именами и mac адресами доступных для подключения устройствами. Следующим шагом метод setAdapter связывает список devicelist с adapter. Вызывается метод myListClickListener когда пользователь нажимает на устройство из списка.

У ListView есть готовый метод setOnItelClickListener, который предназначен для обработки события, когда нажимается какой-либо элемент из списка. С помощью OnItemClick() программа может определить какой элемент списка был нажат. В info получаем текст из нажатого TextView, а в address выбираем mac адрес, который составляет последние 17 символов в info. Затем создается интент для запуска следующей активности.

Далее переходим к главной активности - PowerActivity. В методе onCreate() связываем данный файл с activity\_power, который является главным экраном. На нем расположены надписи TextView «Сила удара», «Время удара» и «Таймер». Под каждой надписью находится соответствующее значение: txtPower, txtTimePower. Также необходимы кнопка btnStop и btnStart и таймер Chronometer. Для начала нам необходимо адаптировать приложение под размеры экрана, это возможно сделать с помощью метода display\_adapter. Сначала получаем размер экрана getWindowManager().getDefaultDisplay() и DisplayMetrics() – структура, описывающая общую информацию о экране, такую как размер, плотность и размер шрифта. Затем из этих данных получаем ширину и высоту экрана в пикселях, а также диагональ. Диагональ находится по формуле sqrt(abs(hight\_display\*hight\_display - width\_display\*width\_display)). Также необходимо выяснить плотность, используя densityDpi – плотность экрана, выраженная в точках на дюйм. High\_dens считается как высота дисплея деленное на плотность, также считается и width\_dens. В результате стало возможным посчитать разрешение - high\_dens\*width\_dens. Следующие действия метода предназначены для установления соответствующих размеров текста и картинок для определенного разрешения. Это сделано для того, чтобы на любом устройстве было корректно отображены элементы интерфейса приложения.

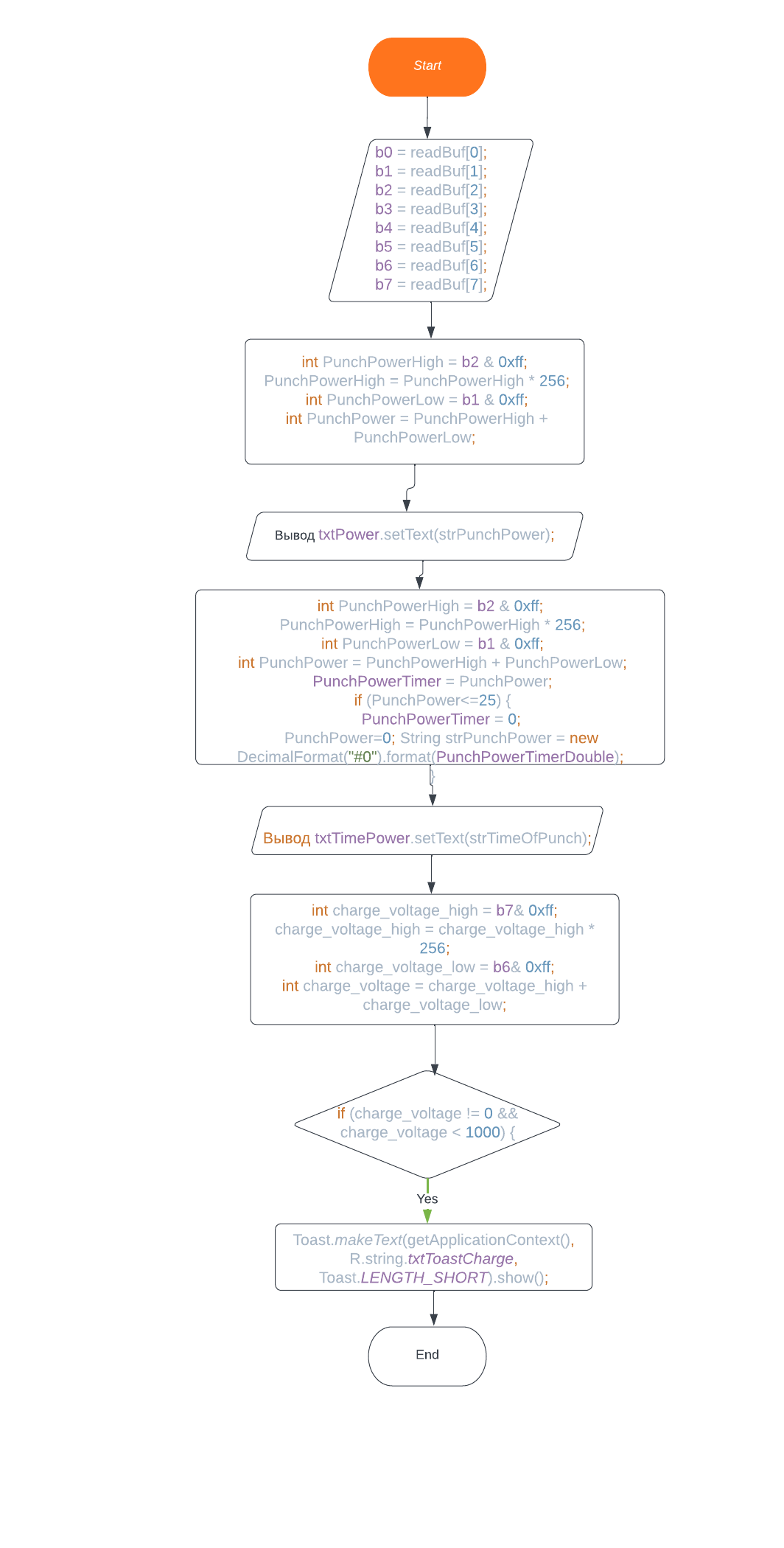
Следующей частью являются обработка нажатия кнопки btnStart. Когда пользовать нажимает на кнопку, btnStart становится невидимой, в то время как btnStop становится видимой. Необходимо использовать setBase(SystemClock.elapsedRealtime()) чтобы установить базовое время на текущее системное т.к. нам нужно, чтобы таймер начинал отсчет от 00:00. Затем следует сразу запустить таймер с помощью start(). Далее идет проверка существует ли соединение. Если нет – приложение выведет сообщение «активируйте соединение Bluetooth в настройках». Также инициализируются переменные, которые понадобятся в алгоритме расчета силы удара.

Чтобы остановить работу, необходимо обработать нажатие кнопки btnStop. В методе onClick() данная кнопка становится невидимой, в то время как btnStart возвращается обратно в видимый режим.

Соединение приложения и тренажера происходит с помощью ConnectBT() в методе onCreate(). Создадим переменные mDevice – Bluetooth устройство, которое мы получаем по mac адресу; myBluetooth – с помощью него будем устанавливать соединение. При разработке приложения понадобится AsyncTask, чтобы соединение с тренажером происходило в отдельном потоке. В параметрах будет void. В onPreExecute() помещаем переменную progress, которая показывает progress dialog – индикатор процесса. В то время как на экране показан индикатор процесса, соединение происходит на заднем плане. Это мы прописываем в методе doInBackground(). Если подключения нет, приложение получит доступ к Bluetooth и соединит с адресом устройства, а также проверит доступно ли оно. Инициализируем myBluetooth = BluetoothAdapter.getDefaultAdapter(), т.е. получаем Bluetooth адаптер устройства, с помощью которого мы сможем включать и выключать Bluetooth, производить соединение, передавать данные и т.д. mDevice = myBluetooth.getRemoteDevice(address) подключается к адресу устройства и проверяет его доступность. Также нам понадобится сокет, который будет налаживать соединение между устройством и приложением. Через данный сокет мы сможем открыть InputStream и OutputStream, т.е. передающий и принимающий потоки. Через них мы будет передавать байты. Используем специальный класс BluetoothSocket, который настраивает соединение и начинает принимать и передавать данные. Далее создается соединение RFCOMM (SPP) btSocket = mDevice.createInsecureRfcommSocketToServiceRecord(myUUID). UUID – специальный идентификатор приложения. BtSocket.connect() запускает соединение. mConnectedThread обращается к классу ConnectedThread, создается поток. mConnectedThread.start() запускает поток. В этом классе принимаются входящие и исходящие потоки tmpIn = socket.getInputStream() и tmpOut = socket.getOutputStream(). Учитываем ошибки, делаем попытку с помощью try-catch блок, что приложение не закрылось с ошибкой. В catch указываем IOException (Input/Output exception). Как только он запускается, в классе запускается функция run(). У потока Thread есть функция run(), в нем инициализируется байтовый массив(буфер) byte[] buffer = new byte[256]. Создаем цикл while (true), чтобы он постоянно запускался, и будет прерывать его, когда больше не сможем получать данные. В блоке try будем считывать данные и записывать их в массив из байтов buffer: bytes = mmInStream.read(buffer). Т.е. продолжая слушать входящий поток, получаем количество байт и само сообщение в байтовый массив. Если данные не считались, сорвалось соединение в случае, если отключили соединение с устройством или питание – указываем break. Далее отправляем в очередь сообщений Handler.

Создадим handler с помощью пустого конструктора, чтобы облегчить задачу. В handleMessage, его методе обработки, будем прописывать логику измерения силы и время удара, уровень зарядки. Происходит проверка: если приняли сообщение в handler, то принимаем байты и распределяем их по переменным. Контролер с датчиком давления передает в первых двух байтах давление, в следующих двух байтах – время, в последних – уровень заряда.

При работе с тренажером крайне полезной функцией будет считывание уровня заряда и, в случае если уровень заряда низкий, приложение отправит Toast уведомление с текстом «Низкий уровень заряда. Зарядите устройство». На рис. 3 представлена блок-схема, изображающая алгоритм считывания байтов и вывода



Осталось создать меню, в котором пользователь сможет либо отключить соединение с тренажером, и тогда приложение завершит свою работу, либо зайти в активность «Настройки», в которой он сможет увидеть все доступные Bluetooth устройства и выбрать необходимое. Метод onCreateOptionsMenu() отвечает за появление меню. Вызываем getMenuInflater().inflate(R.menu.menu\_power\_activity,menu), т.е. указываем какое меню нужно указать активности. С помощью метода onOptionsItemSelected() проверяем случаи какой элемент меню был выбран по id. На рис. 7 указана menu\_power\_activity.

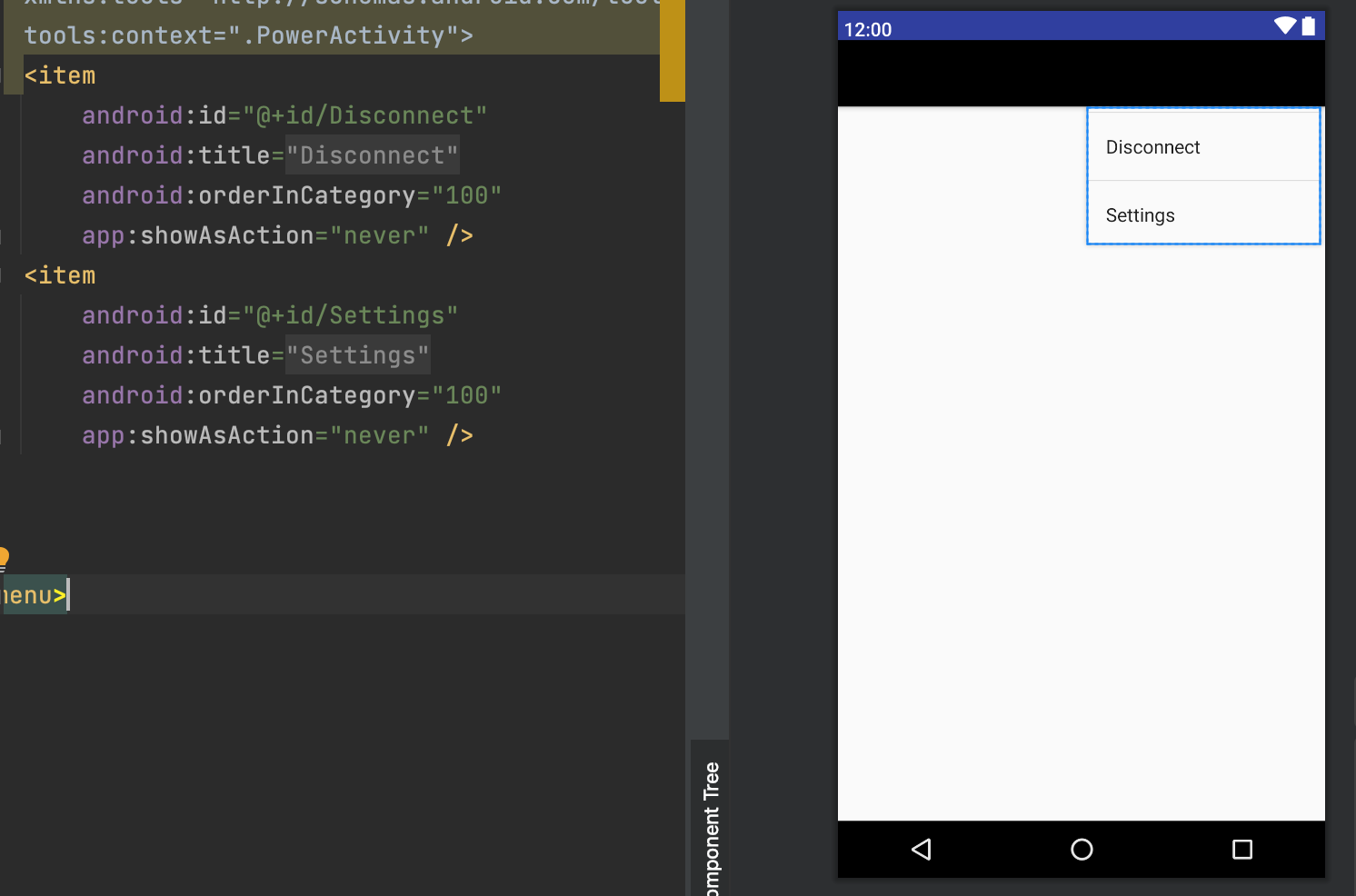


Рис. 7 – создание меню.

На рис. 8 представлен результат работы приложения, а также демонстрация списка, выводящего доступные для подключения устройства.

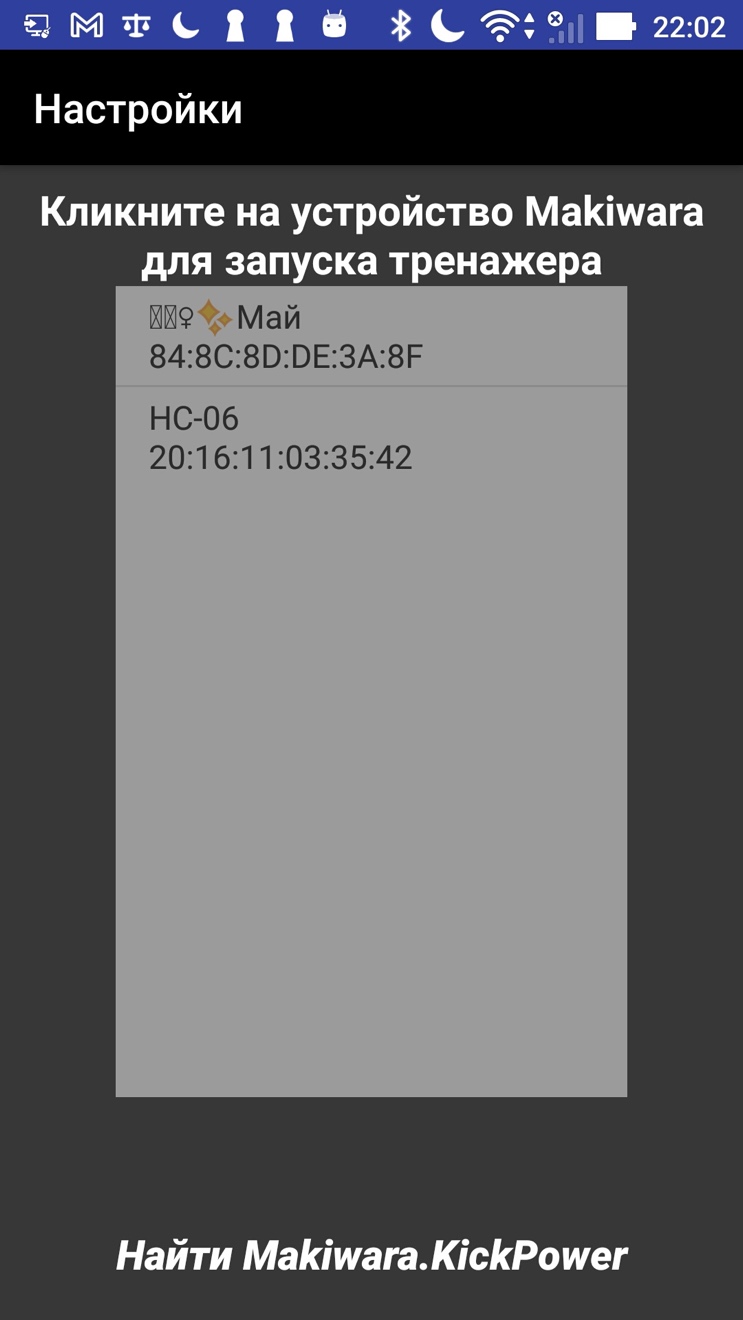


Рис. 8 – результат работы программы.

ЗАКЛЮЧЕНИЕ

В наши дни разработка мобильных приложений стремительно развивается, соприкасаясь со многими сферами жизни, такими как развлечения, организация порядка, информация, а также спорт. Ни для кого не секрет, что мобильные приложения во многом упрощают жизнь человека, они дают возможность более рационально распределять свое время и силы и быть более продуктивным. Целью моей курсовой работы было разработать мобильное приложение, которое бы измеряло силу удара по тренажеру. Для этого понадобилось изучить множество тем, которые позволили создать соединение между тренажером и приложением и обработать полученные данные. Такими темами являются Bluetooth, многопоточность в android, AsynсTask и др. Помимо этого, без базовых знаний написания мобильного приложения у меня бы не получилось достичь поставленных целей. Для начала было необходимо создать пустое приложение, которое было содержало активность и главную страницу. После этого его можно заполнять необходимыми для реализации процессами и методами.

Для создания соединения между тренажером и приложением потребовалось изучить существующие способы беспроводной передачи данных, в результате чего с помощью Bluetooth данная задача была решена. Эта технология позволяет создавать соединение практически с любым устройством, оснащенным специальными модулями и также она является легкой для понимания и реализации.

Также в условиях ограниченности ресурсов потребовалось изучить тему многопоточности в android. Нам не следует загружать главный поток, иначе приложение может перестать реагировать на действия пользователя. Операционная система имеет функцию распознавания загруженности главного потока, в таком случае она закроет приложение с ошибкой. Следовательно, было необходимо использование дополнительных потоков, чтобы не мешать взаимодействию пользователя с интерфейсом. Таким образом, был использован AsyncTask, чтобы решить данную проблему. Итак, главным процессом в моей работе является получение данных от контролера внутри тренажера, который оснащен датчиками давления, и считывание силы удара, времени, а также уровня заряда тренажера.

Исходя из проведенных исследований, можно сделать вывод, что процесс соединения мобильного приложения и устройства является крайне интересным и быстроразвивающимся в наши дни. Регулярно появляется множество идей и методов для создания подобных процессов – от дистанционного управления домашними приборами, лампочками, бытовой техникой до сбора данных о характеристиках человека, таких как частота сердечных ударов или количество пройденных за день шагов. Считаю цель своей курсовой работы достигнутой, созданное приложение способно измерять силу удара при помощи тренажера, оно лаконично, удобно и интуитивно понятно для обычного пользователя. У данной темы есть множество перспектив для развития, например, создание общей базы данных для тренера, чтобы он мог анализировать прогресс своих учеников и т.д.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ И ИНТЕРНЕТ-РЕСУРСОВ

Электронные ресурсы:

1. Bluetooth overview. — Текст : электронный // developer.android : [сайт]. — URL: https://developer.android.com/guide/topics/connectivity/bluetooth (дата обращения: 04.05.2022).
2. Работаем с Bluetooth. — Текст : электронный // developer.alexanderklimov : [сайт]. — URL: http://developer.alexanderklimov.ru/android/bluetooth.php (дата обращения: 04.05.2022).
3. Потоки. — Текст : электронный // developer.alexanderklimov : [сайт]. — URL: http://developer.alexanderklimov.ru/android/theory/thread.php (дата обращения: 04.05.2022).
4. Что такое многопоточность. Работа с потоками в java. — Текст : электронный // fandroid : [сайт]. — URL: https://www.fandroid.info/chto-takoe-mnogopotochnost-rabota-s-potokami-v-java/ (дата обращения: 04.05.2022).
5. Потоки. Класс Thread и интерфейс Runnable. — Текст : электронный // developer.alexanderklimov : [сайт]. — URL: http://developer.alexanderklimov.ru/android/java/thread.php (дата обращения: 04.05.2022).
6. AsyncTask. — Текст : электронный // developer.android : [сайт]. — URL: https://developer.android.com/reference/android/os/AsyncTask (дата обращения: 04.05.2022).
7. Урок 86. AsyncTask. Знакомство, несложный пример. — Текст : электронный // startandroid : [сайт]. — URL: https://startandroid.ru/ru/uroki/vse-uroki-spiskom/149-urok-86-asynctask-znakomstvo-neslozhnyj-primer.html (дата обращения: 04.05.2022).
8. Класс AsyncTask. — Текст : электронный // developer.alexanderklimov : [сайт]. — URL: http://developer.alexanderklimov.ru/android/theory/asynctask.php (дата обращения: 04.05.2022).