ФЕДЕРАЛЬНОЕ Государственное АВТОНОМНОЕ образовательное учреждение Высшего образования

«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ»

(НИУ «БелГУ»)

ИНСТИТУТ ИНЖЕНЕРНЫХ И ЦИФРОВЫХ ТЕХНОЛОГИЙ

КАФЕДРА ОБЩЕЙ МАТЕМАТИКИ

РАЗРАБОТКА МОБИЛЬНОГО ПРИЛОЖЕНИЯ ПО ГЕНЕРАЦИИ И РАСПОЗНАВАНИЮ КОДА МОРЗЕ

Выпускная квалификационная работа

обучающегося по направлению подготовки

01.03.02 Прикладная математика и информатика

очной формы обучения, группы 12001510

Зайцевой Виктории Викторовны

Научный руководитель

к. ф.-м. н., доцент

Никуличева Т.Б.

БЕЛГОРОД 2019

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Азбука Морзе была изобретена в 1838 году. Также называющаяся «Код Морзе», она является системой знакового кодирования, представляющей буквы, знаки препинания, цифры и прочие символы различными последовательностями из двух сигналов – точек, и тире. Азбука Морзе получила широкую популярность с началом Первой мировой войны, когда требовался какой-либо шифр, позволяющий использовать все возможности уже значительно развитой к тому времени радиосвязи. Позже код Морзе получил распространение в армии, стал на десятилетия стандартом для флота, в том числе и гражданского, а некоторые сигналы, записываемые с помощью азбуки, например, SOS, получили всемирную известность.

В данный момент азбука Морзе значительно утратила свою популярность ввиду сильного совершенствования средств радиосвязи. Тем не менее, среди различных энтузиастов-радиолюбителей использование азбуки всё ещё распространено.

Цель данной работы – создание мобильного приложения по генерации и распознаванию кода Морзе.

Данное приложение было бы эффективно при передаче информации с помощью кода на небольших, относительно покрытия мобильной связи, дистанциях. При этом, такой способ передачи информации мог бы быть эффективен в случае отсутствия или прерываний покрытия оператора, в условиях мощных погодных осадков (снегопад, буран), а также для прочих целей. Наконец, приложение было бы актуально как платформа для создания прочих приложений по сообщению различными визуальными или звуковыми кодами на близких дистанциях, для аналогичных целей.

Для достижения поставленной цели требуется решить следующие задачи:

- обзор аналогов приложения, рассмотрение их достоинств и недостатков;

- определение операционной системы и выбор минимальной версии ОС, поддерживаемой приложением;
- выбор инструментальной среды, в которой будет вестись разработка;
- разработка мобильного приложения с помощью данной инструментальной среды;
- тестирование приложения;

Структура и объем работы: выпускная квалификационная работа выполнена на 59 страницах машинописного текста, состоит из введения, четырёх глав, заключения.

В первой главе проводится анализ аналогичные и схожие с разработанным приложения.

Во второй главе выбирается операционная система и интегрированная среда разработки.

В третьей главе описывается процесс разработки мобильного приложения.

В четвертой главе проводится тестирование программного продукта.

1 СОСТОЯНИЕ ВОПРОСА И АНАЛИЗ ПРОГРАММНЫХ ПРОДУКТОВ

1.1 Состояние вопроса

Под криптографией понимают науку о методах обеспечения конфиденциальности (невозможного прочтения информации посторонним), целостности данных (невозможности незаметного изменения информации) и аутентичности (подлинности авторства) информации.

Для осуществления полноценного процесса передачи информации на большие расстояния необходимо, чтобы сообщение дошло до получателя без искажений и в первоначальном объеме, соответствуя первоначальному сообщению отправителя. Для этого удобно подвергнуть сообщение предварительному кодированию (например, записать его через двоичный код), а затем получатель применит алгоритм обратного дешифрования. Общую схему шифрования информации можно представить в следующем виде:

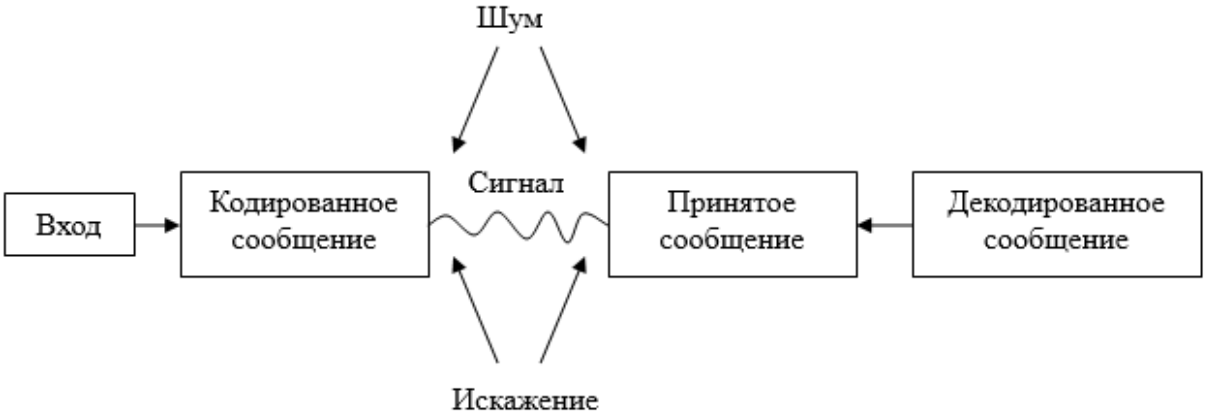

Рисунок 1.1 — Общая схема кодирования и декодирования информации.

Алгоритмы, предназначенные для шифрования информации бывают разными по сложности и надежности, и, в первую очередь, зависят от конечной цели кодирования.

Удобный в применении алгоритм должен отвечать следующим требованиям:

- экономность (т.е., запись должна быть максимально короткой и не занимать большой объем памяти);
- надежность (при необходимости информация может быть засекречена, чтобы она оказалась недоступной не предназначенным лицам);
- удобство обработки или восприятия (алгоритм кодирования и декодирования должен быть интуитивно понятен пользователю).

Для обеспечения целей кодирования и декодирования информации чаще всего используются специальные системы кодов, состоящие из символов и знаков, а также алгоритмов представления информации.

Существует огромное количество алгоритмов и способов шифрования и дешифрования информации. Одним из широко распространенных и часто используемых алгоритмов является кодирование с помощью азбуки Морзе.

Азбука Морзе позволяет кодировать буквы различных алфавитов, цифры, знаки препинания и т.д., используя последовательность двух символов «точка» и «тире» (в случае двоичного кода «0» и «1»). За единицу времени принимается длительность одной точки. Длительность тире равна трём точкам. Пауза между элементами одного знака – одна точка, между знаками в слове – 3 точки, между словами – 7 точек.

Впервые азбука Морзе была использована в 1838 году в качестве телеграфного кода для телеграфного аппарата изобретенного ранее. Современное представление международного кода Морзе появился относительно недавно – в 1939 году, в тот момент, когда была проведена последняя корректировка, затронувшая в основном знаки препинания. Первоначальный вариант кода Морзе использовался вплоть до середины 60-х годов XX века на железнодорожных станциях. Сейчас азбука Морзе продолжает использоваться на флоте при передаче сообщений на дальние расстояния в условиях радиопомех, а также она применяется в МЧС, а именно в Гражданской обороне и весьма популярна в кругах любителей радиоэлектроники.

В таблице 1 представлена современная интерпретация азбуки Морзе.

Таблица 1.1 Символьный код Морзе

Русская буква	Латинская буква	Код Морзе	Русская буква	Латинская буква	Код Морзе	Символ	Код Морзе
А	A	· —	Р	R	· — ·	1	· — — — —
Б	B	— · · ·	С	S	· · ·	2	· · — — —
В	W	· — —	Т	T	—	3	· · · — —
Г	G	— — ·	У	U	· · —	4	· · · · —
Д	D	— · ·	Ф	F	· · — ·	5	· · · · ·
Е (Ё)	E	·	Х	H	· · · ·	6	— · · · ·
Ж	V	· · · —	Ц	C	— · — ·	7	— — · · ·
З	Z	— — · ·	Ч	O	— — — ·	8	— — — · ·
И	I	· ·	Ш	CH	— — — —	9	— — — — ·
Й	J	· — — —	Щ	Q	— — · —	0	— — — — —
К	K	— · —	Ъ	N	— — · — —	Точка	· · · · · ·
Л	L	· — · ·	Ы	Y	— · — —	Запятая	· — · — · —
М	M	— —	Ь (Ъ)	X	— · · —	—	· · — — · ·
Н	N	— ·	Э	E	· · — · ·	!	— — · · — —
О	O	— — —	Ю	U	· · — —	@	· — — · — ·
П	P	· — — ·	Я	A	· — · —	Конец связи (end contact)	· · — · —

Цель дипломной работы – создание мобильного приложения по генерации и распознаванию кода Морзе.

Для достижения данной цели следует рассмотреть аналоги приложения.

На сегодняшний день разработано несколько приложений, использующих алгоритм кодирования и декодирования азбуки Морзе. Рассмотрим некоторые из них, и их достоинства и недостатки.

1.2 Мобильные приложения

1.2.1 Приложение «Азбука Морзе» от разработчика Tiralkapplications.

Приложение предназначено для кодирования и декодирования информации с помощью звуковых сигналов.

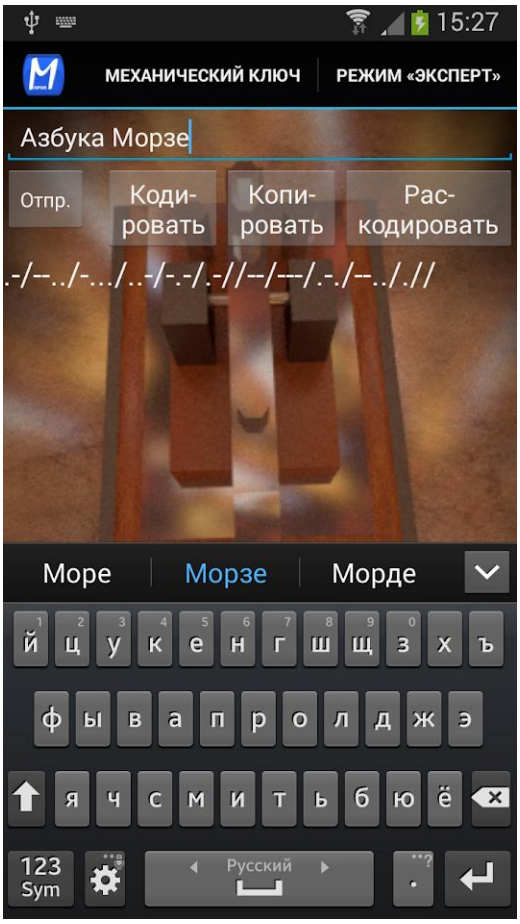

Рисунок 1.2 – Окно приложения «Азбука Морзе»

Достоинства:

1) информацию можно передавать латиницей и кириллицей.

Недостатки:

2) ключ работает с задержкой, подобраны неправильные интервалы между словами;

3) не предусмотрена возможность менять тональность и скорость воспроизведения сигнала;

4) необходимо первоначальное знание азбуки Морзе.

1.2.2 Приложение «Фонарик, азбука морзе + сирена» от разработчика katapuf.

Приложение является конвертером азбуки Морзе в звуковые и световые сигналы.

Рисунок 1.3 – Окно приложения «Фонарик, азбука морзе + сирена»

Достоинства:

1) позволяет передавать информацию не только с помощью звука, но и за счет вспышки фонарика.

Недостатки:

1) при выключенном дисплее выключается вспышка фонарика;

2) очень емкое приложение, сильно расходует заряд батареи;

3) поддерживает только латиницу.

1.2.3 Приложение «Азбука Морзе» от разработчика Android.

Приложение является самоучителем для изучения азбуки Морзе.

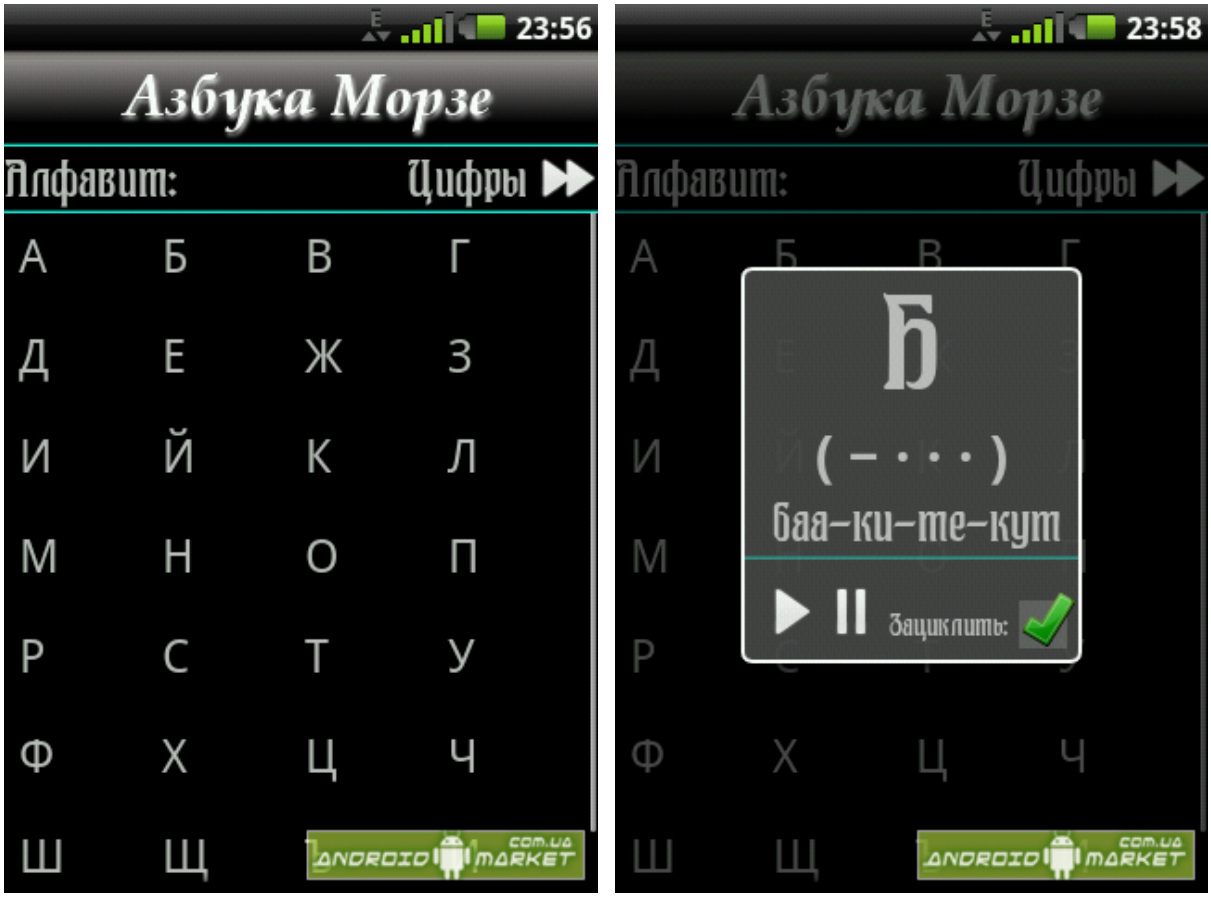

Рисунок 1.4 – Окно приложения «Азбука Морзе»

Достоинства:

1) есть функция воспроизведения и зацикливания звуков азбуки.

Недостатки:

1) не поддерживает передачу светового сигнала;

2) поддерживает только кириллицу.

1.2.4 Приложение «Morse Code» от разработчика predefault.

Данное приложение позволяет преобразовывать текст в Морзе и наоборот.

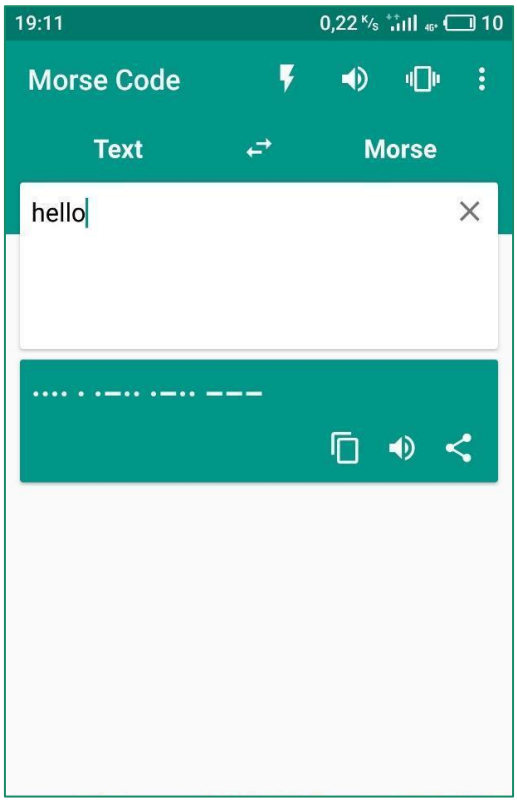

Рисунок 1.5. – Окно приложения «Morse Code»

Достоинства:

1) позволяет передавать информацию не только с помощью звука, вспышки фонарика и вибрации

2) быстрое преобразование текста;

Недостатки:

1) нельзя настроить скорость передачи кода Морзе;

2) реклама;

3) поддерживает только латиницу;

1.3 Результат анализа существующих аналогов

После рассмотрения аналогов можно выделить следующие критерии для общего сравнения:

1) кодирование и декодирование текста;
2) передача светового сигнала;
3) передача звукового сигнала;

4) передача сигнала с помощью вибрации;
5) поддержка латиницы;
6) поддержка кириллицы;
7) распознавание светового и звукового сигнала;

Можно провести сравнение найденных аналогов, основываясь на данных критериях (таблица 1.2).

Таблица 1.2. Сравнение найденных аналогов

	«Азбука Морзе» Tiralkapplications	**«Фонарик, азбука Морзе + сирена»**	**«Азбука Морзе»**	**«Morse Code»**
Кодирование и декодирование текста	+	-	+	+
Передача светового сигнала	-	+	+	+
Передача звукового сигнала	+	+	-	+
Передача сигнала с помощью вибрации	-	-	-	+
Поддержка латиницы	+	+	-	+
Поддержка кириллицы	+	-	+	-
Распознавание светового и звукового сигнала	-	-	-	-

На основании проведённого анализа, сделан следующий вывод.

Разрабатываемое приложение должно поддерживать такие функции, как поддержка латиницы и кириллицы, ввод данных с помощью аудиозаписи и экранной клавиатуры, вывод данных с помощью световых сигналов, символьного вывода, звуковых сигналов и вибрации. Так же, приложение не должно содержать рекламу.

2 ВЫБОР СРЕДСТВ РАЗРАБОТКИ

2.1 Выбор операционной системы

На выбор операционной системы для приложения повлияли следующие факторы:

1) Разрабатываемое приложение мобильное, т.е. предназначено для использования на портативных устройствах в различных, в том числе осложнённых из-за погоды, условиях.

2) Операционная система Google Android, согласно статистике, на данный момент наиболее популярна среди мобильных операционных систем (рисунок 2.1).

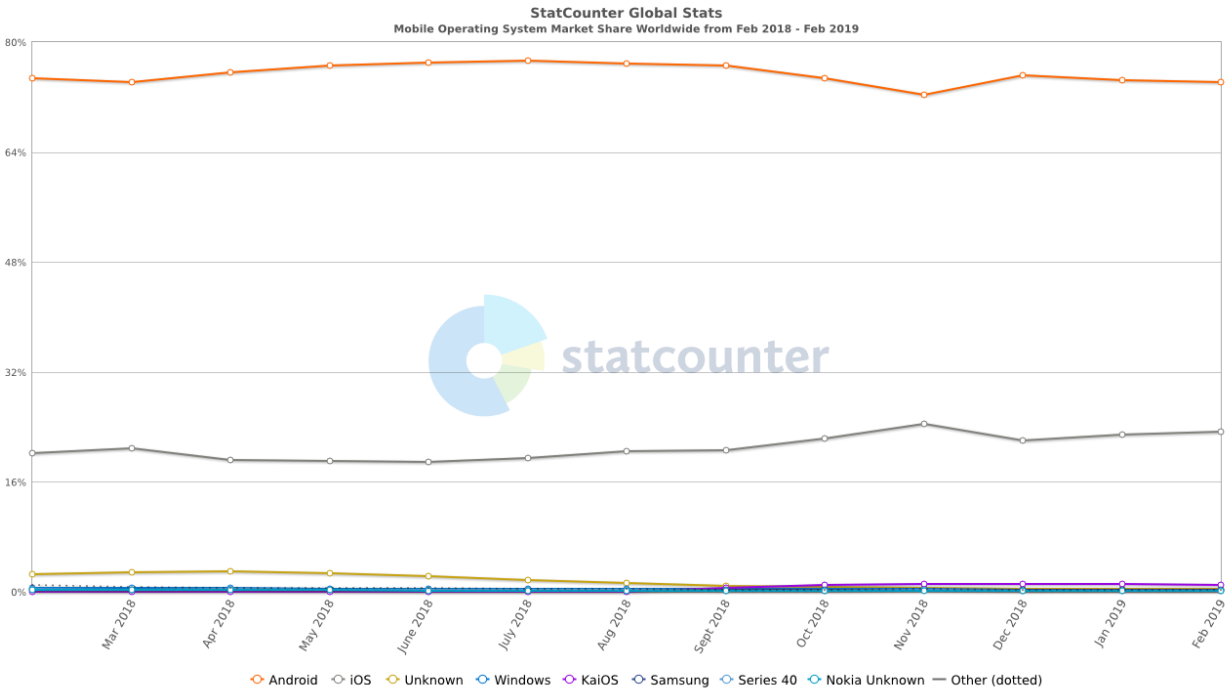

Рисунок 2.1 – Разделение рынка мобильных операционных систем (февраль 2018 – февраль 2019)

Если приложение для ОС iOS было бы пригодно только для взаимодействия с устройствами Apple, то мобильное приложение для Android может быть установлено на широкий ряд платформ самой разной доступности. Та-

ким образом, охват аудитории для приложения на ОС Android в любом случае будет дольше, чем для ОС iOS.

Таким образом, был сделан выбор в пользу ОС Android – оптимальной платформы для реализации проекта. Наконец, выбор минимальной версии ОС, поддерживаемой приложением, основан на данной статистике (рисунок 2.2):

Version	Codename	API	Distribution
2.3.3 - 2.3.7	Gingerbread	10	0.2%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	0.3%
4.1.x	Jelly Bean	16	1.1%
4.2.x		17	1.5%
4.3		18	0.4%
4.4	KitKat	19	7.6%
5.0	Lollipop	21	3.5%
5.1		22	14.4%
6.0	Marshmallow	23	21.3%
7.0	Nougat	24	18.1%
7.1		25	10.1%
8.0	Oreo	26	14.0%
8.1		27	7.5%

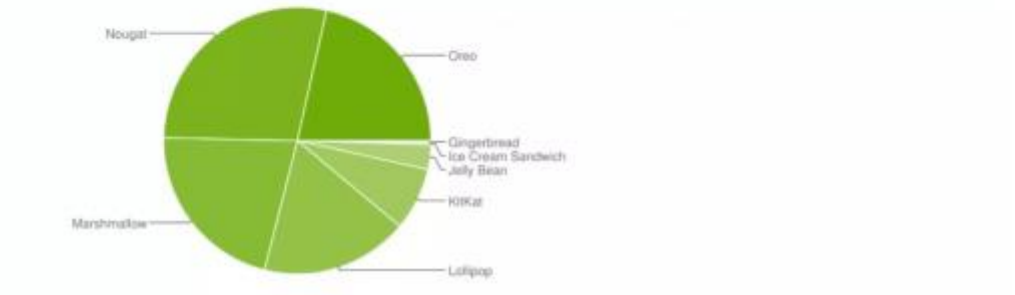

Рисунок 2.2 – Распределение пользователей по версиям Android

Данная статистика принадлежит компании Google, разработчику ОС Android. Согласно этой статистике, наиболее логично выбрать минимальной из поддерживаемых версий ОС 4.4 KitKat, поскольку процент более ранних версий Android незначителен, в то время как оптимизация приложения для

работы с устаревшими версиями операционной системы ведёт, например, к использованию deprecated-методов.

2.2 Выбор интегрированной среды разработки

Для операционной системы Android существуют три основные среды разработки:

- Eclipse;
- Intellij IDEA;
- Android Studio.

2.2.1 Среда разработки Eclipse

Eclipse разрабатывается и поддерживается компанией Eclipse Foundation, последняя сборка датирована 2018 годом. За счёт модульности данную среду разработки легко настроить под себя, например, подключить Gradle в качестве сборщика проектов. Тем не менее, импорт проектов из InteliJ и Android Studio затруднён различием в архитектуре собранных проектов и по ряду других причин.

При создании нового проекта для Android автоматически создается пустой «Hello World» проект, который можно даже запустить на девайсе пользователя или в эмуляторе. Eclipse поддерживает Java-программирование для Android. Отдельный плюс – простота использования, а также множество бесплатных модулей, разработанных сторонними пользователями.

Тем не менее, сравнительно со средами разработки IntelIJ и Android Studio, в Eclipse затруднена работа с визуальным интерфейсом программы, а из-за широкой модульности настройка среды требует большого количества усилий.

На рисунке 2.3 представлен Интерфейс среды разработки Eclipse.

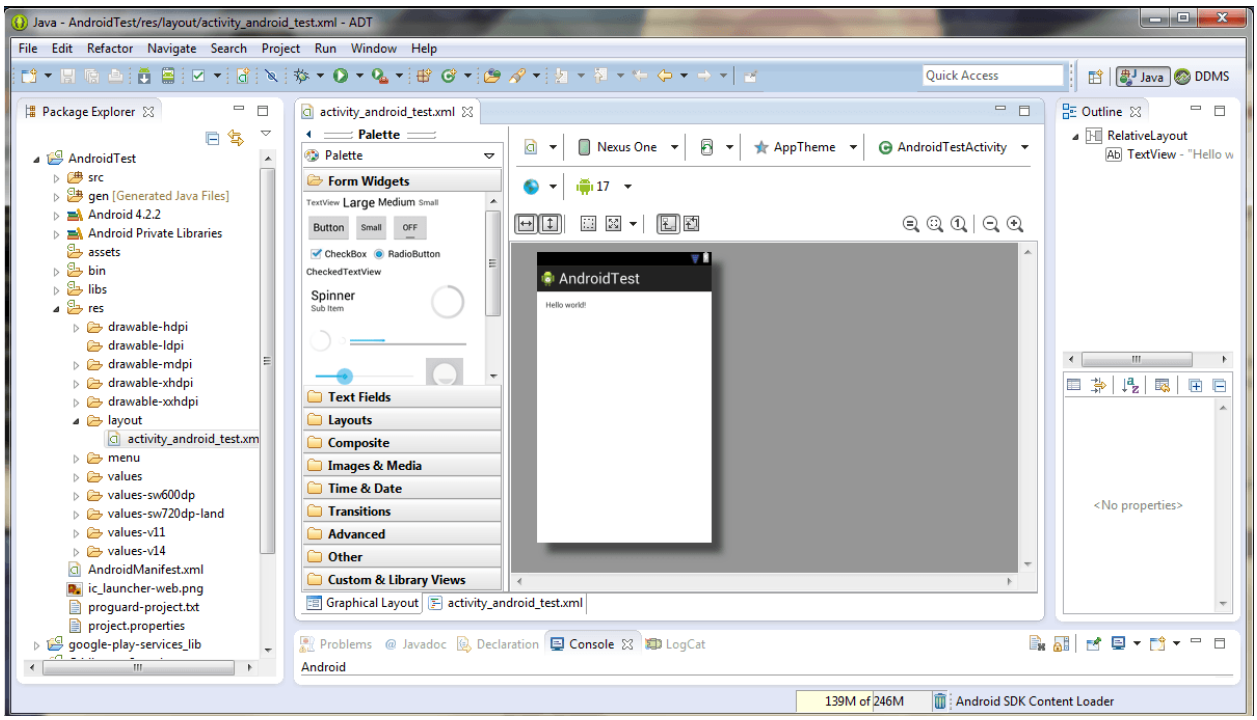

Рисунок 2.3 – Интерфейс среды разработки Eclipse

2.2.2 Среда разработки Intellij IDEA

Данная IDE бесплатна в составе пакета Community Edition. Помимо Android-разработки используется также для написания приложений для других платформ, поддерживает множество языков, включая Ruby, PHP, Go, Python и другие, а также множество различных фреймворков и популярных инструментов разработки, например, Maven. Возможности данной среды также могут быть расширены с помощью различных плагинов от сторонних разработчиков. Также использует Gradle.

В данной IDE весьма удачно решены некоторые немаловажные для программистов проблемы, например, называния переменных. Например, в проекте есть базовый класс «Item». Тогда при создании из этих элементов массив, то среда разработки автоматически предложит назвать их «Items».. Например, по статистике за год на одних только операция ‘rename’ можно сэкономить около 120 часов времени.

Поддерживает взаимодействие с эмуляторами ОС Android. В платной версии среды, пользователи получают также техническую поддержку от специалистов компании JetBrains, разработчика данного ПО.

Из недостатков среды можно отметить то, что версия среды с более полным набором возможностей платна, а также то, что у данной IDE нет заранее предустановленного эмулятора Android.

Также к недостаткам можно отнести то, что техническая документация представлена в основном на английском языке, а также закрытость проекта.

На рисунке 2.4 представлен интерфейс среды разработки Intellij IDEA.

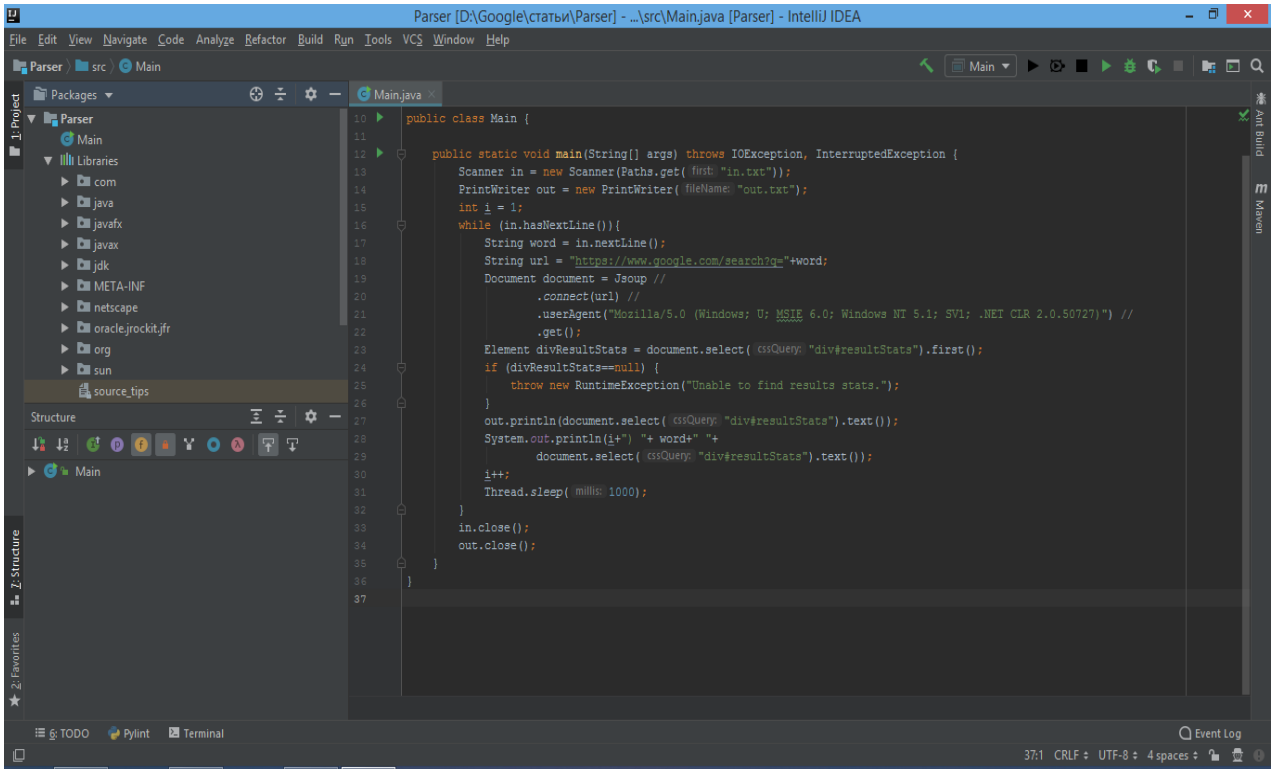

Рисунок 2.4 – Интерфейс среды разработки Intellij IDEA

2.2.3 Среда разработки Android Studio

Android Studio появилась в результате сотрудничества JetBrains (разработчиков Intellij IDEA) и Google, по сути, представляет собой версию InteliJ IDEA, заточенную специально под разработку Android-приложений. Это выражается в различных особенностях интерфейса, например, в возможности

быстро и удобно выбрать при создании проекта версию ОС, выбрать тип мобильных устройств (часы, телевизор, смартфон), в наличии уже встроенного эмулятора Android, а также в различных небольших усовершенствованиях интерфейса, ставших следствием ориентированности IDE под разработку исключительно на одной ОС.

Например, встроенный эмулятор включает в себя максимально широкий спектр устройств на базе Android, от планшетов и смартфонов, до смарт-часов, причём для каждой платформы возможен выбор из нескольких версий операционной системы, и разных версий API.

Недостатками IDE можно назвать закрытость и некоторую нестабильность работы эмулятора. Тем не менее, от основных недостатков Eclipse и IntelIJ Idea данное ПО, с учётом специфики разработки под Android, избавлено.

Поскольку данная среда разработки абсолютно бесплатна, и притом сочетает в себе широкую функциональность InteliJ, заточенность под разработку на Android, доступность интерфейса и простоту настройки, именно она была выбрана для создания приложения.

На рисунке 2.5 представлен интерфейс среды разработки Android Studio.

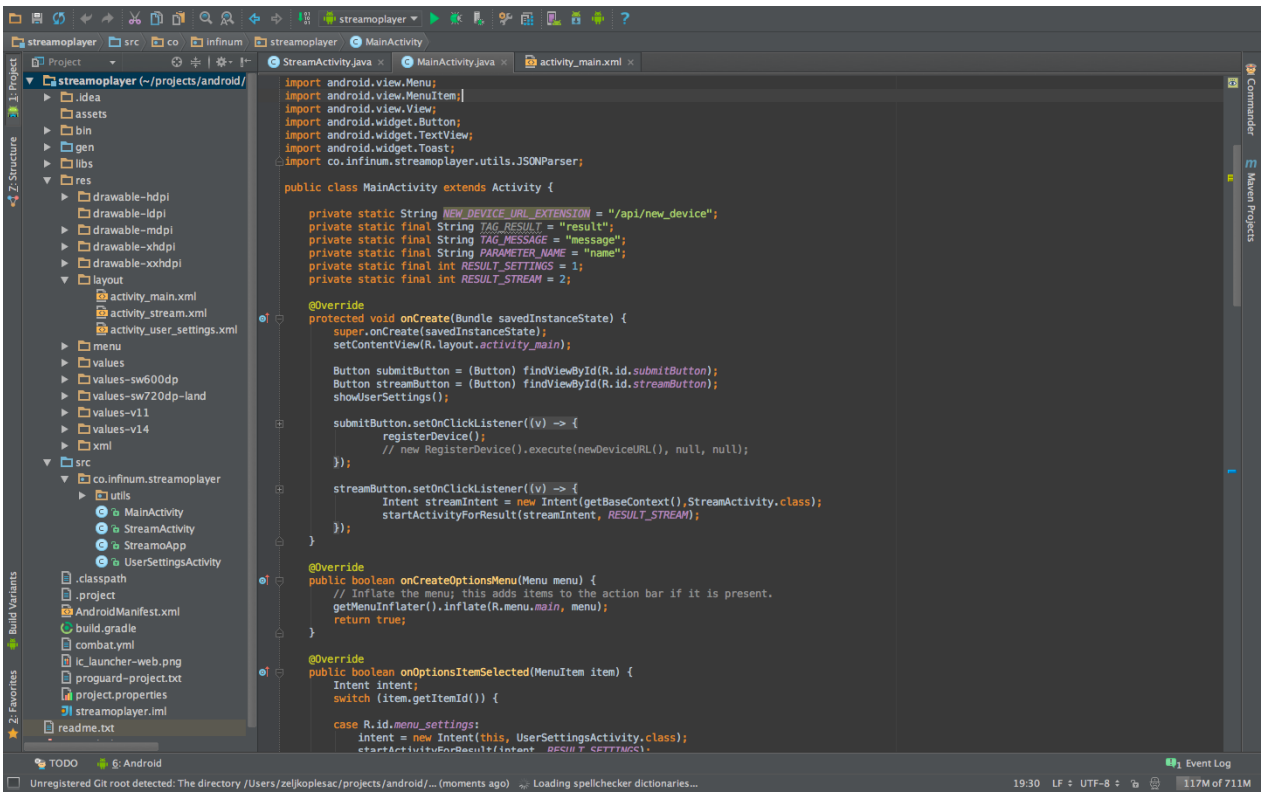

Рисунок 2.5. – Интерфейс Android Studio

3 РАЗРАБОТКА ПРИЛОЖЕНИЯ ПО ГЕНЕРАЦИИ И РАСПОЗНАВАНИЮ КОДА МОРЗЕ

3.1 Формирование требований к разрабатываемому приложению

После рассмотрения аналогов, выбора операционной системы и среды разработки, были сформированы следующие требования к функциональности приложения:

- преобразование текста в Морзе и наоборот;
- передача информации с помощью звука, фонарика и вибрации;
- распознавание светового сигнала;
- распознавание звука;
- поддержка латиницы и кириллицы;
- настройка скорости передачи кода Морзе.

Интерфейс приложения должен обладать следующими свойствами:

- все опции предложения должны быть доступны не более чем за 2 нажатия с момента запуска приложения;
- интерфейс должен предлагать пользователю максимальное количество опций, занимающих минимально возможное пространство на экране, и притом легкодоступных.

3.2 Структура приложения

Структура приложения представлена на рисунке 3.1.

Рисунок 3.1 – Структура приложения

Разработанное приложение имеет главное окно, которое содержит следующие элементы:

- поле записи текста;
- поле с результатом перевода;
- кнопка выбора языка записываемого текста;
- кнопка для распознавания речи;
- кнопка для активации повторения передачи сообщения;
- кнопка для запуска и остановки передачи сообщения;
- кнопки для активации вывода посредством вибрации, звукового или светового сигнала;
- навигационное меню.

Навигационное меню служит для перехода на следующие окна;

- окно распознавания светового сигнала;
- окно записи и распознавания звука;
- окно справки, которое содержит информацию об азбуке Морзе;
- окно настроек с опциями для вывода шифрованных данных.

3.3 Разработка мобильного приложения

3.3.1 Главное окно

В приложении есть несколько классов. Главный класс – MainActivity. Представляет собой главный экран, который запускается, как только пользователь открывает приложение. Именно MainActivity запускает остальные посредством различных методов.

В качестве графического интерфейса MainActivity устанавливает разметку из файла ресурса разметки дизайна activity_main.xml (рисунок 3.2).

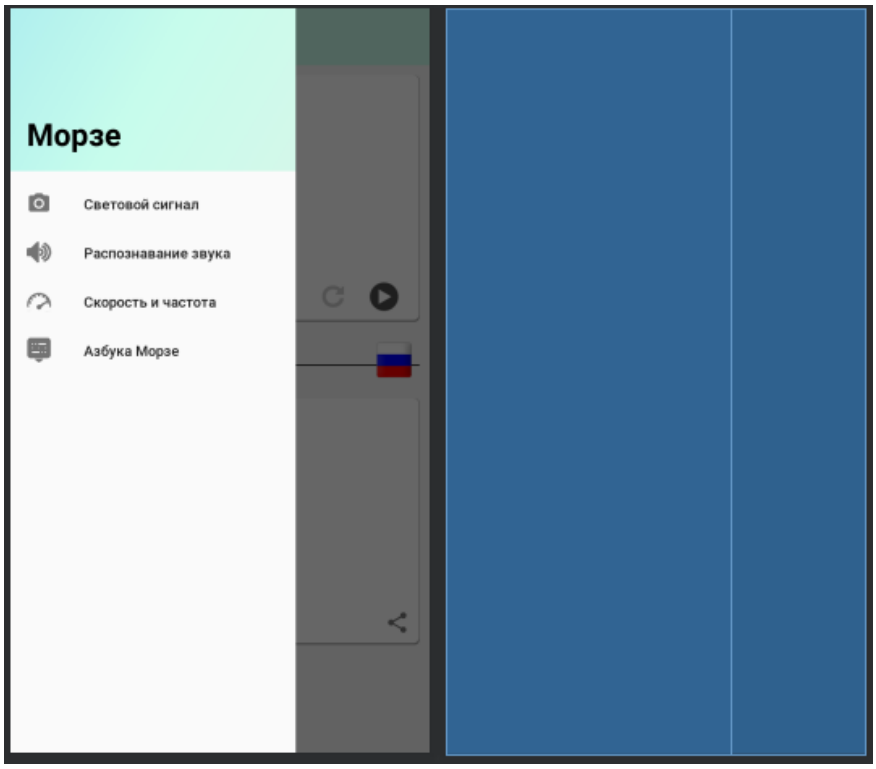

Рисунок 3.2 – activity_main.xml

В качестве корневого элемента макета activity_main.xml используется класс android.support.v4.widget.DrawerLayout для реализации Navigation Drawer. Navigation Drawer представляет собой навигационное меню, выезжающее слева или справа экрана по свайпу пользователя (плавное проведение пальцем).

За выдвигающееся меню отвечает элемент NavigationView, который входит последним в контейнере DrawerLayout. А перед меню находится вставка include, указывающая на разметку app_bar_main.xml.

Тег NavigationView содержит ссылку на собственную разметку в атрибуте app:headerLayout, который указывает на файл nav_header_main.xml – верхняя часть навигационного меню (рисунок 3.3).

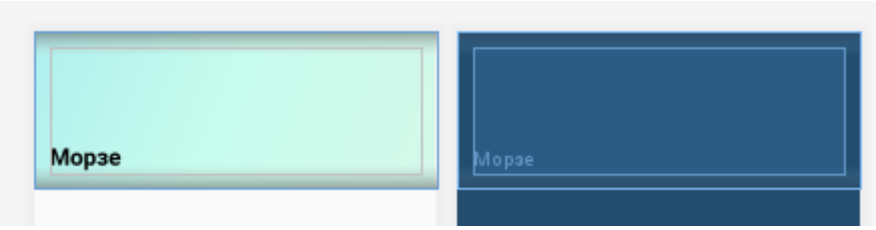

Рисунок 3.3 – nav_header_main.xml

Фон контейнера определён в ресурсе drawable/side_nav_bar.xml и представляет собой градиент. Также NavigationView содержит ссылку на меню в атрибуте app:menu, который ссылается на ресурс меню menu/activity_main_drawer.xml (рисунок 3.4).

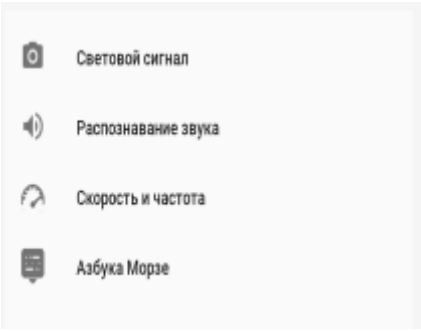

Рисунок 3.4 – activity_main_drawer

С помощью выдвигающегося меню выполняется переход на другие окна.

В классе MainActivity реализуется интерфейс OnNavigationItemSelectedListener с его методом onNavigationItemSelected().

В нём определяется идентификатор выбранного пункта и в зависимости от этого выполняется переход на нужное окно.

Navigation Drawer содержит ссылки на следующие Activity:

1. CameraActivity – окно, которое содержит интерфейс для распознавания светового сигнала.
2. RecordActivity – окно, которое содержит интерфейс для аудиозаписи шифруемого сообщения.
3. SettingsActivity – окно настроек звукового, вибрационного и светового вывода приложения.
4. MorseActivity – окно, содержащее справку по азбуке Морзе.

Для перехода используется Intent, который представляет собой объект обмена сообщениями, с помощью которого можно запросить выполнение действия у компонента другого приложения.

Для запуска другого экземпляра компонента Activity необходимо передать объект Intent методу startActivity(). Объект Intent описывает операцию, которую требуется запустить, а также содержит все остальные необходимые данные.

Если после завершения операции от нее требуется получить результат, то вызывается метод startActivityForResult().

В приложении для запуска CameraActiviry, RecordActivity, SettingsActivity используется метод startActivityForResult(), а для запуска MorseActivity – startActivity().

В методе onNavigationItemSelected() создается Intent, указывается класс второй Acivitю Методу startActivityForResult() в качестве параметров передаётся Intent и requestCode. requestCode – необходим для идентификации. В качестве requestCode используется константы.

Во второй Acivity ставится статус RESULT_OK, указывается, что надо вернуть объект intent в качестве результата и Activity закрывается.

Когда вторая активность закрывается, то в MainActivity срабатывает метод onActivityResult.

В onActivityResult следующие параметры:

- requestCode – идентификатор, по которому определяется Activity, с которой пришел результат. Идентификатор такой же как в методе startActivityForResult().
- resultCode – код возврата, определяющий успешно прошел вызов или нет.
- data – Intent, в котором возвращаются данные

Если resultCode не равен RESULT_OK, значит что-то пошло не так. Это может произойти, например, если нажать кнопку Назад.

Разметка app_bar_main содержит Toolbar и вставку include, указывающую на разметку content_main.

На рисунке 3.5 представлена разметка content_main.xml.

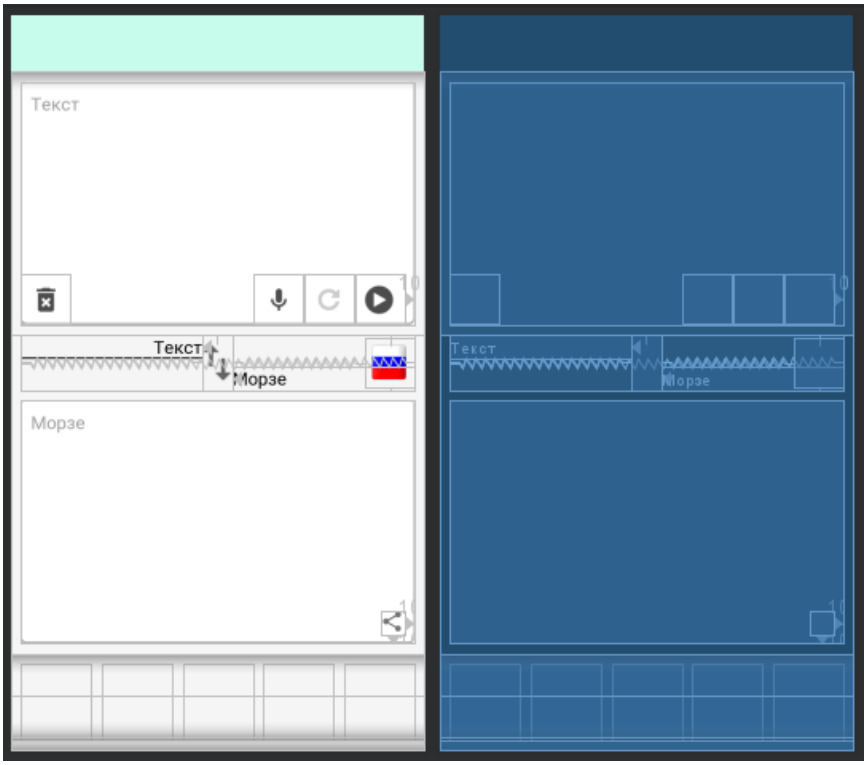

Рисунок 3.5 – content_main.xml

В разметке content_main.xml используются следующие компоненты:

1. EditText – текстовое поле с возможностью ввода и редактирования текста.

2. ImageButton – кнопка с изображением. Компонент ImageButton обладает всеми свойствами обычной кнопки, но имеет возможность использовать изображение вместо текста.

3. TextView – компонент для вывода текста на экран.

4. FrameLayout – слой, внутри которого помещаются компоненты и другие слои. Компоненты располагаются друг на друга.

5. LinearLayout – слой, внутри которого помещаются компоненты и другие слои. Компоненты располагаются последовательно.

6. RelativeLayout – слой, внутри которого помещаются компоненты и другие слои. Компоненты располагаются относительно друг друга или же самого RelativeLayout.

MainActivity имеет два текстовых поля. В верхнее поле можно ввести текст и преобразовать его в код Морзе и наоборот, с помощью специальной клавиатуры ввести код Морзе и преобразовать его в текст. Результат отображается в нижнем текстовом поле.

Для кодирования и декодирования был создан вспомогательный класс Morse. Он содержит методы для кодирования и декодирования текста. В нём используется библиотека Guava. Guava – это набор библиотек, который включает новые типы коллекций.

Можно переводить с русского или английского языков. Для выбора языка добавлена кнопка, при нажатии на которую открывается окно LanguageActivity.

В левом нижнем углу первого текстового поля расположена кнопка для удаления всего введённого текста. В правом нижнем углу находятся три кнопки. Первая кнопка предназначена для голосового ввода текста. Вторая кнопка предназначена для повторения передачи сообщения. Третья кнопка предназначена для передачи сообщения с помощью звука, вибрации, фонарика. Выбрать с помощью чего будет сгенерирован сигнал можно с помощью трёх элементов меню.

Элементы меню определяются в файле menu/main.xml (рисунок 3.6)

Рисунок 3.6 – Ресурс меню main.xml

Меню в приложениях представляет класс android.view.Menu, и каждая activity ассоциируется с объектом этого типа. Объект android.view.Menu может включать различное количество элементов.

Чтобы вывести меню на экран, в классе MainActivity переопределяется метод onCreateOptionsMenu(). Чтобы привязать к меню действия, надо переопределить метод onOptionsItemSelected().

Чтобы понять, какой пункт меню выбран, нужно получить его идентификатор int id = item.getItemId().

Генерация сигнала выполняется в отдельном потоке. Передачу сообщения можно остановить.

Для работы со вспышкой был создан интерфейс FlashLight и классы FlashLight_Post_Marshmallow и FlashLight_Pre_Marshmallow, реализующие этот интерфейс.

В классе FlashLight_Post_Marshmallow для работы со вспышкой используется класс CameraManager, который не поддерживается устройствами на API ниже 23 (Android 6.0). В классе FlashLight_Pre_Marshmallow используется класс Camera, который считается устаревшим на API ниже 23.

Для выполнения некоторых операций или доступа к определенным ресурсам в операционной системе Android, приложение должно иметь разрешение.

Чтобы можно было использовать вспышку, требуется запросить разрешение у пользователя.

Был создан вспомогательный класс Util, который содержит метод requestPermission(), который позволяет запросить нужное разрешение у пользователя.

Также в данном классе находятся методы для проверки, поддерживает ли устройство функцию вспышки и доступна ли камера.

Для генерации звука был создан класс SoundGenerator.

SoundGenerator позволяет генерировать звук определённой длительности, частоты (тона) и частоты дискретизации.

3.3.2 Распознавание звука

Код окна для записи звука, содержится в классе RecordActivity. В качестве графического интерфейса RecordActivity устанавливает разметку из файла ресурса разметки дизайна activity_record.xml (рисунок 3.7).

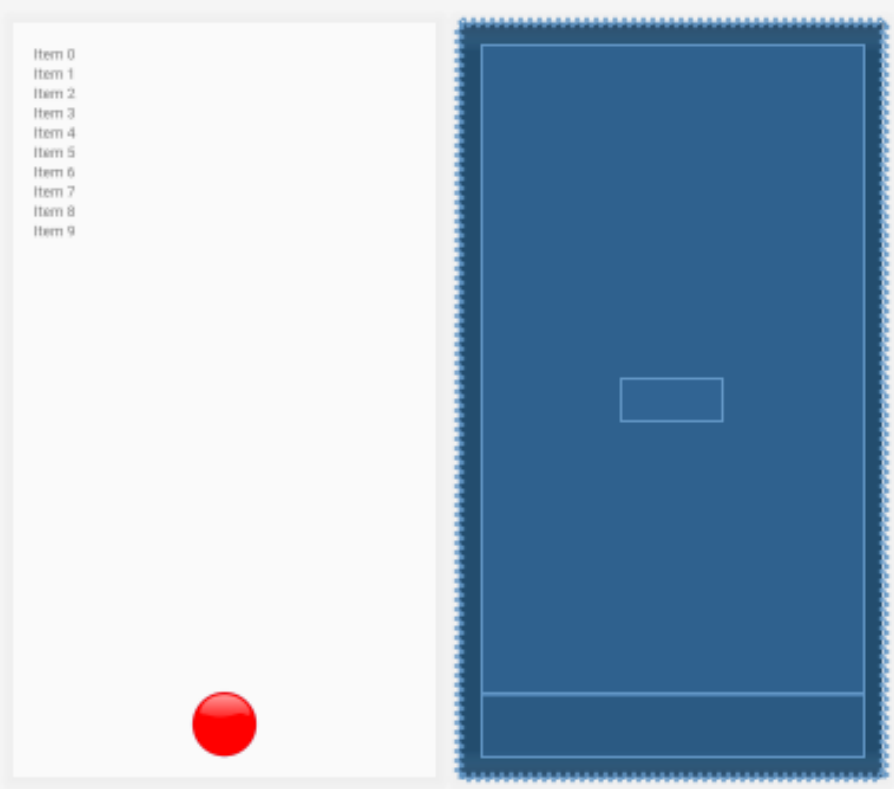

Рисунок 3.7 – activity_record.xml

В разметке activity_record.xml используются следующие компоненты:

1)RecyclerView;

2)Chronometer;

3)ImageButton;

4)FrameLayout;

RecyclerView – это компонент, который позволяет создавать прокручиваемый список. Используется для отображения списка записей.

Для тогда чтобы отобразить компоненты списка RecyclerView, необходимо создать отдельную разметку, как показано на рисунке 3.8.

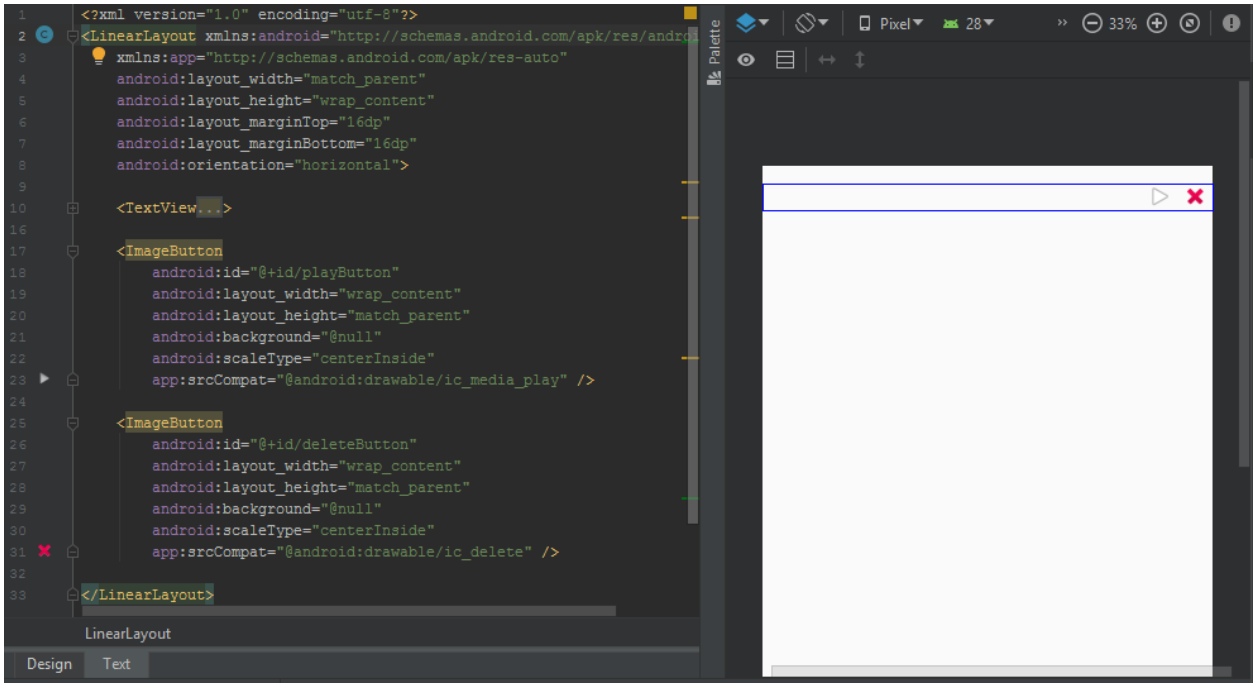

Рисунок 3.8 – Часть кода разметки и визуальное отображение компонентов на экране эмулятора ОС Android

У каждого компонента RecyclerView должен быть свой Adapter и Holder. Holder нужен для того, чтобы объявить все используемые в элементе списка компоненты, задать им имена и присвоить уникальные идентификаторы. В свою очередь, Adapter нужен для формирования самого списка записей.

Chronometer – компонент, позволяющий запускать и останавливать начальный отсчёт времени.

Компонент Chronometer, хоть его и не видно, но он располагается в середине экрана. За видимость компонента отвечает свойство View.Visibility, параметрами которого являются: visible, invisible и gone. Visible присваивает-

ся компоненту, который должен оставаться в видимости для пользователя, invisible – невидимым.

При нажатии на кнопку начинается запись. RecyclerView становится невидимым, а Chronometer видимым.

При повторном нажатии на кнопку запись останавливается. В памяти устройства создаётся файл с расширением wav. В список добавляется новая запись.

Структура WAV-файла изображена на рисунке 3.9.

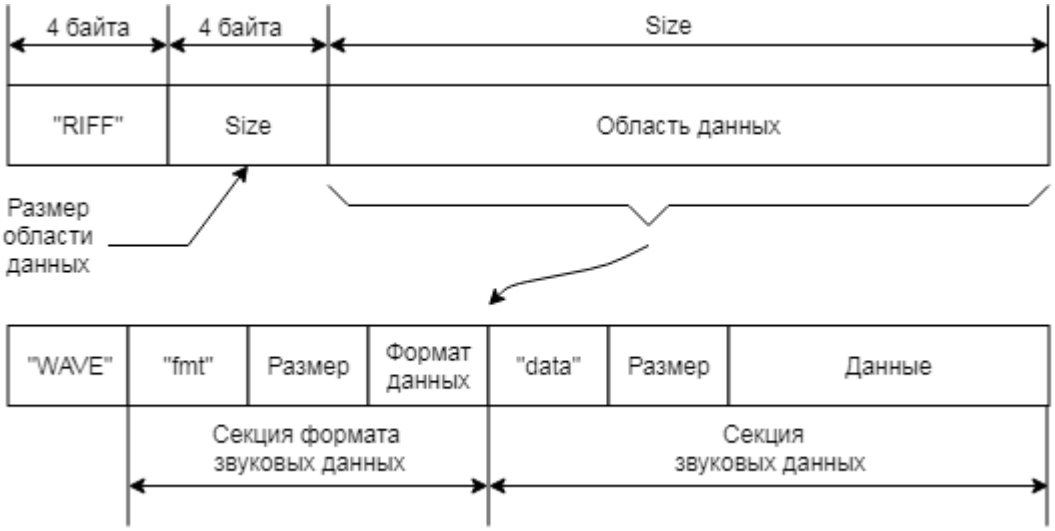

Рисунок 3.9 – Структура WAV-файла

WAV-файл использует стандартную RIFF-структуру (Resource Interchange File Format), которая группирует содержимое файла из отдельных секций (chunks) – формат выборок аудиоданных, аудиоданные, и т. п.

Файл начинается маркером “RIFF” (4 байта), т.к. WAV-файлы являются подмножеством RIFF-формата (Resource Interchange File Format); затем следует поле Size (4 байта), которое указывает размер области данных в байтах без учета размера заголовка. Затем следует область данных.

Область данных начинается маркером “WAVE” (4 байта). Большинство WAV-файлов содержат две секции – секцию формата ("fmt ") и секцию дан-

ных ("data"), которые необходимы для описания формата выборок аудиоданных, и для хранения самих аудиоданных.

Секция формата содержит информацию от том, как сохранены аудиоданные и как они должны воспроизводиться. Информация включает в себя количество каналов, скорость выдачи выборок (sample rate), количество бит в выборке (bits per sample) и другие атрибуты.

Секция данных Wave (Wave Data Chunk) содержит данные цифровых выборок аудиосигнала, которые можно декодировать.

Для работы с WAV-файлами был создан класс WavFile.

Класс WavFile извлекает и обрабатывает информацию из аудиофайла формата wav, в который программа сохраняет аудиопоток входных данных.

Данный класс содержит методы для открытия, закрытия Wav-файла, а также методы для чтения звуковых данных (readSample и readFrame).

Также был создан класс WavFileException, который содержит исключения, необходимые для работы программы.

При нажатии на элемент списка выполняется распознавание выбранного файла. Для распознавания был создан класс MorseProcessor.

Данный класс содержит следующие поля:

1) WavFile wavFile – распознаваемый файл;
2) int numChannels – число каналов;
3) int sampleRate – частота дискретизации;
4) double [] buffer – буфер.

Класс содержит методы:

1) String process();
2) List<MorseSignal> getSignals();
3) MorseSignal readNextSignal();
4) float readSample();
5) List<String> finalize(List<MorseSignal> signals).

Распознавание выполняется в методе process().

Сначала нужно получить последовательность сигналов (точек, тире) из wav файла. Для этого используется метод getSignals(), который возвращает список элементов класса MorseSignal:

В методе getSignals() выполняется поиск сигнала.

Для получения сигнала из потока служит метод readNextSignal() . Сигнал состоит из выборки, длина каждой из которых составляет около 10 мс.

С помощью метода readSample() считывается семплы из wav файла.

Метод readNextSignal() возвращает объект класса Signal.

Класс MorseSignal содержит следующие поля:

1) int length – длина сигнала в семплах;
2) long frame_start – позиция начала сигнала в файле;
3) long frame_end – позиция окончания сигнала в файле;
4) boolean isSignal – сигнал;
5) boolean isSilence – пауза;

Также класс содержит метод length_ms(), позволяющий получить длительность сигнала в мс.

После получения списка сигналов в методе process() вызывается метод finalize(), в котором выполняется преобразование последовательности сигналов в строку, состоящую из точек и тире, сгруппированных по словам.

3.3.3 Настройки

Код окна настроек, содержится в классе RecordActivity. В качестве графического интерфейса RecordActivity устанавливает разметку из файла ресурса разметки дизайна activity_settings.xml (рисунок 3.10).

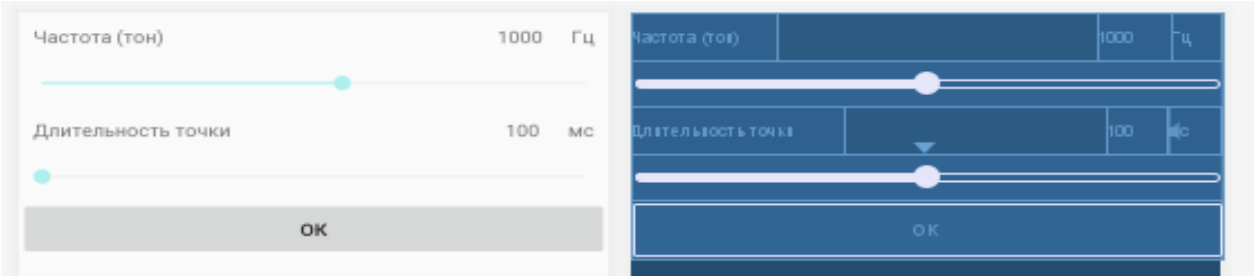

Рисунок 3.10 – activity_settings

В разметке activity_settings.xml используются следующие компоненты:

1. RelativeLayout;
2. TextView;
3. Button;
4. SeekBar.

SeekBar – слайдер с функционалом перетягивания индикатора текущей позиции влево или вправо. Используется для настройки тональности звукового сигнала (от 100 до 2000 Гц) и длительность точки (от 100 до 500 мс).

При нажатии на кнопку выполняется сохранение настроек.

Для хранения настроек используется SharedPreferences.

SharedPreferences — постоянное хранилище в ОС Android, используемое приложениями для хранения своих настроек.

Значения сохраняются в виде пары: ключ, значение.

3.3.4 Обнаружение светового сигнала

Код окна обнаружения светового сигнала, содержится в классе CameraActivity. В качестве графического интерфейса CameraActivity устанавливает разметку из файла ресурса разметки дизайна activity_camera.xml.

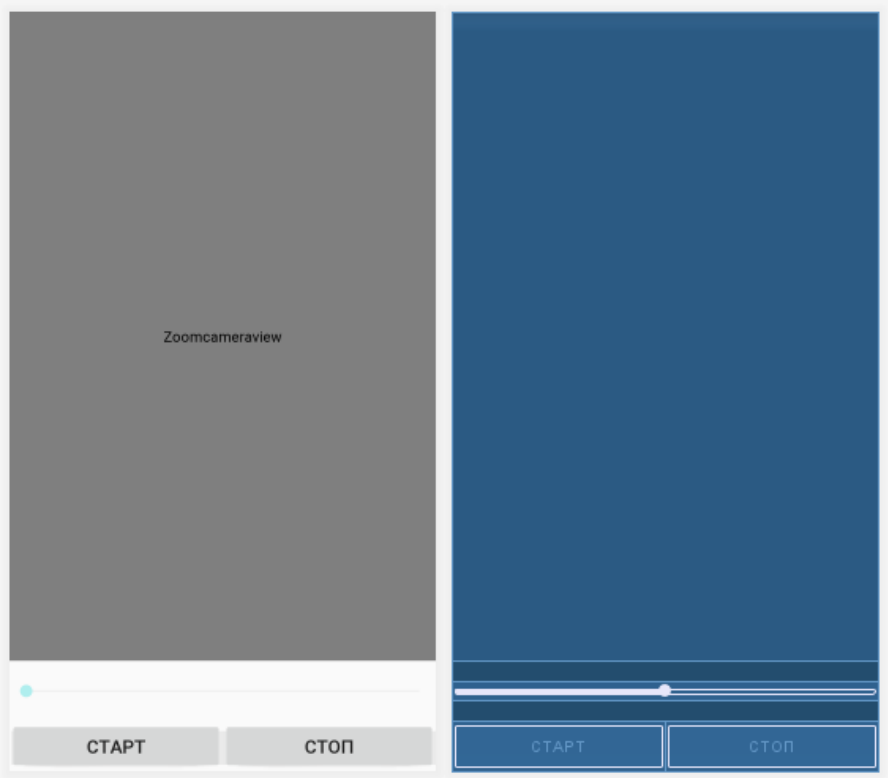

Рисунок 3.11 – разметка activity_camera.xml

В разметке activity_camera.xml используются следующие компоненты:

1) ZoomCameraView;
2) SeekBar;
3) TextView;
4) Button;
5) LinearLayout.

ZoomCameraView – модифицированный компонент для работы с камерой. Унаследован от компонента JavaCameraView.

JavaCameraView реализует только необходимые для работы с камерой устройства. В класс ZoomCameraView добавлена функциональность изменения масштаба. Масштаб изменяется с помощью SeekBar.

JavaCameraView находится в библиотеке OpenCV.

OpenCV (Open Computer Vision) – библиотека компьютерного зрения с открытым исходным кодом, предоставляющая набор типов данных и числен-

ных алгоритмов для обработки изображений алгоритмами компьютерного зрения.

OpenCV позволяет обрабатывать каждый кадр.

Для обнаружения вспышки используется библиотека OpenCV.

Алгоритм работы обнаружения вспышки света состоит из нескольких основных шагов:

1) преобразование кадра в формат HSV;
2) фильтрация методом свертки
3) фильтрация в заданном диапазоне HSV;
4) размытие;
5) детектирование вспышки;

На рисунке 3.12 представлен исходный кадр со вспышкой.

Рисунок 3.12– Исходный кадр

Сначала выполняется преобразование кадра в цветовую модель HSV. Она является нелинейным преобразованием модели RGB (рисунок 3.13).

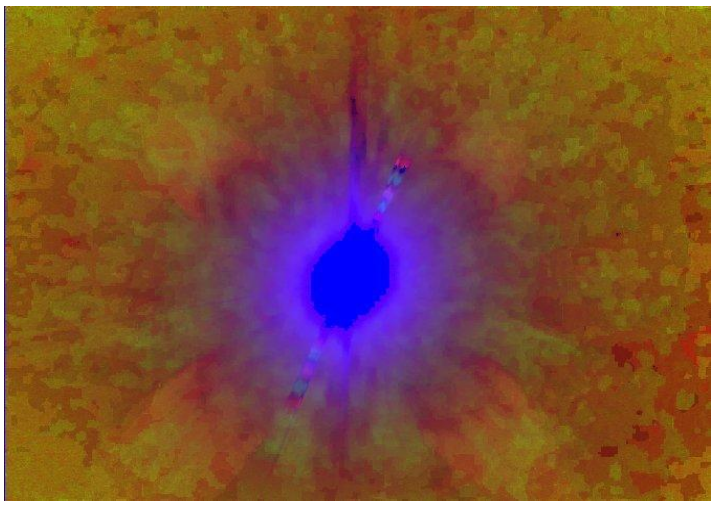

Рисунок 3.13 – Кадр, преобразованный в цветовую модель HSV

В цветовой модели HSV координатами цвета являются:

- Hue — цветовой тон
- Saturation — насыщенность.
- Value (значение цвета) или Brightness — яркость.

Далее выполняется фильтрация методом свертки, с помощью функции filter2D() (рисунок 3.14).

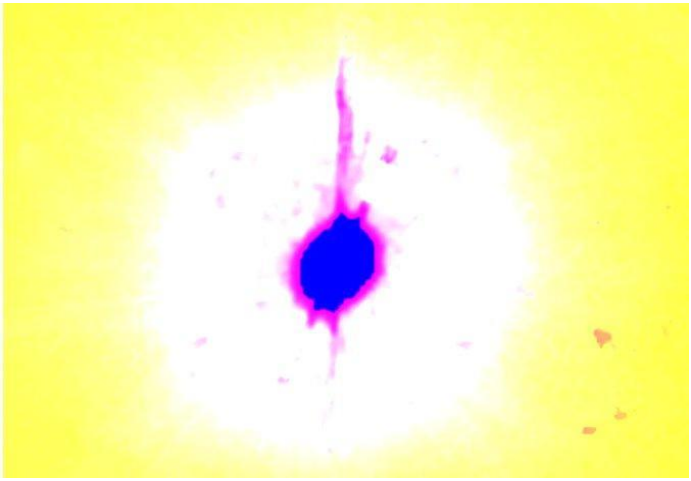

Рисунок 3.14 – Кадр после фильтрации

Затем с помощью функции inRange() на кадр накладывается цветовой фильтр в заданном диапазоне. Этот алгоритм используется для того, чтобы убрать из кадра всё лишнее по цветовому признаку.

Функция inRange() преобразует цветную картинку в черно-белую маску. В этой маске, все пиксели, попадающие в заданный диапазон – становятся белыми. Остальные – черными (рисунок 3.15).

Рисунок 3.15 – Наложение цветового фильтра

Затем выполняется медианное размытие (функция medianBlur()). Размытие необходимо для сглаживания краев.

После размытия применяется детектор границ Канни (Canny).

Границы – это такие кривые на изображении, вдоль которых происходит резкое изменение яркости. Результат работы детектора представлен на рисунке 3.16.

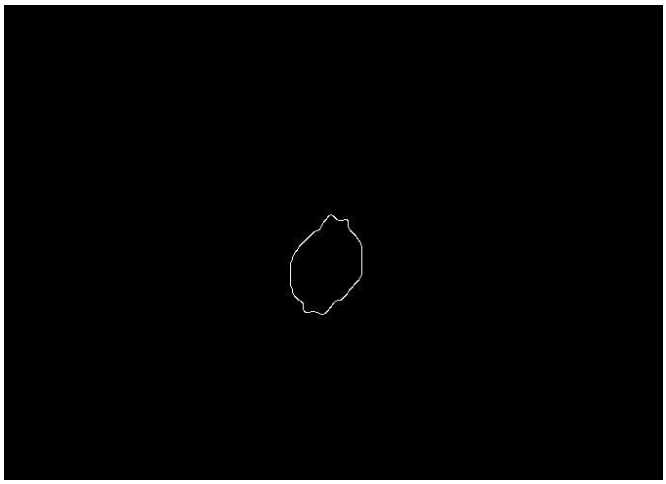

Рисунок 3.16 – Границы вспышки

Далее выполняется поиск контуров. В OpenCV для поиска контуров есть функция findContours().

В параметрах можно указать режим группировки (контуры группируются в многоуровневую иерархию), метод упаковки (склеиваются все горизонтальные, вертикальные и диагональные контуры).

В результате выполнения функции можно получить список всех найденных контуров, представленных в виде векторов и иерархию — информацию о топологии контуров.

Затем находится максимальный контур из полученных с помощью функции findContours() контуров и рисуется в кадре (рисунок 3.17).

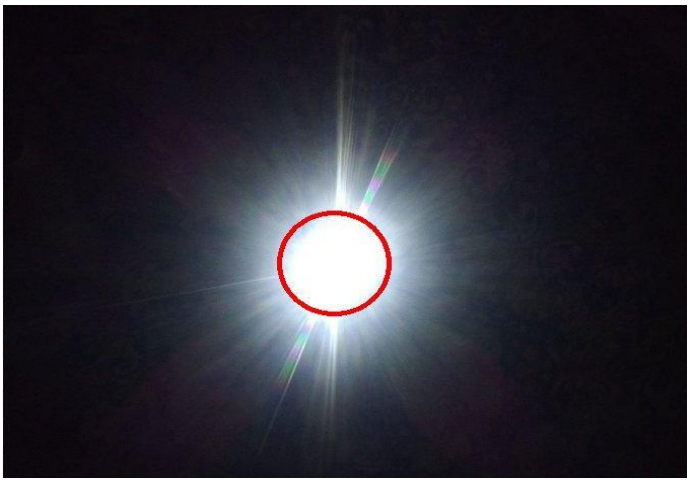

Рисунок 3.17 – Контур вспышки

Для измерения времени был создан вспомогательный класс StopWatch. Класс имеет следующие поля:

1) boolean running – запущен ли отсчет времени;
2) Date startTime – начальный момент времени;
3) Date endTime – конечный момент времени;

Класс содержит следующие методы:

1) void start() – запуск отсчета времени
2) void stop() – остановка отсчета времени
3) boolean isRunning() – проверка, запущен ли отсчет времени
4) long getElapsed() – позволяет получить разницу между начальным и конечным моментом времени

С помощью данного класса измеряется время вспышки и паузы.

3.3.5 Выбор языка

Код окна выбора языка, содержится в классе LanguageActivity. В качестве графического интерфейса LanguageActivity устанавливает разметку из файла ресурса разметки дизайна activity_language.xml (рисунок 3.18).

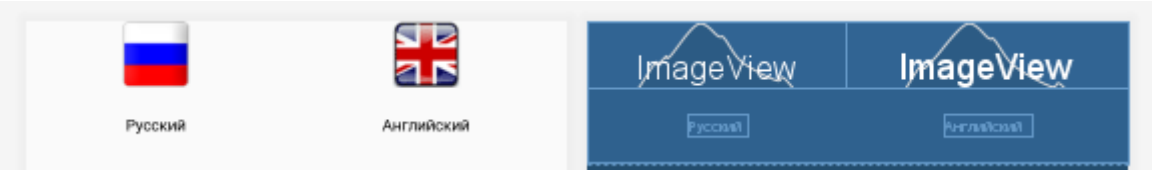

Рисунок 3.18 – activity_language.xml

В разметке activity_language.xml используются следующие компоненты:

1) TableLayout;
2) TableRow;
3) TextView;
4) ImageView;

TableLayout – слой, внутри которого помещаются компоненты и другие слои. Компоненты позиционируются в строки и столбцы.

TableRow представляет собой строку в таблице.

При нажатии на ImageView выполняется переход на MainActivity. С помощью объекта Intent передаётся результат.

3.4.5. Окно справки

Код окна справки, содержится в классе MorseActivity. В качестве графического интерфейса MorseActivity устанавливает разметку из файла ресурса разметки дизайна activity_morse.xml (рисунок 3.19).

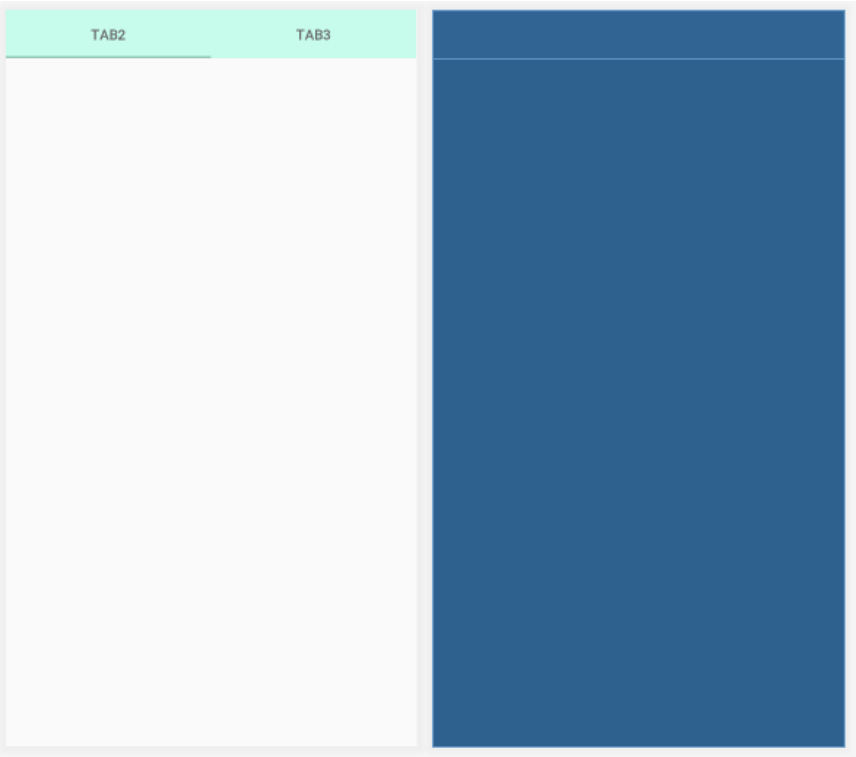

Рисунок 3.19 – activity_morse.xml

В разметке activity_morse.xml используются следующие компоненты:

1) CoordinatorLayout – слой, внутри которого помещаются компоненты и другие слои. Позволяет координировать некие зависимости между включенными в него компонентами.

2) AppBarLayout – это вертикальный LinearLayout, который реализует многие из особенностей материальной концепции дизайна приложения, а именно жесты прокрутки и поведение с определенными параметрами.

3) TableLayot – виджет, позволяющий добавить в приложение вкладки для переключения между фрагментами.

4) ViewPager – компонент, позволяющий организовать просмотр данных с возможностью перелистывания влево-вправо. Данные для отображения компонент берёт из адаптеров.

В компоненте ViewPager находятся ссылки на фрагменты. Сами фрагменты наследуются от android.app.Fragment.

Существует подклассы фрагментов:

- MorseLettersFragment
- MorseDigitFragment.

В разметке morse_letter_fragment.xml используются следующие компоненты:

1)ConstraintLayout;

2)NestedScrollView;

3)ImageView.

ConstraintLayout – контейнер, который является усовершенствованным RelativeLayout. Он позволяет создавать гибкие и масштабируемые интерфейсы.View внутри ConstraintLayout располагаются на основе линий Constraints.Constraints могут быть привязаны к сторонам самого ConstraintLayout или к сторонам других view внутри ConstraintLayout.

NestedScrollView – компонент, который поддерживает прокрутку вложенных в него элементов.

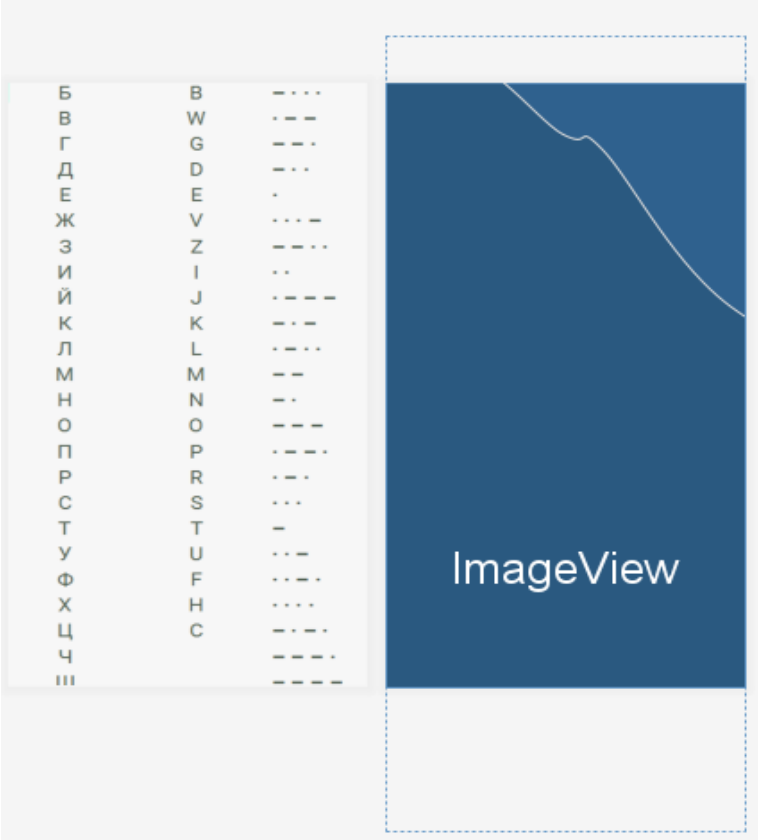

Рисунок 3.20 – фрагмент morse_letter_fragment

4 ТЕСТИРОВАНИЕ ПРИЛОЖЕНИЯ

В данном приложении необходимо протестировать ввод и преобразование текстовых данных, генерацию кода Морзе с помощью звука, вибрации, светового сигнала, распознавание светового и звукового сигналов.

Тестирование проводилось на смартфоне MeizuPro 7 со следующими техническими и аппаратными характеристиками:

1. Размер экрана: 5.2 дюйма;
2. Разрешение экрана: 1920*1080 px;
3. Процессор: MediaTekHelio P25
4. Частота процессора: 2600 МГц;
5. Число ядер: 8;
6. Разрешение камеры 12,0 Мп (4000 * 3008)
7. Объем встроенной памяти: 64 Гб;
8. Объем оперативной памяти: 4 Гб;
9. Операционная система: Android;
10. Версия ОС: 7.0.

Результат кодирования текста на русском языке представлен на рисунке 4.1 (слева).

Данный рисунок отражает, тот факт, что программа корректно распознает кириллицу и преобразует ее на язык Морзе.

Текст вводится с помощью клавиатуры смартфона на русском или английском языке. Введённый текст можно удалить с помощью кнопки .

За минимальную единицу сигнала рассматривается одна точка. Тире соответствует трем точкам. Пауза, используемая для разделения слов в предложении принимается равной семи точкам.

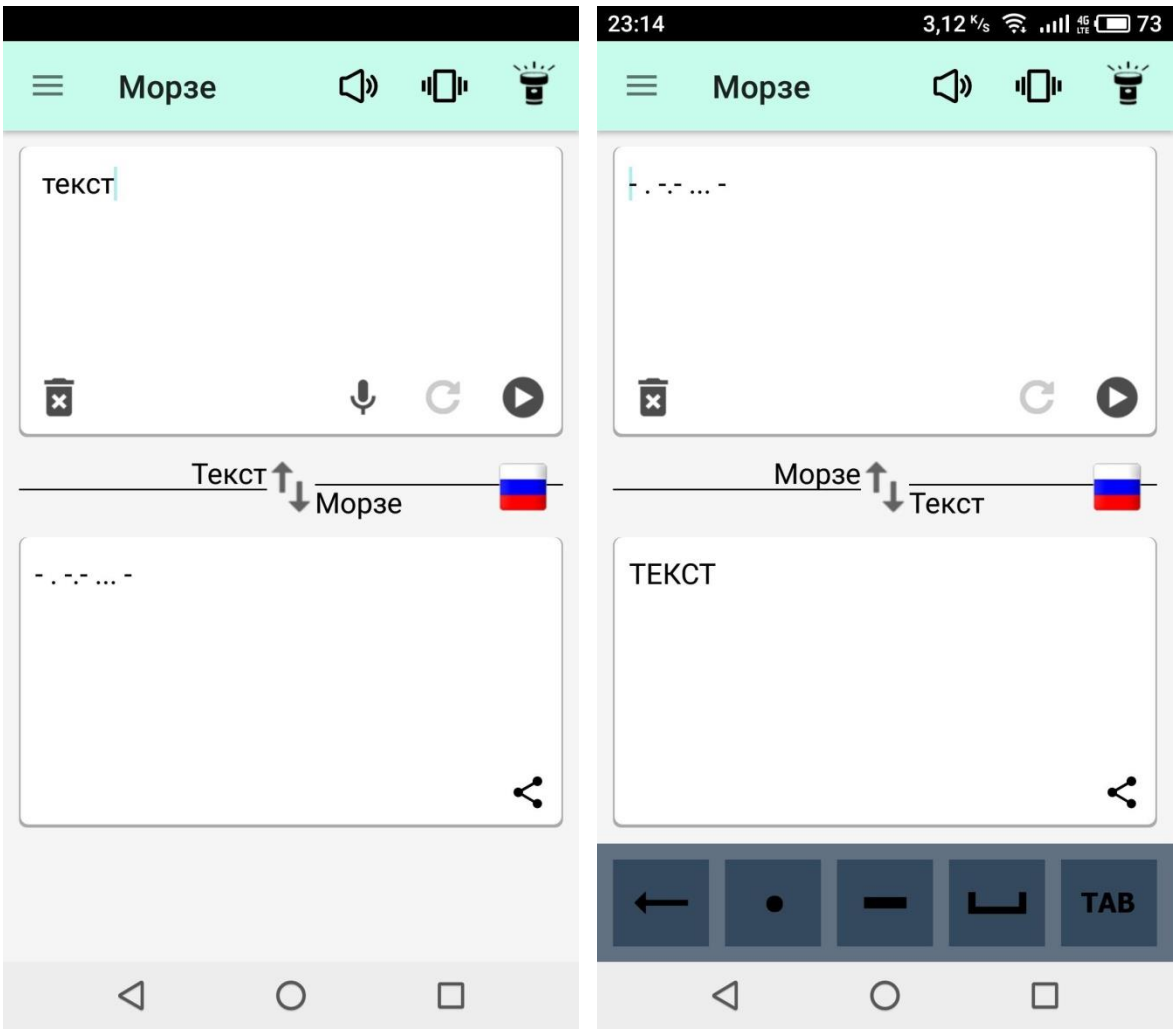

Рисунок 4.1 – Кодирование текста на русском языке (слева) и результат обратного преобразования текста (справа)

Приложение также работает и в обратную сторону. Результат обратного преобразования текста с кода Морзе на кириллицу представлен на рисунке 4.1 (справа).

Код Морзе вводится с помощью специальных кнопок, расположенных внизу экрана: отмена введенного символа ←, минимальная единица сигнала «точка» •, длительная единица сигнала «тире» —, пробел между словами ␣, перенос на новую строку TAB.

В приложении доступен голосовой ввод текста (рисунок 4.2).

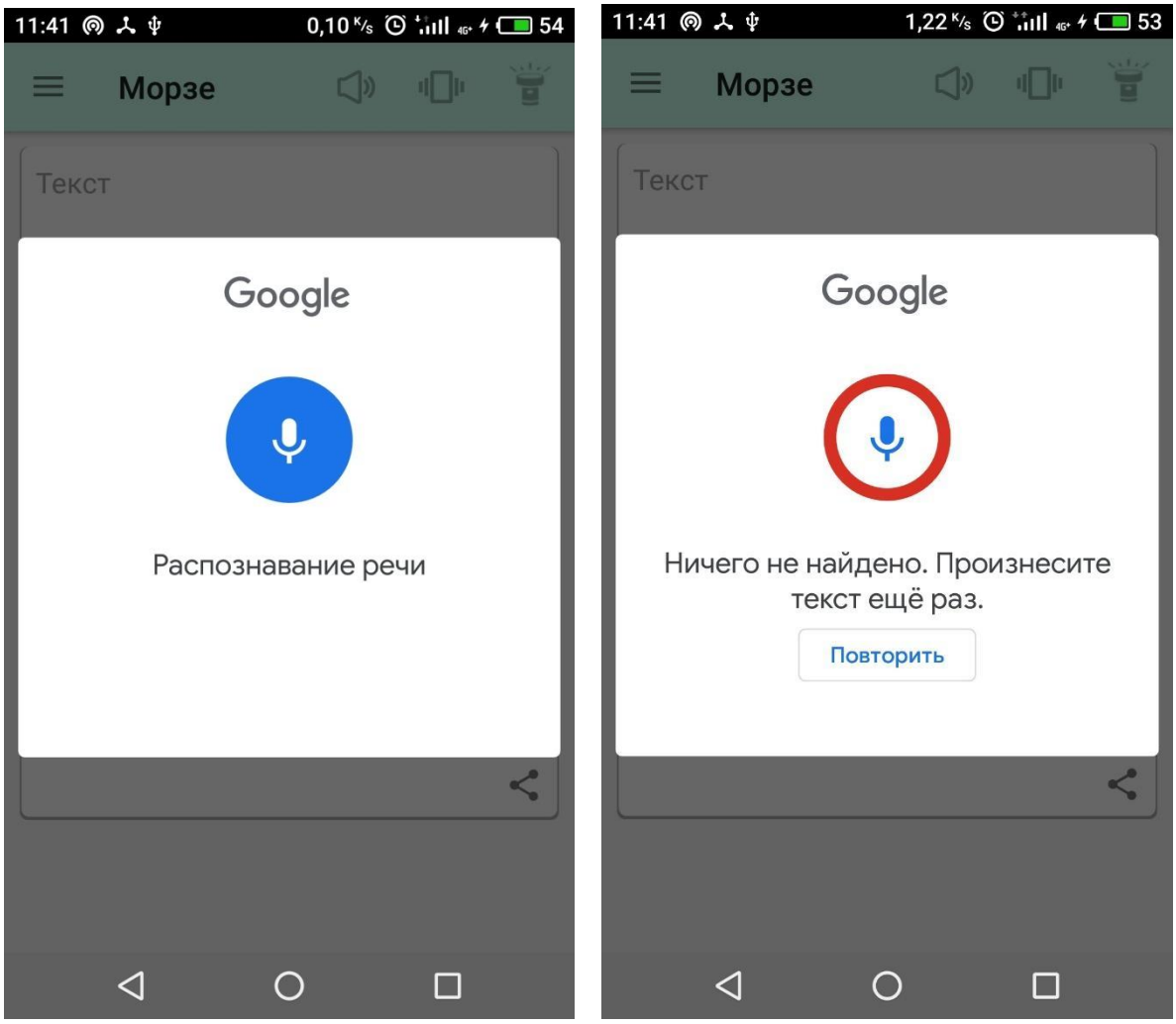

Рисунок 4.2 – Голосовой ввод текста (слева) и предложение повторения голосового ввода (справа)

В случае, если процедура распознавания речи не сработала, предлагается перезапустить распознавание речи.

Для распознавания речи не нужен доступ к интернету.

Можно выбрать язык записываемого текста. Для выбора доступны английский и русский язык (рисунок.4.3).

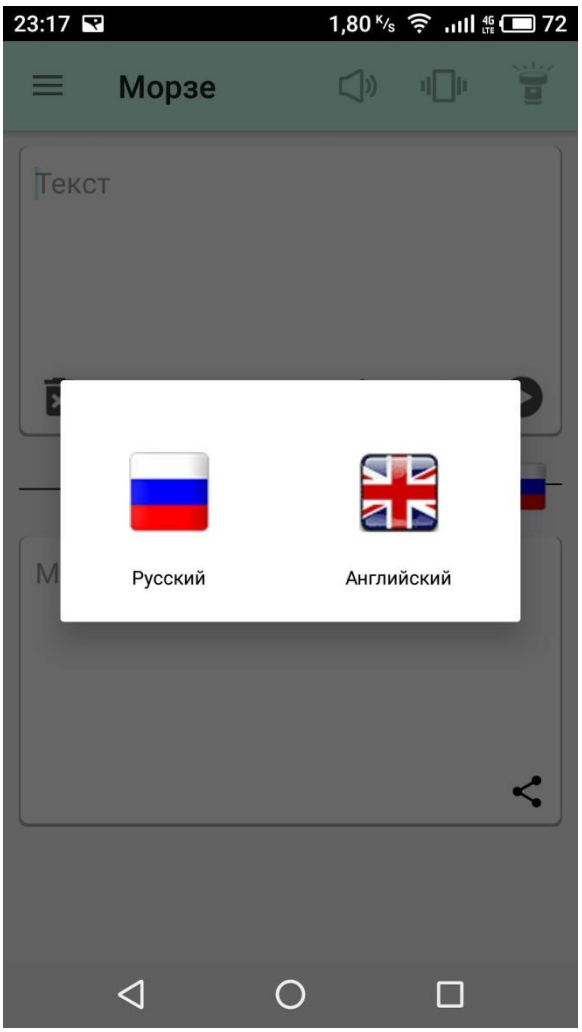

Рисунок 4.3 – Окно выбора языка

На рисунке 4.4 представлен результат кодирования текста на английском языке.

Способы передачи сообщения можно сочетать между собой в любых вариантах, а можно выбирать по отдельности.

В верхнее поле вводятся буквы, цифры и знаки препинания на выбранном языке, затем нажимается кнопка . Программа поместит азбуку Морзе в нижнее поле или выдаст соответствующее сообщение, если введенный текст не может быть переведен. Можно остановить передачу сообщения, нажав кнопку .

Можно зациклить передачу сообщения. Для этого нужно нажать кнопку .

При генерации внизу экрана отображается слово и код Морзе. Символ, который передается в текущий момент, выделяется красным цветом (рисунок 4.4). Это удобно, чтобы наблюдать за корректностью передаваемого сообщения.

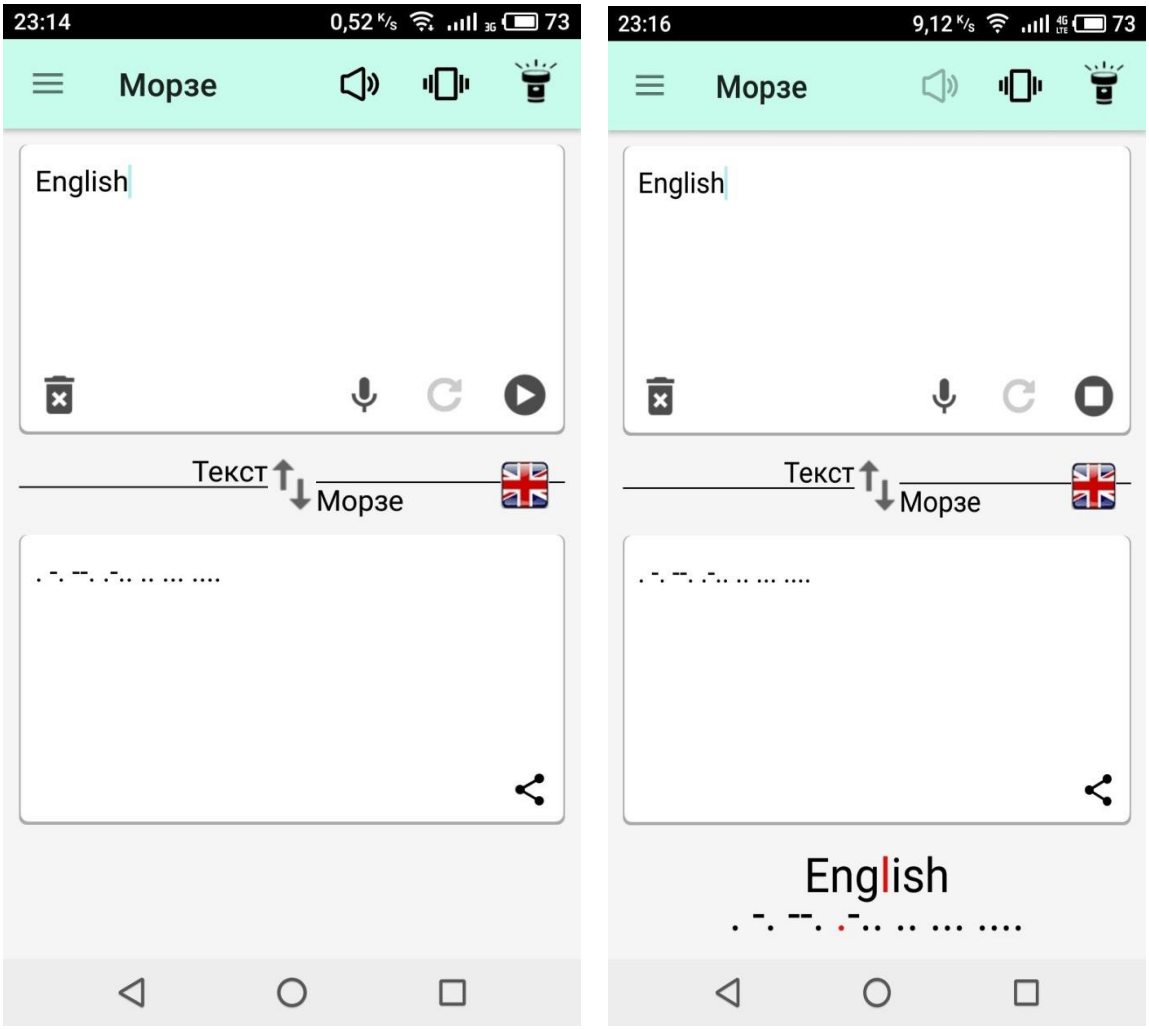

Рисунок 4.4 – Кодирование текста на английском языке (слева) и передача сообщения в реальном времени (справа)

На рисунке 4.5 показано выдвигающееся меню для перехода на другие окна. Выдвигающееся меню содержит следующие пункты:

- Световой сигнал;
- Распознавания звука;
- Скорость и частота;
- Азбука Морзе.

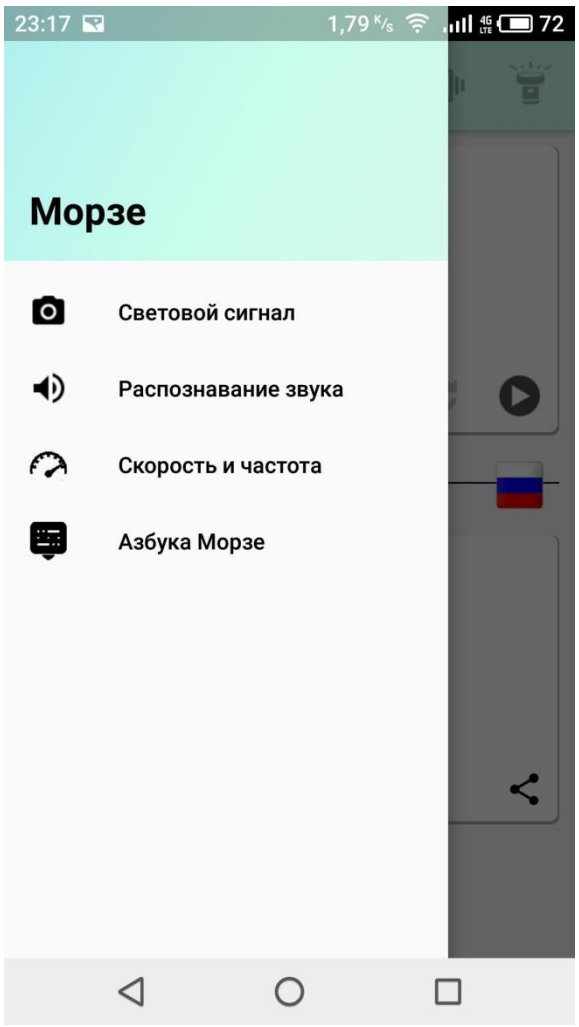

Рисунок 4.5 – Выдвигающееся меню

Программа может передавать и распознавать сообщения в текстовом, звуковом или световом форматах, или сочетать эти форматы по желанию пользователя. Также для справочной информации приводится азбука Морзе, что удобно для самостоятельного изучения или проверки корректности сообщений.

На рисунке 4.6 представлено окно настроек. Пользователь может выбрать длительность точки в миллисекундах, и задать частоту звукового вывода в герцах.

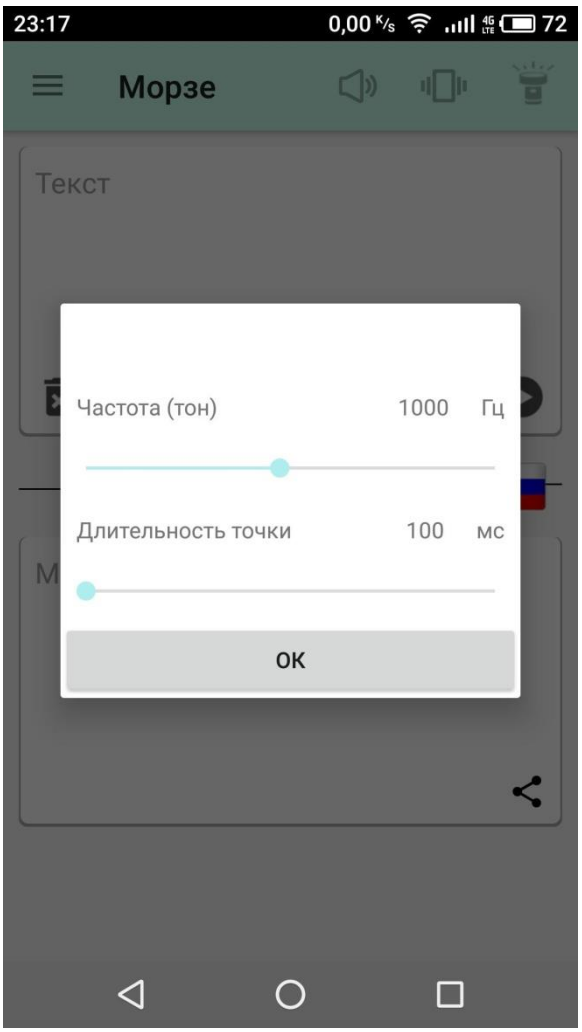

Рисунок 4.6 – Окно настроек

Можно увеличивать диапазон частот, отрегулировав минимальные и максимальные частоты с помощью клавиши управления громкостью звука на смартфоне. Регулируя минимальную и максимальную громкость, пользователь может отфильтровать нежелательный фоновый шум (например, увеличив минимальную громкость до -60 дБ).

Для преобразования сигнала требуется извлечь тоны сигнала азбуки Морзе из звука, поступающего в микрофон. Для этого желательно, чтобы тоны Морзе были самыми громкими звуками, которые будут получены. В противном случае искажается обработка звука и необходимо провести регулировку частот, как описано выше.

На рисунке 4.7 представлено окно для записи звука.

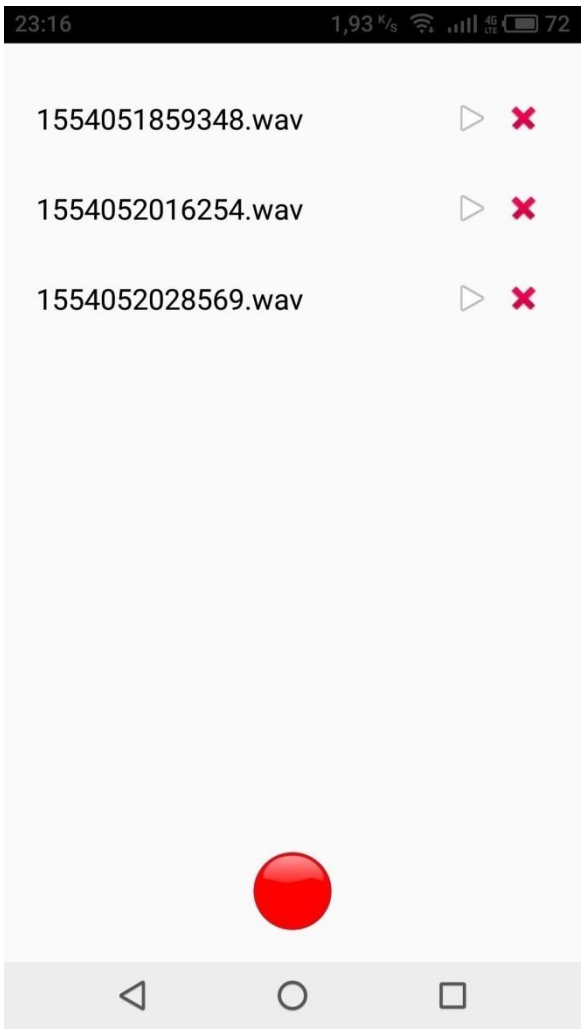

Рисунок 4.7 – Окно записи звука

В окне отображается список записей, имеющихся в наличии. Каждую запись можно прослушать или удалить.

На рисунке 4.7 представлены некоторые предварительно записанные файлы, по которым проводился анализ. Первый образец 1554051859348.wav был загружен из интернета и, следовательно, имеет чистый, неискаженный звук. Второй, 1554052016254.wav был записан через микрофон, с устройства действительно посылающего азбуку Морзе. Третий, 1554052028569.wav, представляет собой записанный отрывок телепрограммы.

Анализ показал, что звуки, обработанные через микрофон, дают больше помех, чем чистые образцы из-за наличия шума.

На рисунке 4.8 показана процедура записи звукового сообщения в реальном времени. При нажатии на кнопку ● начинается запись звука и запускается отсчет времени.

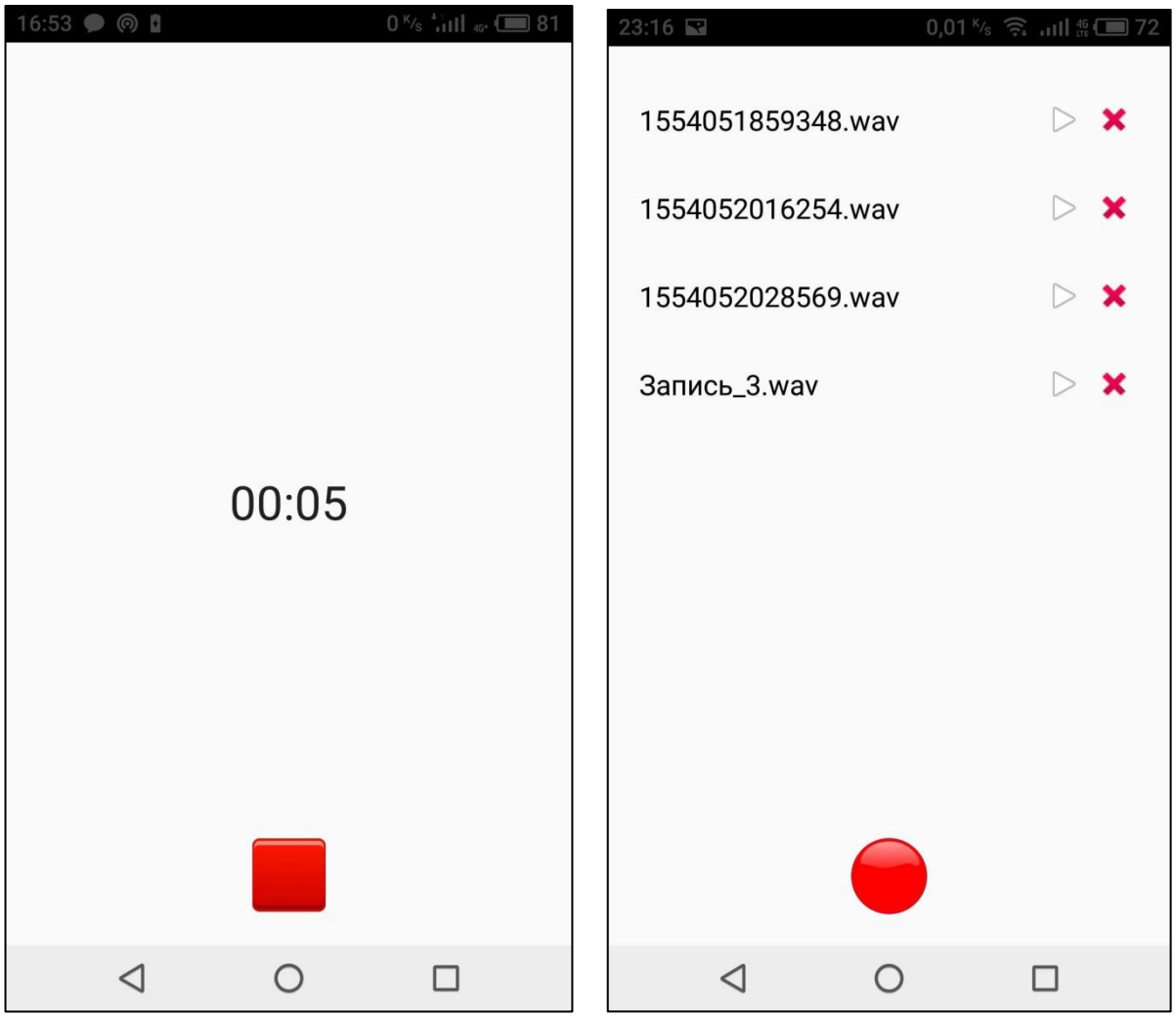

Рисунок 4.8 – Запись звука (слева) и добавление новой записи (справа)

При нажатии на кнопку ■ запись звука останавливается. В памяти устройства создаётся файл с расширением wav. В список добавляется новая запись.

На рисунках 4.9 представлено окно справки.

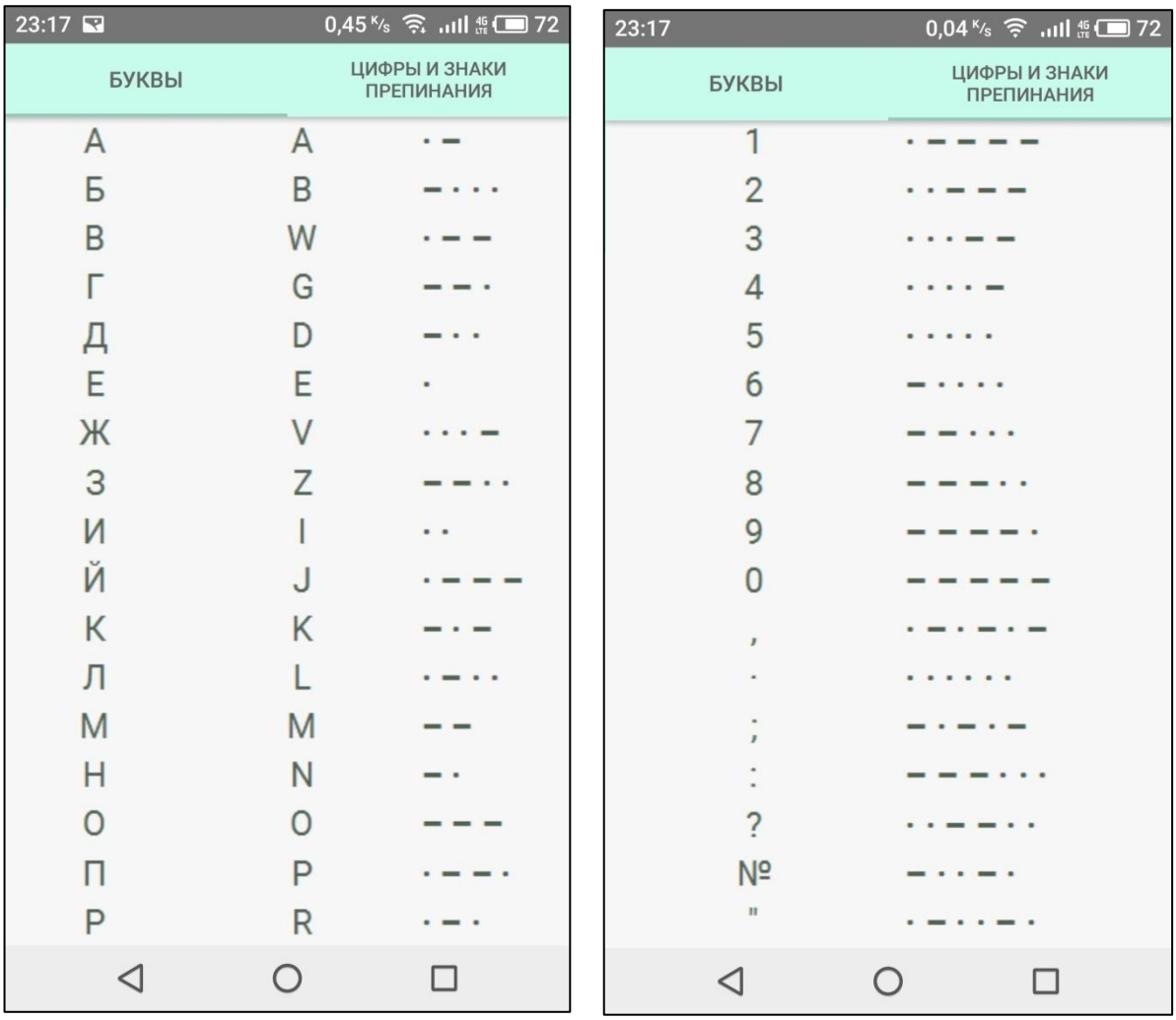

Рисунок 4.9 – Окно справки: буквы (слева); цифры и знаки препинания (справа)

Данное окно можно использовать для изучения азбуки Морзе самостоятельно.

Окно справки содержит две вкладки.

На первой вкладке находится таблица с буквами на русском и английском языке.

На второй вкладке находится таблица с цифрами и знаками препинания.

В таблицах приведён, соответствующий символам код Морзе.

На рисунке 4.10 представлено окно распознавания светового сигнала.

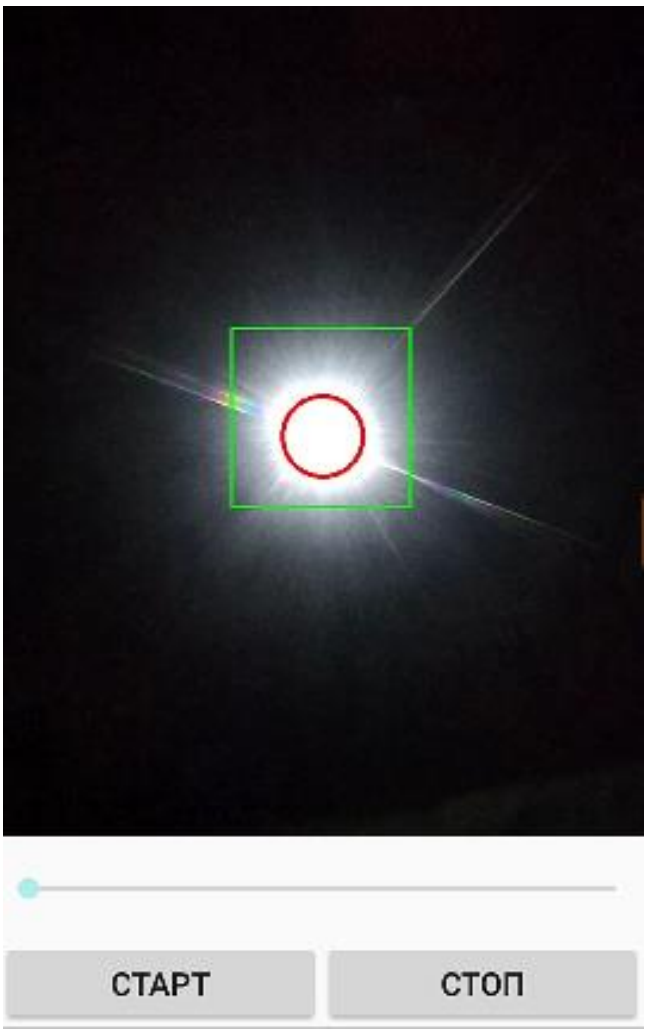

Рисунок 4.10 – Распознавание светового сигнала

. Для запуска распознавания нужно нажать кнопку “Старт” и навести камеру на вспышку так, чтобы свет оказался в рамках небольшого квадрата, как указано на рисунке.

Приложение подсчитывает количество кадров, на которых подается сигнал, и в зависимости от длительности этого сигнала, он распознается как точка или тире. Пока сигнал принимается, он декодируется в текстовое сообщение и накапливается в памяти смартфона.

Когда пользователь нажимает кнопку «Стоп», все сообщение выводится в текстовое поле.

В приложении есть возможность приблизить вспышку.

При выходе из приложения срабатывает защита от случайного выхода. На рисунке 4.11 изображено окно с подтверждением выхода.

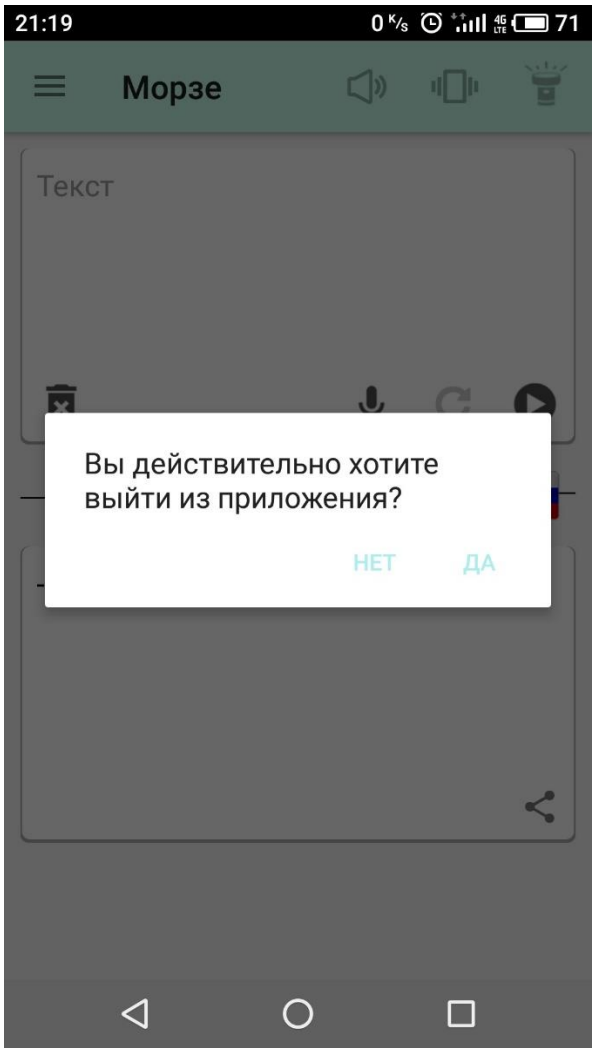

Рисунок 4.11 –Окно подтверждения выхода из приложения

При клике на соответствующую кнопку в окне, будет произведен выход из приложения, либо сворачивания окна подтверждения. При клике не в области окна засчитывается клик на кнопку «Нет».

ЗАКЛЮЧЕНИЕ

В ходе работы над выпускной квалификационной работой были получены следующие результаты:

- был проведен анализ существующих аналогов, рассмотрены их достоинства и недостатки, выполнено сравнение;
- для реализации приложения была выбрана ОС Android, которая является наиболее популярной среди мобильных операционных систем;
- на основе статистики Google, в которой оценивается количество устройств с различными версиями операционной системы Android, минимальной версией ОС, поддерживаемой приложением была выбрана версия Android 4.4 KitKat;
- в результате ознакомления со средствами разработки для ОС Android, была выбрана Android Studio IDE, поскольку данная среда разработки бесплатна, обладает широкой функциональностью, множеством полезных инструментов для разработки под Android, доступностью интерфейса и простотой настройки;
- в результате рассмотрения аналогов, выбора операционной системы и среды разработки, были сформированы требования к функциональности приложения;
- с помощью интегрированной среды разработки Android Studio на языке программирования java и языке разметки xml было разработано приложение, удовлетворяющее сформированным требованиям;
- приложение было протестировано на работоспособность;

Все поставленные задачи были решены, достигнута цель ВКР – разработано мобильное по генерации и распознаванию кода Морзе.

Разработанное приложение отличается от аналогов тем, что оно позволяет не только генерировать, но и распознавать звуковой и световой сигнал, поддерживает и латиницу, и кириллицу. Также в приложении доступен голосовой ввод и отсутствует реклама.

Разработанное приложение можно улучшать, добавляя новые функции и совершенствуя работу существующих.

Можно добавить следующие функции:

- поддержка нескольких языков;
- автоматическое определение языка вводимого текста;
- интерфейс на английском языке;
- самоучитель азбуки Морзе.

Приложение может служить платформой для создания прочих приложений по сообщению различными визуальными или звуковыми кодами.

Дальнейшим этапом развития приложения может стать разработка версии для операционной системы iOS, так как согласно статистике, на данный момент она вторая по популярности среди мобильных операционных систем.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Блох Д. Java. Эффективное программирование – М.: Лори, 2014 – 310 с.
2. Борисенко И.Г. Прием на слух радиотелеграфных знаков. Пособие для обучения летного состава. – М.: ДОСААФ, 1971. – 40 с.
3. Голощапов А.Л. Google Android. Создание приложений для смартфонов и планшетных ПК. 2-е изд., перераб. и доп. – Спб.: БХВ-Петербург, 2014.
4. Ёранссон А. Эффективное использование потоков в операционной системе Android. – М.: ДМК Пресс, 2015 год – 304 с.
5. Кинтцель Т. Руководство программиста по работе со звуком – М.: ДМК Пресс, 2000 – 432 с.
6. Михалин,В.А. Методические рекомендации по первоначальному обучению телеграфной азбуке – Ульяновск: Центр ГА СЭВ, 1990. – 44 с.
7. Прохоренок Н. А. П84 OpenCV и Java. Обработка изображений и компьютерное зрение. – СПб.: БХВ-Петербург, 2018 – 320 с.
8. Сато Ю. Обработка сигналов. первое знакомство. / Пер. с яп.; под ред. Ёсифуми Амэмия. – М.: Издательский дом «Додэка-XXI», 2002, С. 15–18.
9. Скляр Б. Цифровая связь. Теоретические основы и практическое применение. Пер. с англ. – М.: Издательский дом «Вильямс», 2003, 1104 с.
10. Уорбертон Р. Лямбда-выражения в Java 8 – М.: ДМК Пресс, 2017 – 192 с.
11. Филлипс Б., Стюарт К., Марсикано К.Android. Программирование для профессионалов. 3-е издание – СПб.: Издательский дом «Питер», 2017 год – 688 с.
12. Хацкевич Ю.А. Сборник упражнений и тренировочных текстов для передачи ключом и датчиком кода Морзе. – Томск: Издательский Дом Томского государственного университета, 2018. – 56 с.

13. Шилдт Г. Java. Полное руководство. 10-е издание – Киев: Диалектика, 2018 – 1488 с.

14. Howse, J. Android Application Programming with OpenCV – Packt Publishing, 2013 –130 p.

15. Android Arsenal URL: https://android-arsenal.com.

16. Android Studio // Android Developers URL: https://developer.android.com/studio.

17. Architecture Components // Start Android URL: https://startandroid.ru/ru/courses/architecture-components.html.

18. Distribution dashboard // Android Developers URL: https://developer.android.com/about/dashboard.

19. Documentation for app developers // Android Developers URL: https://developer.android.com/

20. Eclipse IDE // Eclipse Foundation URL: https://www.eclipse.org/eclipseide/

21. Google Play Store. Главная страница магазина приложений. URL: https://play.google.com/store/apps.

22. Guava: Google Core Libraries for Java // GitHub URL: https://github.com/google/guava

23. IntelliJ IDEA // JetBrains URL: https://www.jetbrains.com/idea/

24. Material Design URL: https://material.io/

25. Mobile Operating System Market Share Worldwide // Statcounter URL: http://gs.statcounter.com/os-market-share/mobile

26. OpenCV URL: https://opencv.org/

27. OpenCV шаг за шагом // RoboCraft URL: http://robocraft.ru/page/opencv/

28. Солнцев А. Почему IDEA лучше Eclipse URL: https://habrahabr.ru/post/112749/

29. Пример создания Navigation Drawer в Android // JavaDevBlog.com URL:https://javadevblog.com/primer-sozdaniya-navigation-drawer-v-android.html

30. Про азбуку Морзе // Habr URL: https://habr.com/ru/company/megafon/blog/193090/

31. Программирование под ОС Андроид // METANIT.COM URL: https://metanit.com/java/android/

32. Разработка под Android // Habr URL: https://habr.com/ru/hub/android_dev/

33. Разработка под Android // Сайт Александра Климова URL: http://developer.alexanderklimov.ru/android/

34. Структура WAV файла // Audio Coding URL: https://audiocoding.ru/article/2008/05/22/wav-file-structure.html

35. Уроки по Android // Start Android URL: https://startandroid.ru/ru/uroki/vse-uroki-spiskom.html.